UN/CEFACT CCTS BASED E-BUSINESS DOCUMENT DESIGN AND CUSTOMIZATION ENVIRONMENT FOR ACHIEVING DATA INTEROPERABILITY

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

FULYA TUNÇER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

JUNE 2009

Approval of the thesis:

UN/CEFACT CCTS BASED E-BUSINESS DOCUMENT DESIGN AND CUSTOMIZATION ENVIRONMENT FOR ACHIEVING DATA INTEROPERABILITY

submitted by **FULYA TUNÇER** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering , Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, **Graduate School of Natural and Applied Sciences** _____

Prof. Dr. Müslim Bozyiğit
Head of Department, **Computer Engineering** _____

Prof. Dr. Asuman Doğaç
Supervisor, **Department of Computer Engineering, METU** _____

Prof. Dr. Müslim Bozyiğit
Co-supervisor, **Department of Computer Engineering, METU** _____

Examining Committee Members:

Prof. Dr. İsmail Hakkı Toroslu
Department of Computer Engineering, METU _____

Prof. Dr. Asuman Doğaç
Department of Computer Engineering, METU _____

Prof. Dr. Özgür Ulusoy
Department of Computer Engineering, Bilkent University _____

Assoc. Prof. Dr. Ahmet Coşar
Department of Computer Engineering, METU _____

Yıldıray Kabak
SRDC Ltd. _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: FULYA TUNÇER

Signature :

ABSTRACT

UN/CEFACT CCTS BASED E-BUSINESS DOCUMENT DESIGN AND
CUSTOMIZATION ENVIRONMENT FOR ACHIEVING DATA INTEROPERABILITY

Tunçer, Fulya
M.S., Department of Computer Engineering
Supervisor : Prof. Dr. Asuman Doğaç
Co-Supervisor : Prof. Dr. Müslim Bozyiğit

June 2009, 106 pages

The leading effort for creating a standard semantic basis for business documents to solve the electronic business document interoperability problem came from the UN/CEFACT (United Nations Centre for Trade Facilitation and Electronic Business) Core Components Technical Specification (CCTS) through a conceptual document modeling methodology.

Currently, the main challenge in using UN/CEFACT CCTS based approaches is that the document artifacts are stored in spreadsheets and this makes it very difficult to discover the previously defined components and to check their consistency. Furthermore, businesses need to customize standard documents according to their specific needs. The first XML implementation of UN/CEFACT CCTS, namely, Universal Business Language (UBL) provides detailed text-based descriptions of customization mechanisms. However, without automated tool support, it is difficult to apply the customization and to maintain the consistency of the customizations.

In this thesis, these problems are addressed by providing an online e-business document design and customization environment, i.e. iSURF eDoCreator, which integrates the machine

processable versions of paper-based UN/CEFACT CCTTS modeling methodology and UBL customization guidelines, accompanied with an online common UN/CEFACT CCTS based document component repository. In this way, iSURF eDoCreator environment aims to maximize re-use of available document building blocks and minimize the tedious document design and customization efforts. The environment also performs the gap analysis between different customizations of UBL to show how interoperable is the compared document models.

The research leading to these results has received funding from the European Community's FP7/2007-2013 under grant agreement n° 213031, the iSURF Project.

Keywords: eBusiness, Document Modeling, Document Customization, data interoperability, UBL, UN/CEFACT CCTS

ÖZ

VERİ BİRLİKTE İŞLERLİĞİ İÇİN UN-CEFACT CCTS TABANLI ELEKTRONİK DOKÜMAN DİZAYN VE KİŞİSELLEŞTİRME ORTAMI

Tunçer, Fulya
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Asuman Doğaç
Ortak Tez Yöneticisi : Prof. Dr. Müslim Bozyiğit

Haziran 2009, 106 sayfa

Elektronik doküman birlikte işlerliğini sağlamak için gerekli olan standart anlamsal bir taban oluşturma girişimi UN/CEFACT CCTS (Birleşmiş Milletler İdari, Ticari ve Ulaşımla İlgili Uygulama ve Usulleri Kolaylaştırma Merkezi Esas Parçalar Teknik Spesifikasyonu) liderliğinde önerilen doküman modelleme methodolojisiyle başlamıştır. Fakat şu anda UN/CEFACT CCTS tabanlı sistemlerin yeterince yaygınlaşmamasına bağlı olarak elektronik doküman birlikte işlerliği hala sağlanamamıştır. Bu yaygınlaşma sürecenin aksamasındaki sebeplerden başlıcaları oluşturulan doküman parçalarının çizelgelerde tutulmasından kaynaklı yarartılmış olan doküman parçacıkların bulunamaması ve tutartlılık kontrollerinin yapılamamasıdır.

Ayrıca UN/CEFACT CCTS'in ilk XML geçekleştirilmesi olan UBL (Universal Business Language - Evrensel İş Dili) doküman kişiselleştirmesi için yazı tabanlı detaylı yönergeleri sunmaktadır. Fakat otomatik bir araç desteği olmadan, bu yönergeleri uygulamak ve uygulanan hareketlerin yönergelerle tutarlılığını kontrol etmek çok zordur. Bu tezde, özde kısaca bahsedilen problemleri çözmek için UN/CEFACT CCTS'in doküman modelleme ve UBL'in kişiselleştirme yönergelerinin makine işlenebilir hale getirip birleştiren çevirimiçi elektronik

iş dokümanı dizayn ve kişileştirme ortamı ve çevirimiçi UN/CEFACT CCTS tabanlı doküman parçacık deposu sunulmaktadır. Bu sayede varolan doküman parçacıklarını yeniden kullanılmasının artırılması ve zaman ve dikkat gerektiren doküman dizayn ve kişiselleştirme işlemini kolaylaştırılması hedeflenmektedir. Ayrıca ortam UBL dokümanlarının farklı kişiselleştirmeleri arasında ayrım çözümlenmesi yapılmasını sağlayarak iki doküman versiyonunun ne kadar birlikte işler olduğunu da göstermeyi hedeflemektedir.

Tezde sunulan bu çalışma bir Avrupa Komisyonu ICT FP7 projesi olan IST-213031 iSURF projesinin bir parçası olarak gerçekleştirilmiştir.

Anahtar Kelimeler: e-İş, Doküman Modelleme, Doküman Kişiselleştirme, veri birlikte işlerliği, UBL, UN-CEFACT CCTS

To my family

ACKNOWLEDGMENTS

I would like to express my sincere gratitude and appreciation to Prof. Dr. Asuman Doğaç for her encouragement and support throughout this study.

I would also like to express gratitude to my co-supervisor, Prof. Dr. Müslim Bozyiğit for his guidance and support during my study.

I am deeply grateful to my family for their love and support. Without them, this work could not have been completed.

I am deeply indebted to my friends, Gökçe Banu Laleci Ertürkmen and Yıldıray Kabak for sharing their deep expertise throughout this thesis work. Furthermore, I thank Erdem Alpay, Şenan Postacı, Suat Gönül, Server Çimen and all the other colleagues at the Software Research and Development Center, whose help, stimulating suggestions and encouragement helped me in all the time of research for and development of this thesis.

I would thank the Scientific and Technological Research Council of Turkey (TÜBİTAK) for providing the financial means throughout this study.

Finally, my special thanks go to my friends, İrem Uslu, Işıl Şekerci and Gözde Özbal for their help, support and cheerful presence not only through the course of this study but also any course of my life. Thanks for giving me a shoulder to lean on whenever I need.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ABIE	Aggregate Business Information Entity
ACC	Aggregate Core Component
API	Application Programming Interface
ASBIE	Association Business Information Entity
BBIE	Basic Business Information Entity
BCC	Basic Core Component
BIE	Business Information Entity
CC	Core Component
CCL	Core Component Library
CCTS	Core Components Technical Specification
CDT	Core Data Type
CII	UN/CEFACT Cross Industry Electronic Invoice
CODICE	COmponentes y Documentos Interoperables para la Contratacion Electronica (Interoperable Components and Documents for Electronic Procurement)
COPI	Classification Of the purposes of non Profit Institutions serving households
CPFR	Collaborative Planning, Forecasting and Replenishment
CXML	commerce eXtensible Markup Language
EDI	Electronic Data Interchange
EFM	Electronic Freight Management
FP7	Seventh Framework Programme

GDSSU	Global Data Synchronization Service Utility
GENESIS	Integration for SMEs, Governmental Organizations and Intermediaries in the New European Union
GIB	Revenue Administration (Gelir İdaresi Başkanlığı)
GS1	Global Standards One
GUI	Graphical User Interface
ICT	Information Communication Technologies
ISIC	International Standard Industrial Classification
ISO	International Organization for Standardization
ISURF	An Interoperability Service Utility for Collaborative Supply Chain Planning across Multiple Domains Supported by RFID Devices
JDBC	Java Database Connectivity
MIME	Multipurpose Internet Mail Extensions
NDR	Naming and Design Rules
NES	North European Subset
OAGIS	The Open Applications Group Integration Specification
OASIS	Organization for the Advancement of Structured Information Standards
ODS	Open Document Spreadsheet
OIOUBL	Offentlig Information Online UBL
POJOS	Plain Old Java Objects
QBDT	Qualified Business Data Type
QDT	Qualified Data Type
RFID	Radio Frequency Identification
SITC	Standard International Trade Classification
SME	Small and Medium Enterprises

SOAP Simple Object Access Protocol

SQL Structured Query Language

UBL Universal Business Language

UDT Unqualified Data Type

UML Unified Modeling Language

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

UNSD United Nations Statistics Division

UNSPSC Universal Standard Product and Service Specification

URI Uniform Resource Identifier

URL Uniform Resource Locator

W3C World Wide Web Consortium

WCO World Customs Organization

XCBL XML Common Business Library

XLS Microsoft Excel file format, a spreadsheet file format

XML Extensible Markup Language

XPATH XML Path Language

XSD XML Schema Definition

XSL XML Stylesheet Markup Language

CHAPTER 1

INTRODUCTION

In order to realize a collaborative business process over the Internet, first of all participating parties should agree on the information content and semantics of the documents that are exchanged among enterprises [1]. Documents include purposeful and self contained packages of information and are used for this purpose for nearly thousands of years.

In virtual collaborations, e-Business document schemas provide business interfaces among trading partners to agree on information content and loosely couple collaborating systems [2]. Interoperability is the ability of two or more systems or components to exchange information and to use the information that has been exchanged [3]. With the recent technological developments such as Web Services, a degree of interoperability at the transport and the communication layer has been achieved. However, the interoperability of exchanged documents is still a difficult problem. Although there is a lot of standardization effort trying to design e-business document interfaces, which are re-usable for different collaborative processes, still short-comings exists on providing generic business interfaces, which are adaptable to different collaborations and have common semantic basis. Furthermore, even the businesses that use a standard need to tailor it for their specific needs due to the fact that the messages need to include information specific to the industry domain that they operate and their geopolitical as well as regulatory contexts. Therefore, different customizations of even the same standard may have some interoperability problems.

The earlier standards have focused on static document definitions, which were inflexible for adapting different requirements that arise according to a given context which could be a vertical industry, a country or a specific business process. The leading effort for creating a standard

semantic basis for business documents to solve the electronic business document interoperability problem came from the UN/CEFACT (United Nations Centre for Trade Facilitation and Electronic Business) Core Components Technical Specification (CCTS) [4] which provides a document modeling methodology. The ultimate aim is to derive all electronic documents from common building blocks with well-defined rules. This implies creating core document building blocks with common semantics and then specializing them to the contexts. Additionally, both the core components and the specialized document building blocks must be discoverable from a common repository to be able to reuse them. In other words, the well-defined semantics for the core document components together with a discovery facility will improve their re-usability and enhance interoperability. In addition to a conceptual model of document artifacts, UN/CEFACT CCTS also provides guidelines on Working Process and Methodology for document artifact modeling in order to allow creation of standard-based new document schema. The UN/CEFACT CCTS is syntax independent. One of its first syntax dependent implementations in XML is Universal Business Language (UBL) [5]. Currently, the approved version of UBL is 2.0 and there are thirty-one XML schemas for common business documents like ”order” and ”invoice”. In addition to the document definitions, UBL 2.0 provides a library of XML schema definitions (XSDs) for reusable common data components like ”Address”, ”Item”, and ”Payment” from which the context specific documents are customized.

In addition to this, UBL provides a solution to the interoperability problem arising due to ”all-in-one approach” schema design with UBL Customization Guidelines [6]. Most of the standardization organizations try to design e-Business Document Schema for a broad range of requirements in order to have high adoption rate and address the needs of horizontal industries [7]. Unfortunately, ”all-in-one approach” design may result in un-interoperable systems using the same standard, since different customizations may be applied by the organizations. Therefore, the e-business document schema shall be flexible to allow customizations, but at the same time they shall still preserve their meaning after some modifications and customizations. In response to this requirement UBL presented the customization guidelines for the document artifacts in order to aid users in developing custom solutions based on UBL. UBL Customization Guidelines [6] provide very valuable guidelines on how to sustain data semantic that is required providing data interoperability while having a customized document schema. Currently the main challenge in using these UN/CEFACT CCTS based approaches

like UBL is the fact that the document artifacts are stored in spreadsheets and this makes it very difficult, if not impossible, to discover the previously defined components to reuse and to check their consistency. Furthermore, as already mentioned, businesses need to customize standard documents according to their specific needs. The UBL provides detailed text-based descriptions of customization mechanisms. Creating, extending, customizing document schema conforming to UN/CEFACT CCTS methodology are tedious, labor intensive and time-consuming processes. For example, for creating a new UBL schema requires:

1. Analysis of available component interfaces
2. Design of spreadsheet model of the document
3. Creation of XSD files
4. Creation of genericode files for each of the coded attributes.

UBL is selected as a common denominator in the modeling environment since UBL is considered as the Lingua Franca for common business information and UBL 2.0 is recognized as appropriate first-generation XML documents for eBusiness. Furthermore, UBL has been successful in real-world implementations in worldwide such as OIOUBL [8], Svefakturan [9], CODICE [10], UBL-NES [11] and the Electronic Freight Management (EFM) [12] project of the U.S. Department of Transportation. Although UN/CEFACT CCTS and UBL provide guidelines for document modeling and document customization respectively, there is no machine processable process implemented to help the designers. However without an automated tool support, it is difficult to apply the customization and maintain the consistency of the customizations. In this thesis, we address these problems by providing a document modeling environment with a common on-line UN/CEFACT CCTS based component repository which integrates the machine processable version of the paper-based UBL customization guidelines and UN/CEFACT CCTS modeling methodology. In this way, it aims to maximize the reuse of available document building blocks and minimize the duplicative efforts of document designers while customizing the document schema. The tool also generates the spreadsheet model of the document schema and the XSD files along with the genericode files.

The repository, available at [13], currently contains all of the Business Information Entities (BIEs) in the common library of UBL 2.0; all the BIEs of the UN/CEFACT Cross Industry Electronic Invoice (CII) [14]; l the BIEs of NES [11] and UBLTR [15] .The repository is

gradually evolving as new document models are created or document building blocks are customized and committed to the repository. Furthermore, its impact on the data interoperability will increase since it enables sharing all available document building blocks with a wider audience.

This thesis has been realized as a part of the iSURF project (An Interoperability Service Utility for Collaborative Supply Chain Planning across Multiple Domains Supported by RFID Devices) [17] supported by European Commission Information Communication Technologies (ICT) Seventh Framework Programme (FP7). The iSURF project aims to develop a collaborative supply chain planning environment based on CPFR guidelines addressing the interoperability challenges of deploying a CPFR process within a supply chain consortium. The motivation behind the project is the ”network is the business” vision and the project, mainly addresses the needs of SMEs. To enable SMEs to be more agile and competitive in today’s competitive world, the project envisions that the knowledge is the main driver of the competitiveness. Therefore, it proposes a system framework in which the collaborating partners share information on the supply chain visibility, individual sales and order forecast of companies, the current status of the products in the manufacturing and distribution process, and the exceptional events that may affect the forecasts in a secure and controlled way.

The iSURF general architecture is presented in Figure 1.1. The thesis research, which is highlighted in the Figure 1.1 with a rounded box, is a part of a larger effort, namely, iSURF Interoperability Service Utility. The main objective of iSURF Interoperability Service Utility is to provide electronic business document interoperability, which enable the exchange of the planning data between enterprises and especially across the domains. For this purpose, two parallel challenges are being addressed: One is to semantically annotate the document schema as described in [18] and the other is to provide a graphical environment for the customization and re-use of UN/CEFACT CCTS based document schema accompanied with on-line document repository, as described in this thesis.

The Interoperability Service Utility can facilitate the semantic mediation of Electronic Business Documents conforming to different standards such as UBL, GS1 and OAGIS. The methodology is based on UN/CEFACT Core Component Technical Specification (CCTS). It aims to provide standard semantic representations of electronic document artifacts based on CCTS and hence to facilitate the development of tools to support semantic interoperability. The ba-

sic idea is to explicate the semantic information that is already given both in the CCTS and the CCTS based document standards in a standard way to make this information available for automated document interoperability tool support. The repository handles UBL documents of planning related messages such as ”forecast”, ”product activity”, ”exception”, ”replenishment proposal”. The iSURF Interoperability Service Utility uses this semantics in the UBL customizations to automatically mediate the message instances represented in different UN/CEFACT CCTS based document standards by reasoning over the ontologies.

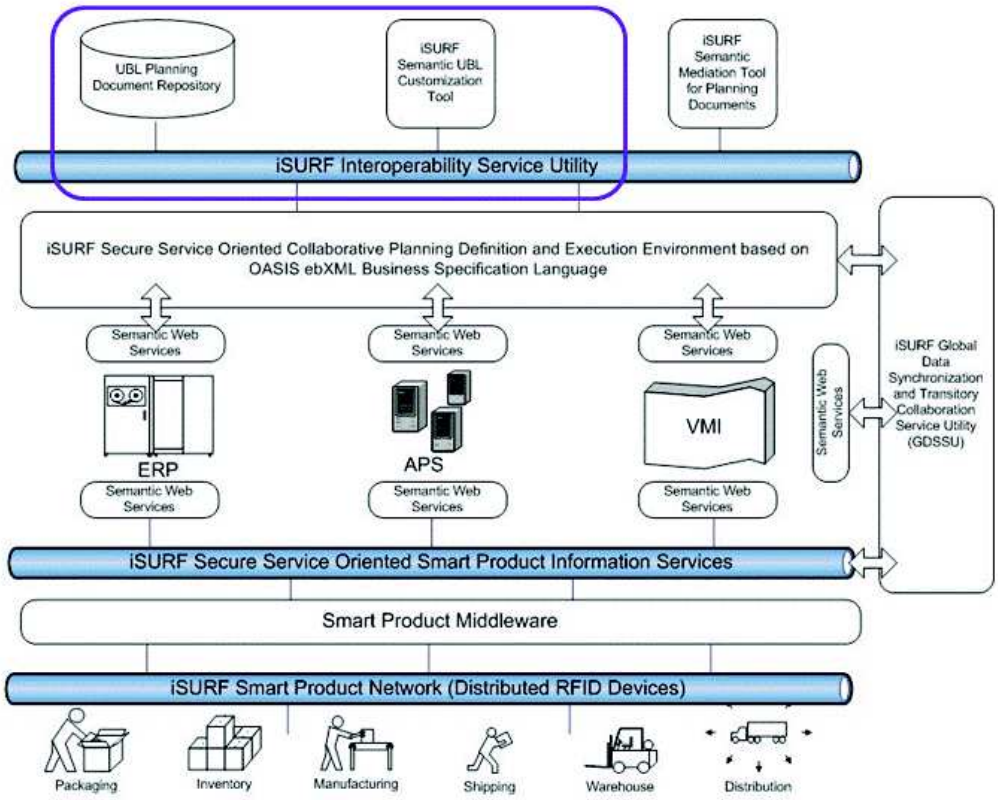

Figure 1.1: iSURF Architecture

iSURF Service Oriented Supply Chain Planning Process Definition and Execution Environment enables the definition and execution of inter-enterprise collaboration. It is the main controller of the iSURF Framework which organize interactions among the components. The interaction with legacy planning applications is achieved through the semantically enriched Web services, called as legacy adapters. The supply chain visibility data is ensured through a smart product architecture implemented based on the EPCGlobal [19] guidelines, and master data synchronization is achieved through iSURF Global Data Synchronization and Transitory

Collaboration Service Utility.

This thesis is organized as follows: Chapter 2 summarizes the background on the enabling technologies and standards. In Chapter 3, the design and implementation of the environment are presented. The use cases of tool in two different business processes and Gap Analysis Reporting Tool are described in Chapter 4. In Chapter 5, the related work is presented on UBL and on document modeling. Finally, Chapter 6 concludes this thesis and presents the future work.

CHAPTER 2

BACKGROUND ON ENABLING TECHNOLOGIES AND STANDARDS

2.1 The UN/CEFACT (United Nations Centre for Trade Facilitation and Electronic Business) ebXML Core Components Technical Specification (CCTS)

The United Nations/Centre for Trade Facilitation and Electronic Business (UN/CEFACT) is a chartered activity of the UN Economic Commission for Europe (UN/ECE). The UN/CEFACT mission is to support, enhance, and promote trade facilitation between developed, developing and transitional economies [20].

To achieve this mission, UN/CEFACT focuses on simplifying and harmonizing processes, procedures, and information exchanges through development of a comprehensive set of technical specifications and standard business processes [20].ISO 15000-5 CCTS developed by the UN/CEFACT and ISO Technical Committee (TC) 154 provides a methodology for semantic data modeling on a syntax independent level. It achieves a common understanding of data structures and message types in order to provide interoperability at the data level among e-Business applications [4]. It provides a methodology which enables re-use of general and commonly used data entities instead of defining a stable business message interface for business processes thus it provides data interoperability at the semantic level. The CCTS methodology has gained widespread adoption by various standard initiatives, since it provides context driven and collaborative framework for evolutionary modeling of documents through usage of reusable artifacts and thanks to syntax independency it can be transformed in different syntaxes by preserving the same semantic understanding.

CCTS achieves reuse of common building blocks and common understanding of data entities

through context and semantic. It provides a template for the data models and building of context specific data models for a specific business process. Furthermore, by setting Naming and Design Rules (NDR), which defines how to name, structure and assembly the components, it constitutes common understanding among business partners as it eliminates ambiguities arising from misinterpretation.

2.1.1 Key Concepts in UN/CEFACT CCTS

The key concepts of CCTS based on Core Components (CC) and the Business Information Entities (BIE): Core Components are building blocks with generic semantics and purpose; they are context-neutral templates so that it can later be adapted to different contexts and reused. Once the business contexts are identified on the CCs, in other words, it is contextualized, they become BIEs that reflects the requirements of a given business context. By this methodology, CCTS achieves development of components for a specific business process (i.e. BIE) with semantically and logically correct structure and content as it base on CCs.

To constitute common understanding of data entities in business messages, CCs shall accomplish to be a semantically concise template. Therefore, they have a common and generic modeling concept for objects and data, a naming convention for definition of the generic semantic meaning, and a fixed set of reusable data types. Many core components defined by UN/CEFACT are available to users from UN/CEFACT Core Component Library. There is an analogy with the UML modeling concept and the CCTS modeling concept: they represent UML object classes, and they may have associations with other CCs as UML object classes.

A Core Component contains only the information pieces necessary to describe a specific concept. There are three types of CCs: The definition of the types of core component as follows as they are defined in Core Components Technical Specification Version 3.0 [4].

- Basic Core Components: A Core Component which constitutes a singular business characteristic of a specific Aggregate Core Component that represents an Object Class. It has a unique Business Semantic definition. A Basic Core Component represents a Basic Core Component Property and is therefore, of a Data Type, which defines its set of values. Basic Core Components function as the Properties of Aggregate Core Components.

- Association Core Components: A Core Component which constitutes a complex business characteristic of a specific Aggregate Core Component that represents an Object Class. It has a unique Business Semantic definition. An Association Core Component represents an Association Core Component Property and is associated to an Aggregate Core Component, which describes its structure.

- Aggregate Core Component: A collection of related pieces of business information that together convey a distinct business meaning, independent of any specific Business Context.

In CCTS, as seen in Figure 2.1, a compound document artifact (termed as ACC) is composed of either atomic document artifacts (termed as BCC) or by defining associations (termed as ASCC) to another compound document artifact. Core Component is high-level semantic representation of data, and it does not include any details on data type except declaring its type. This detailed information is captured via Core Data Type (CDTs) defined by the CCTS, on which Basic Core Components shall be based. Core Components Data Types represent the smallest piece of information in a business data model, but they have no business meaning themselves. They define the nature of the content of the BCC, and provide supplementary components that give essential extra definition to the content. For example the Content Component carries the value of 12. This value has no semantic meaning on its own. However, 12 Euro, where Euro is the Supplementary Component that gives essential extra definition to the Content Component, does have meaning [4].

The CCs cannot occur in the business messages or data models as they are conceptual in nature. When a Core Component is restricted to be used in a specific business context, it becomes a Business Information Entity (BIE) and is given its own unique name. After they are contextualized, they can appear in the data models and business messages as BIEs. Eight applicable context are defined in CCTS, namely Business process, Product classification, Geopolitical region, Industry Context, Official constraint, Business process role, Supporting role, and System capabilities. For example, If ”address” is defined as a generic ACC, an ABIE with the geopolitical region set to ”U.K.” might be a ”U.K. address”. Similarly, when an Association Core Component is used in a context, it becomes Association Business Information Entity (ASBIE) and Basic Core Component becomes Basic Business Information Entity (BBIE). While contextualizing the CCs, in order to preserve naming and structuring CCTS

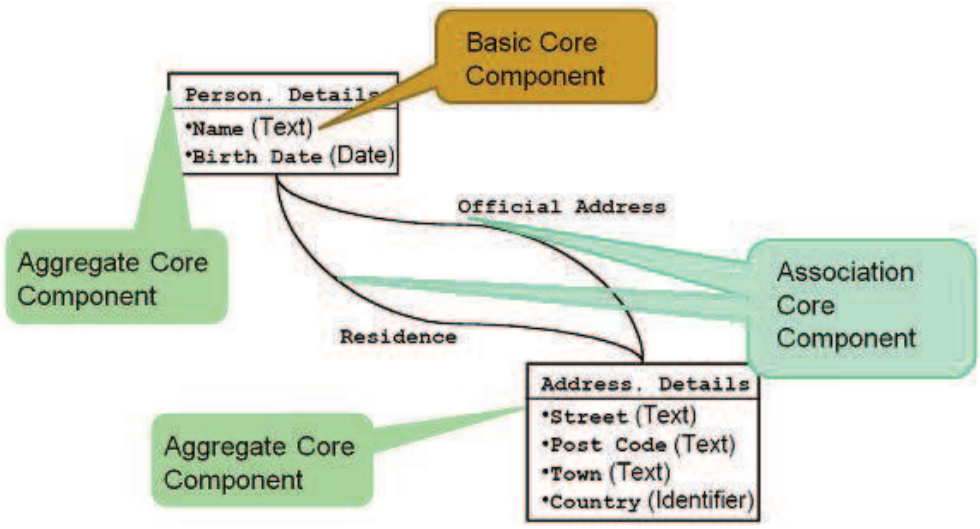

Figure 2.1: An example Aggregate Core Component Structure

derivation by restriction methodology is used, which provides representation of business requirements either via the restriction of content model or the restriction of business semantic: The content model is restricted through omitting some of the properties of the CCs while transforming it into BIEs and business semantic is restricted through application of qualifiers. Furthermore, Business Date Types can be restricted when a BBIE based on a Business Data Type on which its conceptual content model is based is contextualized; this can be accomplished by creating a Qualified Business Data Type (QBDT). Business Data Types are created for each CDTs. Each BBIE Property has a Business Data Type (BDT) that describes its value domain and they are derived from CDT of BCC.

Other concepts of CCTS, which will help the user to understand the terms used in the tool, are defined as follows in the CCTS [4].

- Artifact: A piece of information that is produced, modified, or used by a process. CCTS artifacts include all registry classes.
- Business Terms: Business Terms is the list the synonyms of the dictionary entry name under which the artifact is commonly known and used in business. A CCTS artifact may have several business terms or synonyms.
- Cardinality: An indication of the minimum and maximum occurrences for a character-

istic: not applicable (0..0), optional (0..1), optional repetitive (0..*) mandatory (1..1), mandatory repetitive (1..*), fixed (n..n) where n is a non-zero positive integer.

- Classification Scheme: An officially supported scheme to describe a given context category.

- Context Category: A group of one or more related values used to express a characteristic of a business circumstance.

- Definition: It is description of the object in English. It is recommended that the CC definition be developed first and the Dictionary Entry Name extracted from it

- Dictionary Entry Name: This is the official name of a CCTS-conformant artifact. It has some specified format for every artifact.

- Object Class Term: It represents the logical data grouping or aggregation (in a logical data model) to which a property belongs.

- Property Term: A semantically meaningful name for the characteristic of the Object Class that is represented by the core component property.

- Representation Term: The type of valid values for a Basic Core Component or Basic Business Information Entity such as Text, Code etc.

- Qualifier Term: A word or group of words that help define and differentiate an item (e.g. a business information entity or a business data type) from its associated items (e.g. from a core component, a core data type, another business information entity or another business data type).

- Usage Rules: Usage rules describe a constraint that describes specific conditions that are applicable to a component in the model.

- Version: An indication of the evolution over time of an instance of a core component, data type, business context, or business information entity.

All of these components are termed as ”document building block” within this paper.

2.1.2 Discovery and Document Design by UN/CEFACT CCTS

In UN/CEFACT CCTS document design process is initiated with discovery of the largest component which is standard business document and consists of a series of events that go deeper levels. The analysis of Business Processes, Product Context give clues regarding Context Values of sought components and reveals the requirements of business document schema.

In Figure 2.2 the steps of business document are shown. As shown in the flow chart at every level the sought component is searched in the repository after key requirements are identified, and then if it is available, it is added to the message model. If the requirements are met with the found component, then the process is finalized. Otherwise the user continues to search the repository with other required components. If the sought component is not found in the repository, the process continues with the finer-grained component discovery and creation. At the final step, finer-grained components are assembled to reach a new model and submitted to the repository.

2.2 Universal Business Language (UBL)

The Universal Business Language [5] initiative from Organization for the Advancement of Structured Information Standards (OASIS) adopts the UN/CEFACT Core Component Technical Specification approach and develops a set of standard XML business document definitions.

UBL Standard Business Schemas are now version 2.0, and they are working on v 2.1 to enlarge their document content. UBL 2.0 provides a library for a number of standard document schema, which are common to all business processes such as ”Request for Quotation”, ”Order”, and ”Invoice”. In addition to the document definitions, UBL 2.0 also provides a common library of elements, which are the basic elements of the document schemas such as Name of Person or Postal Code of Address. Furthermore, since UBL is based on UN/CEFACT CCTS mythology, and it reuses Core Component Library of UN/CEFACT.

Figure 2.3 shows the structure of the UBL Documents. It should be noted that in addition to identifying conceptual Business Information Entities (BIEs), UBL uses the CCTS artifacts

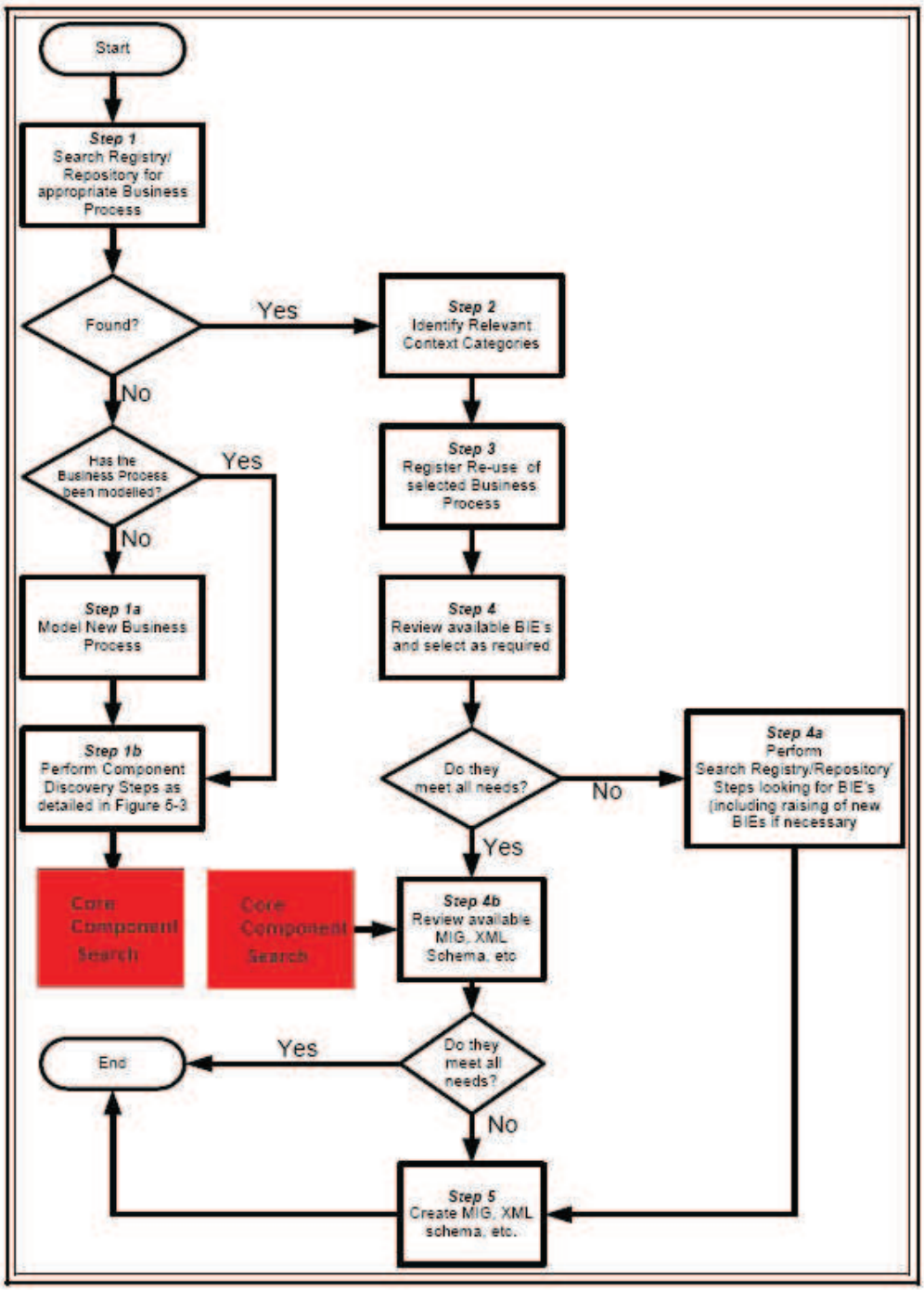

Figure 2.2: Discovery from Business Process to Core Component [4]

such as ABIE, ASBIE and BBIE to compose its document schemas. This is in contrast to some other standards, which use CCTS components in different document artifacts of their own and also name them differently. In UBL, there are two types of ABIEs: (1) The document

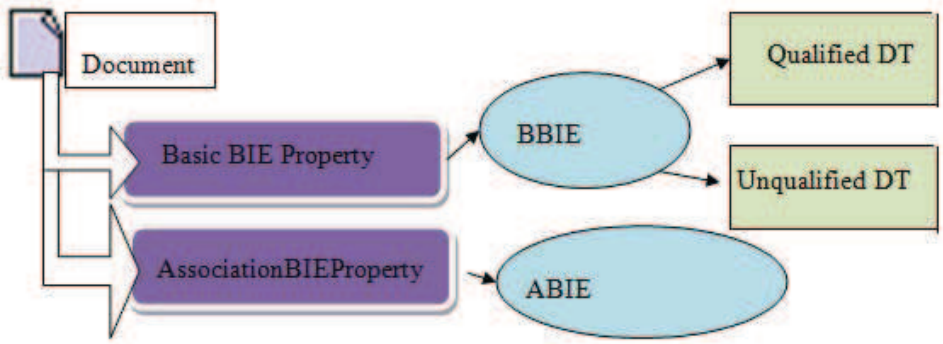

Figure 2.3: UBL Document Structure

ABIEs which represent UBL Documents such as ”Order” and ”Invoice” and (2) More fine-grained reusable ABIEs such as ”Address” and ”Party”. In UN/CEFACT CCTS an ABIE is composed of BBIEs and ASBIEs. In UBL 2.0, according to the UBL 2.0 Naming and Design Rules, this composition is realized through BIE Properties.

There are two types of BIE Properties: (1) The Basic BIE Property, which is used for relating the ABIE with a BBIE, represents an intrinsic property of an ABIE. However, in UN/CEFACT CCTS Methodology BBIEs are specialized from Basic Core Components: as already mentioned UBL started creating its BIEs before UN/CEFACT Core Components were available. (2) The Association BIE Property, which establishes an association from one ABIE to another ABIE, represents an extrinsic property. In other words, it is the Association BIE Properties that express the relationship between ABIEs. The Association BIE Properties correspond to the Association Business Information Entities (ASBIEs) in the UN/CEFACT CCTS. A BBIE has a single content whose type is specified either with Qualified Data Types (QDT) or Unqualified Data Types (UDT).

2.2.1 UBL Customization Guidelines

There are two types of customizations specified in UBL 2.0 intending to aid users in developing custom solutions based on UBL: Conformant customization and Compatible customization. The UBL subcommittee has announced guidelines on how to customize UBL documents in order to preserve common understanding at the data level after an UBL document is cus-

tomized [6].

UBL customization is defined as ”The description of XML instances, or XML-based applications acting on those instances, that are somehow based on or derived from the UBL Standard.” in the guidelines.

2.2.1.1 UBL Conformant Customization

The conformant customization is defined as

> There are no constraint violations when validating the instance against a UBL standard schema. A UBL conformant instance is an instance that validates against a UBL standard schema[6].

This is illustrated in Figure 2.4. To sustain these requirements that are provided in guidelines for Conformant customization is very tedious and error-prone.

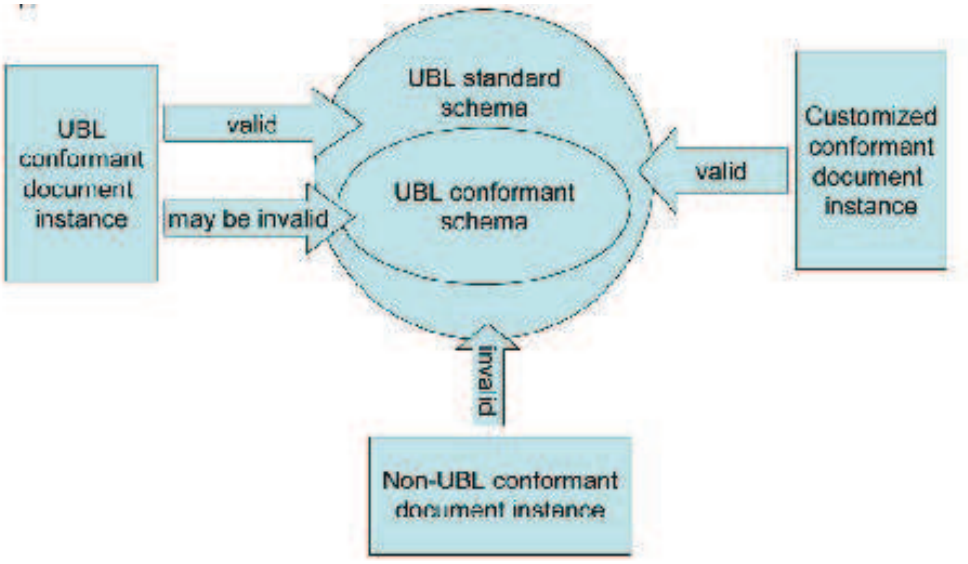

Figure 2.4: UBL Validation for Schemas and Document Instances [6]

UBL Conformant Customization Guidelines propose to model new schemas by applying restrictions. Basically there are four ways of conformant restriction:

- Subsets of a document model

- Constraints on a document content
- Using an extension area
- Using the code lists

Subsets of a document model

The UBL document schemas have designed in order to fit a broad range of horizontal industry needs, so most of the elements defined in the UBL Documents are optional and may not be needed in the most of the implementations. By sub-setting of a document model, the implementer may not have to account for caring all of the elements defined in the original document schema. In conformant design guidelines only optional elements may be excluded from the document schema in order to preserve validity against the original schema. Furthermore, the cardinality of the elements may also change: minimums can be increased to their maximum, maximums can be decreased to their minimum, and data types can be refined but not extended.

Constraints on a document content

Some additional value constraints or cardinality restrictions can be required to adapt a generic business artifact to a special profile. Since the idea of behind the UBL is not creating a new model for each requirement, schematron may be adopted to define additional constraints. Schematron provides such a dynamic validation mechanism on a document content model without restricting its schema model. For example, an organization may need to have Delivery. Details element both in Order and Invoice document. However, Expected Delivery Date is an important element of Order, whereas it is not required in the Invoice Element. For addressing such issues, most standardization organization defines schematron rules special to profiles while keeping the core schema as defined in the standard core model in order to be conformant to standard and have one base model, which does not change according to different profiles.

Furthermore, some elements may be constrained with some predefined values or conditionally found in one schema depending on existence of a value or artifact. For example, a co-occurrence constraint may constrain that for each itemized information entity that is based on the UBL party, one or both of cac:PartyIdentification/cbc:ID and cac:PartyName/cbc:Name

must be present, but not neither.

These types of constraints are defined through Schematron [22] or XSL rules [23] and feed these rules into the second phase of validation, which follows XSD validation.

Using an extension area

Some additional artifacts may be needed in a schema, which are specific the business processes. To have still conformant schema including these additional requirements can be obtained by extension area which is an exception to the general rule that only subsets are conformant.

UBLExtension elements are used as the first child of all UBL 2.0 documents and their type are defined as ”xsd:any” in UBL 2.0 to meet the requirements of any additional data need.

Using the code lists

A code list is used in UBL document schemas in order to impose instance value constraints. For example, document designer may declare that a standard code value enumeration needs to be used in the coded value attribute to be conformant.

2.2.1.2 UBL Compatible Customization

The other type of customization is compatible customization, which meets the requirements of organization, which needs more radical changes in the core UBL standard schema and still applies the rules behind the UBL. If an organization needs extending an ABIE, creating a new ABIE or creating a new document, compatible customization approach can be used in order to handle these cases. In such cases the principles behind UN/CEFACT CCTS Discovery and Document Design is used, which is described in the Section 2.1.2. In addition to this, when performing compatible customization, the users follow the UBL Naming and Design Rules.

CHAPTER 3

eBusiness Document Design and Customization Environment Design, Implementation and Features

Chapter 2 explained the UN/CEFACT CCTS Discovery and New Item Submission guidelines and UBL Customization guidelines, which were defined as the document schema generation practices enabling the trading partners to have their own document models according to their own business requirements, geopolitical region or so on.

UBL has widespread adoption around Europe and USA, especially in electronic government applications and extends its community and coverage steadily. In order to ensure interoperability among larger communities countries have started to establish larger collaborations and are working on interoperate their regional/national implementations of UBL such as ”Offentlig Information Online UBL (OIOUBL) [8] Project, which is Denmark’s initiative for e-Government applications and Svefaktura [9] Project of Sweden National Financial Management Authority. Northern European Subset (NES) [11] and UBL and European Committee for Standardization Workshop on Business Interoperability Interfaces for Public Procurement (CEN ISSS WS BII) [14] are some of these efforts which are trying to provide interoperability at a larger extent.

However, without document design and customization tool generating UBL complaint or conformant schemas is time consuming, tedious and error-prone process. Although modeling and customization guidelines provide detailed flow charts or paper-based instructions, there is no machine processable process templates defined. Today most of the organizations working on generating UBL based document models or customizing UBL documents for national/regional electronic collaborations declare that they need a tool which aids them while following UBL guidelines.

3.1 iSURF eDoCreator Requirements and Design

The iSURF eDoCreator tool provides on-line and 7/24 accessible environment for the graphical modeling of business documents by integrating UN/CEFACT CCTS modeling and UBL Customization guidelines. It is designed to aid the document designers and lighten the workload of them by automating processes. The requirements of iSURF eDoCreator have been elicited by examining paper-based guidelines and the work of UBL. Furthermore, we have gained hands-on experience while generating UBL document schemas for Collaborative Planning Forecasting and Replenishment (CPFR) guidelines. In the lights of this knowledge, the following use-case diagram for the e-Business Document Creation and Customization Guideline is figured as shown in Figure 3.1.

3.1.1 iSURF eDoCreator System Architecture and Components

The general architecture of the iSURF Document Design and Customization Environment handling the above requirements is given in Figure 3.2.

The tool can be studied under three main parts: (i) Graphical User Interface which communicates the interaction between users and the Guideline Execution Engine, (ii) the Persistence Layer, i.e. Repository architecture, which stores document building blocks and application specific data (iii) the Guideline Execution Engine side which implements main features of the environment such as query formation, XML serialization. The add-on tools such as Gap Analysis Reporting Tool and Schematron Editor are described in Section 4.3 and Section 3.2.3.4, respectively.

As shown in the Figure 3.2, the tool basically gathers the information to initiate the design process and activates the necessary mechanisms to present requested data or functionality by interacting with the Guideline Execution Engine. Meanwhile, the engine communicates with the Persistence Layer through web service invocation in order to enable working on conceptual document building block models.

i *Graphical User Interface*

The Graphical User Interface handles users' interactions and helps to visualize UN/CEFACT

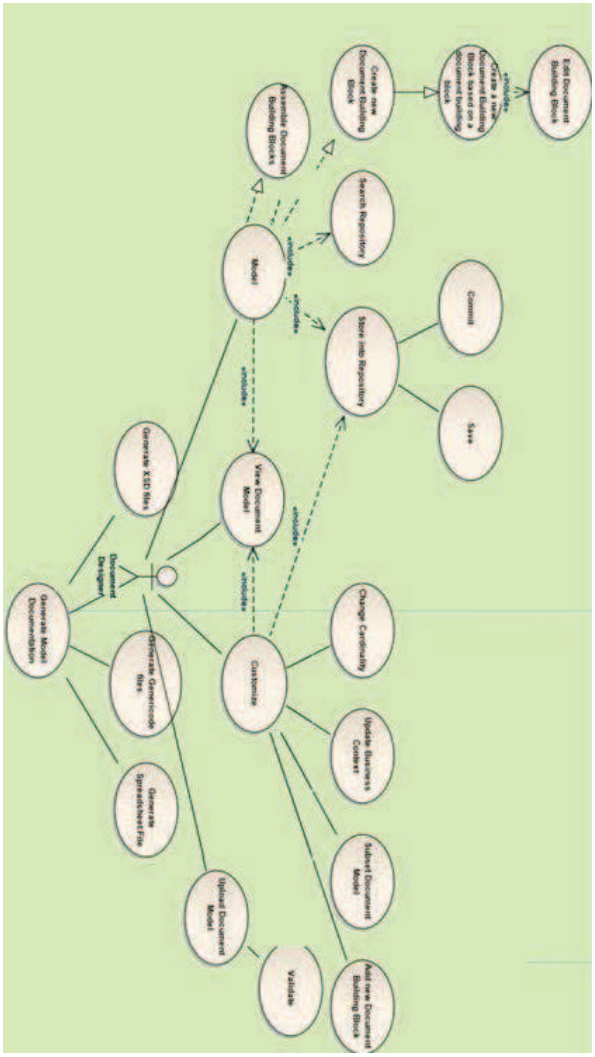

Figure 3.1: eBusiness Document Design and Customization Environment Use Case

CCTS based conceptual model in a hierarchical tree view and converts the conceptual model to graphical model at the presentation layer. The hierarchical organization of document components is shown through expandable tree interface. Furthermore, a graphical interface is provided for each feature of the document design environment. The user interfaces of the modeling environment are implemented with Flex [19]. Since, first of all, Flex provides Rich Internet Application API providing flexible and ease-to-use components, the applications developed by Flex becomes user-friendly and highly-interactive web applications. Furthermore, since it is web based, it enables the tool to be hosted on the web servers, and this will enable us to make the services publicly available and lets the users to collaboratively work. Finally, it is platform-

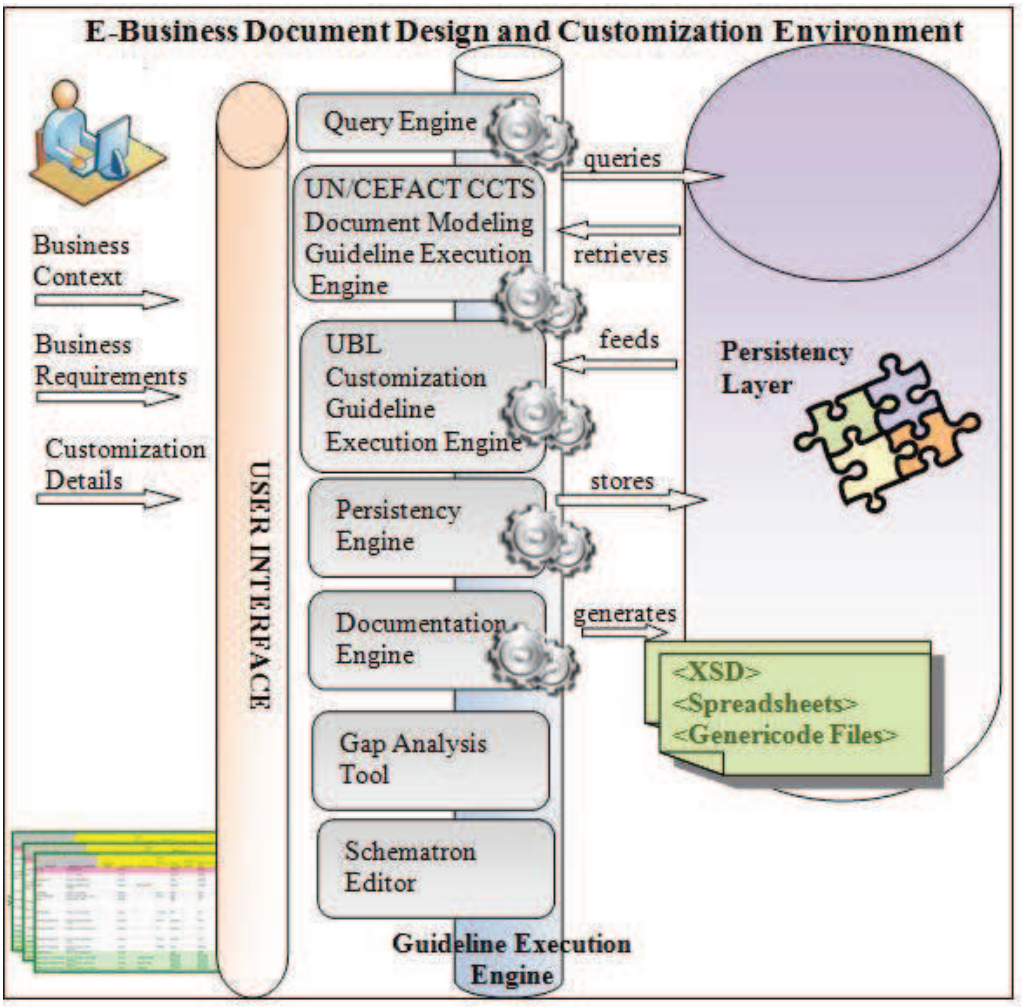

Figure 3.2: iSURF eDoCreator System Architecture Components

independent, which means it can be launched independent of the underlying operating system.

The premise behind the Flex solution is to establish a presentation layer independent from the server layer to provide a robust, feature-rich, and portable client-side execution environment. Therefore, in the architecture, the client business logic is embedded into client side user interfaces in contrast to most of the web application development platforms. The communication between client side and the server are realized through web services by SOAP messages.

The user interface, i.e. the client side, stores the document building blocks and documents in memory as data structures independent of the registry and repository object

model; it provides some functionalities such as exporting document schema without interacting with the persistence layer if the model is available in its side. In other words, it can handle some interactions with the user without requesting information from a server at every event call.

ii *Online Repository*

The on-line repository component provides persistence and graphical access to the components over the Internet through its Web-based environment. The models generated by the users are persisted by using a relational database and retrieved through Structured Query Language (SQL) which is abstracted by graphical queries. Furthermore, the wiki-based collaborative nature of the repository enables users to participate in collaborations by sharing the work. The repository makes the document building block accessible when the user commits it. Currently, iSURF eDoCreator repository [5] contains all of the BIEs in the common library of UBL 2.0; all the BIEs of the UN/CEFACT Cross Industry Electronic Invoice (CII) [6]; all the BIEs of NES [7] and UBLTR [12]. And it gradually evolves as the new document models are created or document building blocks are customized and committed to the repository.

iii *Guideline Execution Engine*

The Guideline Execution Engine is the main controller of the modeling environment. It guides the users in following the UN/CEFACT modeling and UBL customization guidelines during the generation of standard-based but at the same time customized business documents. The engine checks the consistency between the users' actions and the guidelines at all steps. Furthermore, it handles interactions between the graphical user interface and the repository. It implements a number of key features of the environment. The guideline processing engine is implemented in Java and its integration with the User Interfaces is realized through web services. The Guideline Processing Engine is the server side and provides functionalities requested by the user interface. The user interface invokes a feature of the guideline execution engine when a user submits his/her requests. The communication with persistence layer is handled through JDBC Database API. The services provided by the guideline execution engine to communicate with the persistence layer can be summarized as follows:

- *executeGenericQuery operation*: This service returns XML serializations of the

available document building blocks in the repository. The return list elements include the keywords specified in the query, in other words, the document building blocks meeting the specified qualifications in its Dictionary Entry Name, its Object Class Term, Property Term, and Representation Term.

- *saveToRegistryRepository operation*: This service takes the intermediate XML format for the document building block that will be saved and persisted into the repository for following retrievals and queries.
- *setStatusCommitted operation*: This service changes the status of document building blocks and its related components to ”Committed” from ”In preparation”. Then the document building block becomes visible to the users that are subscribed to the document building blocks’ group.
- *getClassificationSchemes operation*: UN/CEFACT CCTS specifies some standard coded values for classification schemes. For example, it states that Business Process Context values may be taken from UN/CEFACT Catalogue of Common Business Processes. In the persistence layer the recommendations of UN/CEFACT CCTS are also stored to be retrieved at the time of a request. This service retrieves these stored coded values from persistence layer and present enumerations.
- *getDictionaryEntryNamesByTypeAndUser operation*: This service takes a username and the type of the requested document building blocks as input and returns all specified type document building blocks that are visible to the user as response. The type can be ”Unqualified Data Type”, ”Qualified Data Type”, ”Basic Business Information Entity”, ”Aggregate Business Information Entity”, and ”Message Assembly”.
- *getByUniqueID operation*: This service returns the document building block content with its properties, which have the specified identifier.
- *deleteByUniqueID operation*: The service deletes the identified document building block.
- *saveUBLXSD operation*: This service generates the documentation files for the specified Message Assembly and returns them in a zip format.
- *validateSpreadsheetFile operation*: The environment accepts the new document building block models uploading from the spreadsheet file. This service validates

the structure of the spreadsheet file by checking it against the UBL spreadsheet design rules.

There are also a number of other miscellaneous services implemented for adjusting grouping options of both users and document building blocks and storing unique identifiers, or checking login credentials and so on.

The components of the environment and their basic functionalities are as follows:

- Query Engine: It gathers a number of criteria for sought document building block from the user interface and converts it to a query execution language, i.e. SQL, and executes it over persistence layer.
- UN/CEFACT CCTS Document Modeling Engine: In the engine, there are six steps starting with document model discovery and finishing with documentation generation, which are modeled according to UN/CEFACT CCTS methodology described in Section 2.1. However, the system does not include Core Component Levels since UBL only uses Common Core Components defined by UN/CEFACT CCTS Library and UN/CEFACT is responsible for introducing new Core Components.

  The six steps of the Document Design and Customization Tool can be summarized as follows:

  - Step 1: Searching for a Document Schema
  - Step 2: Selecting a Business Document
  - Step 3: Searching for an Aggregate Business Information Entity
  - Step 4: Selecting Aggregate Business Information Entities
  - Step 5: Aggregating available Business Information Entities
  - Step 6: Generating Schema Documents

  Although the environment provides UN/CEFACT CCTS methodology steps, it does not mandate a strict sequence in applying these steps; rather it lets the users freely create the document models and guides them by notifying and reminding the steps that need to be taken: Some steps may be repeated or some steps may be skipped. The user can organize the flow according to his needs, and may finalize the generation or

customization of Business Document Schema, whenever he thinks that the generated schema is satisfying.

- UBL Customization Guideline Execution Engine: Similar to UN/CEFACT CCTS Document Modeling Guideline Engine, this engine implements the paper-based UBL Customization guidelines in a computer processable way. It enables the user to exclude some building blocks from a core document model, or extend the document model. It presents a number of customization options to constrain the document model through graphical user interfaces.

- Persistence Engine: It mediates the interaction between the user interface and the persistence layer. It serializes document building block models into an intermediate XML format in order to enable visualization of document building blocks in graphical user interface and deserializes customized document building block models into the intermediate XML format and then to the object model in order to persist it into the repository.

- Documentation Engine: UN/CEFACT CCTS provides conceptual modeling and does not mandate the technical implementation details. UBL adopts UN/CEFACT CCTS conceptual models and provides XML representation of the conceptual models. In this tool, three layers are provided for representation of the document models as shown in the Figure 3.3.

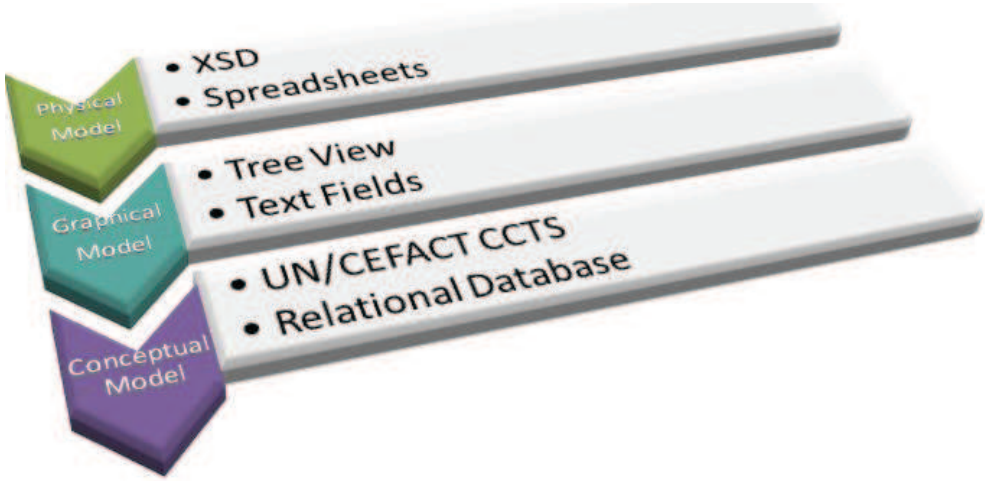

Figure 3.3: Representation of Document Models

For the exchange of business documents among organizations, the designed conceptual

model of the document is converted to a physical model which will provide technical interoperability. In order to preserve common semantic, which was achieved in a conceptual model via graphical modeling, the modeled artifacts should be represented in a common and unambiguous way. Documentation Engine handles the upper layer. It produces physical models for the graphically generated document models. For this purpose, it generates both spreadsheet files and XML schema files accompanied with genericode files, which stores possible enumeration values for coded types such as ”EN”, ”US”, ”TR” for coded type Country.

For sustaining common and unambiguous representation of conceptual models, UBL Naming and Design Rules (NDR) which allow for a unique representation of document artifacts are utilized. After finalization of modeling of document schema, the physical representation of the produced conceptual model in XML Schema (XSD) is automatically generated by following the Naming and Design Rules of UBL. In addition to XSD of the document model, the XML schema generator automatically detects dependencies in the Business Information Entities and generates additional XML schema files for contained data type definitions, Business Information Entities etc. In this physical model, customizations are represented as XSD derivation operations.

The XSD schema of a conceptual document model is accompanied with some informative supporting materials such as spreadsheet models defining the document artifacts and genericode files. UBL prefers to present all details of the document artifacts in spreadsheet formats, which are versatile and manageable for maintaining the concept details. The tool provides the same structure that is used for defining UBL document models and generates one folder named as ”maindoc” for the spreadsheet model of a document model and a folder named as ”common”, which includes files for the spreadsheet model of Common Library Document artifacts included in the document. The generated genericode directory, i.e. ”gl”, includes a file for controlled vocabularies used in Code.Type in a specific format defined by UBL.

Furthermore, the new models can be added to the repository of the tool by uploading spreadsheet models. The tool can handle both Microsoft Excel and Open Office Calc spreadsheet formats. It parses the spreadsheet file and loads document building blocks into the repository in order to be shared with the users.

- Gap Analysis Reporting Tool: Interoperability is the main goal of the most of the ap-

plications in order to collaborate with other applications seamlessly. The Gap Analysis Reporting Tool compares the interoperability level of two messages according to identified 4 levels of problems. The details of the tool are presented in Section 4.3.

- Schematron Editor: UBL uses XSD for expressing normative document constraints. However, XPATH expressions can be used to subset the document model given the context. In a specific implementation of UBL 2.0, there may be additional constraints on the value space of information entities. For example, ”The Total Value of an Order cannot be more than 50,000 USD”. There may also be rules about dependencies between values of the elements, such as ”The Shipping Address must be the same as the Billing Address” or ”The Start Date must be earlier than the End Date”. The former type of requirements can be reflected to the UBL schemas by type restriction; however, it requires schema modification. On the other hand, the latter type of requirements cannot be represented through XSD schemas. Therefore, users can describe these constraints through Schematron or XSL rules. Schematron Editor is a tool to define such subsetting constraints on a document model through graphical user interfaces. It generates a schematron file to validate the documents against the subset model. The user interface provides an abstraction on XPATH expression and let the user easily produce schematron files without any knowledge on XPATH or schematron. The details of the tool are presented in the Section 3.2.3.4

3.2 Features of the iSURF eDoCreator with Implementation Details

This section provides the usage and implementation details for the elicited requirements presented in Figure 3.1.

3.2.1 Providing the graphical view of the document building blocks in the repository

The first functionality of the tool is to provide a list of all available document building blocks such as Message Assemblies, Aggregate Business Information Entities, Basic Business Information, Qualified Data Types, and Unqualified Data Types in the repository by switching over tabs as shown in Figure 3.4. In the list, the document building blocks are identified according to their Dictionary Entry Name, Status, Owner, Creation Date, Customization Identifier and

Profile Identifier. Dictionary Entry Name is the official name of the artifact which is defined according to the UN/CEFACT CCTS guidelines. Status of an item can be either ”Committed” or ”In preparation”. If it is in preparation, this means that it is in the sand-box of the user, and it is not available, or in other words, visible, to all other users of iSURF eDoCreator. If the status is committed, then the component is visible to users or groups that the owner selects. The owner column specifies the username of the creator of that component.

For initialization of the screen, user interfaces invokes getDictionaryEntryNamesByTypeAndUser operation of the guideline execution web service.

The tool provides the details of properties of these document building blocks that are listed in the table when a user selects the ”properties” option of the pop-up menu that appears when a right click is realized on the selected item. The right click invokes getByUniqueID operation of the guideline execution web service and retrieves the details of the selected document building block from the persistence layer via guideline execution engine.

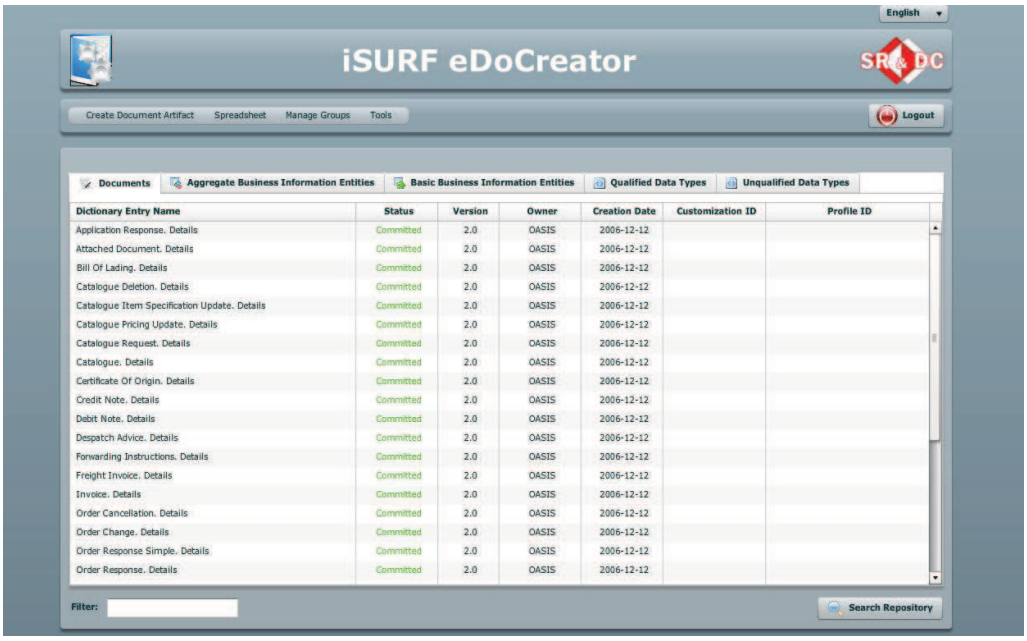

Figure 3.4: Main Screen of iSURF eDoCreator

In the eBusiness Document Design and Customization Environment, the data models are presented to the user as hierarchical graphical data models. The model presents the document schema in a Russian Doll model. The artifacts are encapsulated within each other according to

their hierarchy while being presented to the users. This encapsulation hierarchy is presented to the user via Tree Interface as in Figure 3.6. In a tree node, a document artifact is represented by its Dictionary Entry Name and the icon on the node indicates the type of document artifact. By clicking on the nodes of the tree, users can expand the tree node and -if it is not a leaf node- see the composition details of building blocks such as which Aggregate Business Information Entities, Basic Business Information Entities or Association Business Information Entities are included in Message Assembly, or what is the Data Type of the Basic Business Information Entity.

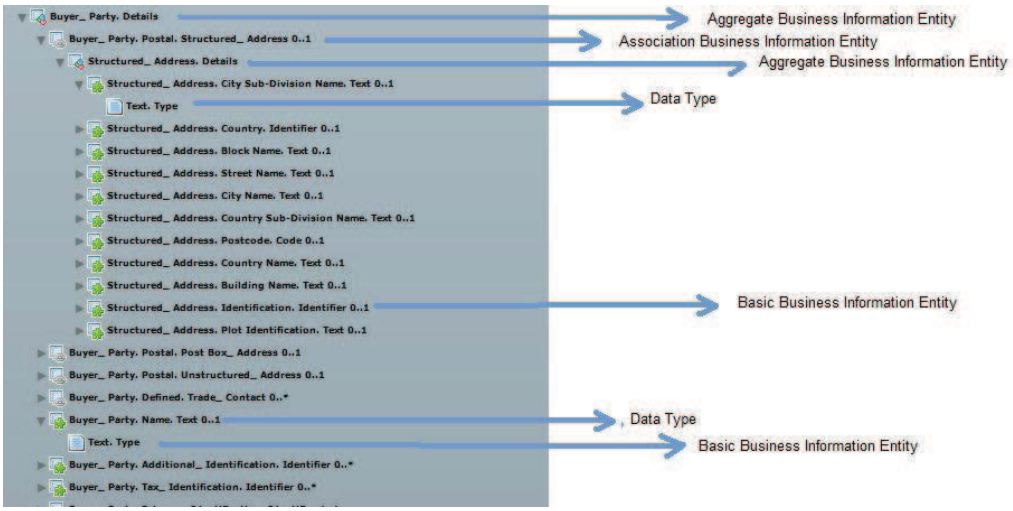

Figure 3.5: Document Building Blocks in a Tree View

The relation with the artifacts and node icon are as follows:

This expandable view of the document building blocks allows users to see the whole data content of the component at a glance. However, UBL provides the definition of UBL Document Schema in a spreadsheet format in which rows correspond to document building blocks, and columns give the details of the component on that row. Furthermore, the properties of encapsulated document building blocks are presented in separate spreadsheets, so in order to have a complete view of a document model; the user needs to explore more than one spreadsheet by going back and forward among document folders. For example, when a user examines ”Message Assembly” content, he first needs to explore ”maindoc” folder and open the spreadsheet file. Then for each ”Association Business Information Entity” and ”Basic Business Informa-

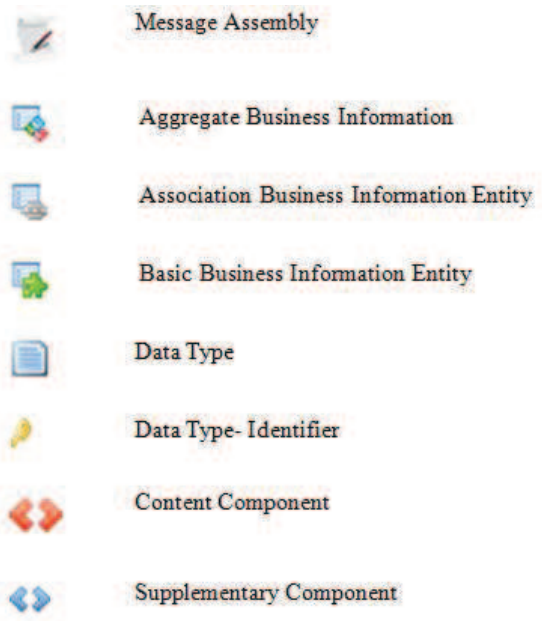

Figure 3.6: Tree Node Icons

tion Entity", he goes back to common directory and open "UBL Common Library" spreadsheet. Then for each Business Information Entity in each "Aggregate Business Information Entity", he should look for "Basic Business Information Entity" spreadsheet. After that for finding correct data type, the user shall need to open Data Type spreadsheet folder for each Basic Business Information Entity. This tool eases the visualization of the components of the documents by presenting them graphically.

Furthermore, further details of the component such as cardinality, namespace, or definition are presented to the user on the right panel when a node that represents a document building block is selected as shown in Figure 3.7.

The business context of Message Assemblies and Business Information can also be viewed by clicking on the link button "View Context" in the right panel. The context of the entity shows how the document building block is customized according to business context, product classification etc.

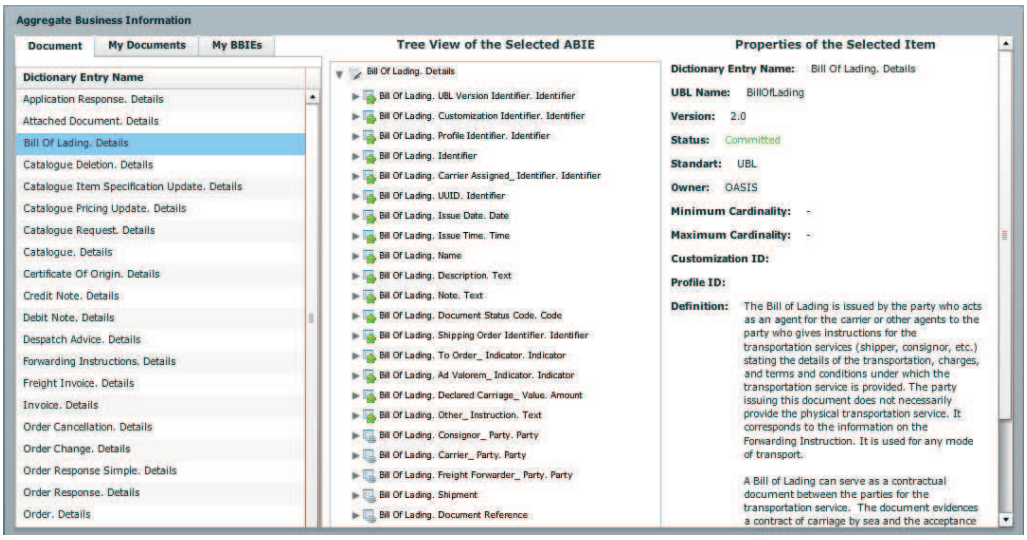

Figure 3.7: View of Additional Annotations of Document Building Block

3.2.2 Modeling a Document Schema

Basically, it encapsulates six steps as described in the introduction of UN/CEFACT CCTS Modeling Guideline Execution Engine.

3.2.2.1 Querying the Repository to Discover a Document Building Block

At start-up view, all available document building blocks, which the user has right to visualize, are shown. But due to an excessive amount of available building blocks users may have some difficulties to find out the component that (s)he is looking for. For this purpose, within the tool two different discovery features are provided namely, filter and search.

i *Filtering the list*

At the bottom of each table listing document building blocks, there is a text box which narrows downs the list of visible document building blocks according to input value. This operation tries to match Dictionary Entry Name of the components against the input keyword. For example, as shown in the Figure 3.8, when the ”ed” is entered, it eliminates most of the document building blocks from the list, and lists ”Attach**ed** Document. Details”, ”Cr**ed**it Note. Details”, and ”Self Bill**ed** Cr**ed**it Note. Details”

and so on.

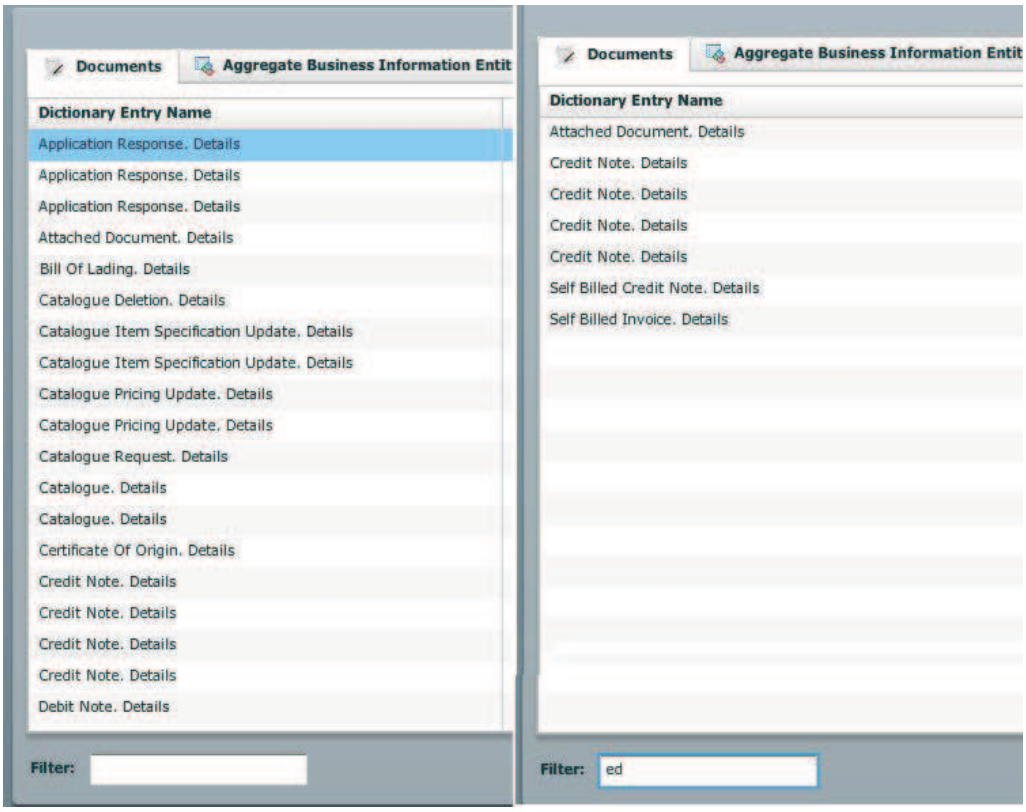

Figure 3.8: Filtering the List of Document Building Blocks

ii *Querying the repository* As declared previously one of the burdens of adopting UN/CEFACT CCTS methodology is the lack of an on-line queriable repository architecture. Users need to examine lots of different spreadsheets to find out the component that they require. As we built up an on-line repository on top of relational database, we can execute SQL queries to retrieve the document building blocks from the repository. The result list includes the standard document artifacts as well as the document building blocks that are created by iSURF eDoCreator users.

Abstraction on SQL query construction is provided for users through developed Graphical User Interfaces. The tool handles the interaction between the user and the registry/repository architecture in a user-friendly way through the implemented Rich Internet Application Interfaces. The keywords which are used in building the query constraints are gathered from users via a graphical interface. Then, these constraints can

also be connected with logical Boolean ”AND” and ”OR” connectors.

The client side sends gathered data fields, keywords and logical constraints to the Guideline Execution Engine, which forms the SQL query and executes it, by invoking executeGenericQuery operation. For example, if a user wants a document building block of which Dictionary Entry Name includes ”Order” and Definition includes ”Item”, the results presents ”Order. Details”, in the initial repository content, this result set may change later as the repository evolves.

3.2.2.2 Creating a New Document Artifact by Assembling Available Document Building Blocks

To enlarge the content of a document building block, already available document building blocks may be assembled to the target message schema.

For example, the invoice document in Turkey requires having a ”T.C Kimlik No”, the citizenship number, field which is used in Turkey to identify people. As this is a local requirement, a new identifier document artifact needs to be assembled to ”Party. Details” in order to model a larger component satisfying Turkey’s national requirement. To handle the above requirements, in the tool a graphical functionality is provided through drag and drop features.

After the editing or customization process is initiated, a tree view of a selected component is shown at the middle of the panel. This area can be thought as a sand box of the document designer: A user may have experiment some document structures until save button is pressed found in the control bar. All the changes made in this sand-box are memory-resident, after Message Assembly creation is finalized; it is submitted to registry/repository architecture and becomes visible to other users. The middle panel always shows the document schema which is created by the user; a user can customize, modify the structures presented on this panel. The structures presented in the left panel are repository items, and they are not editable, and just presented for examination purposes. The repository items are also presented in a tabbed view. In the tabbed view only components that can be assembled into the selected document building blocks are presented. For example, while modeling a Message Assembly, Aggregate Business Information Entity tabs and Basic Business Information Entity tab are presented. There are also special tabs for the items created by users with the label starting with ”My ...” tag in order to ease the users’ work. Because users generally first create the components that

will be assembled, then assemble those components into the larger document building block.

A document building block assembly process is started with the selection of an applicable document building block from the document building blocks list. The user is expected to drag and drop the document building block from the left panel to the right panel in order to assemble the selected schema. This process is visualized in 4 steps of Figure 3.9.

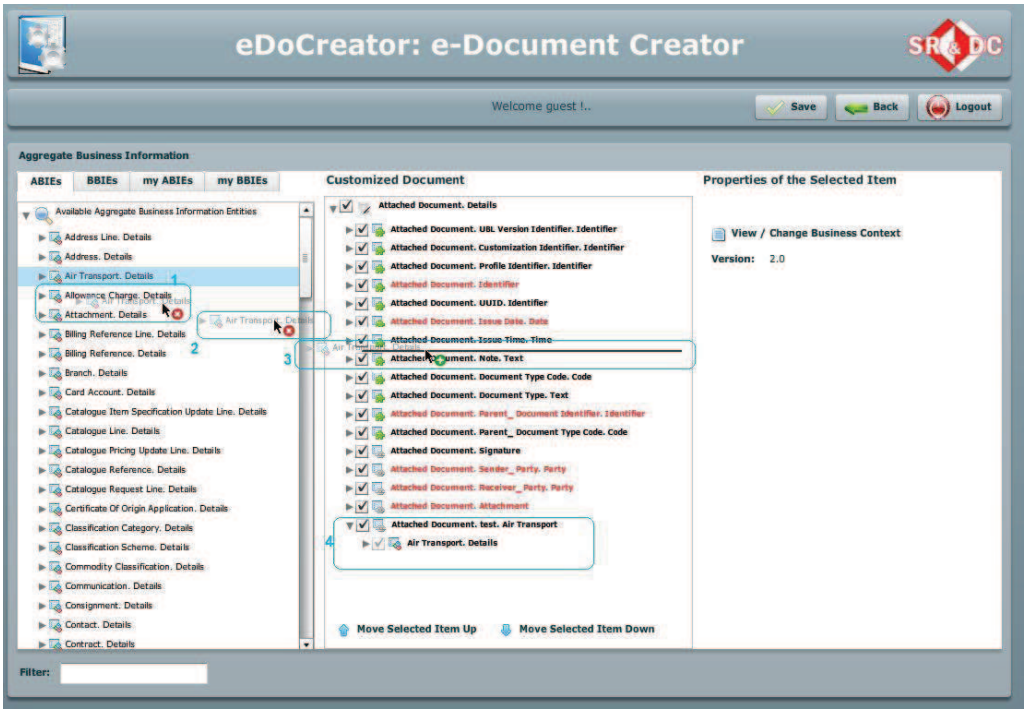

Figure 3.9: Implemented Drag & Drop Feature to Assemble Document Building Blocks

When the user drops one of the document building blocks into the target message schema, the information shown in Figure 3.10 requested in order to create a new association that will link the Message Assembly and added Business Information Entities. The requested information is the property of this new association.

When the drop is completed the related fields of the Message Assembly and included Association Business Information Entity Properties or Basic Business Information Entity Properties are updated according to input entered Property Term and Property Term Qualifier. The Object Class Term of the dragged Basic Business Information Entity is automatically updated accordingly by the tool. For example, when ”Address. Line. Text” is dropped to ”Location. Details”, then Dictionary Entry Name is updated to ”Location. Line. Text”.

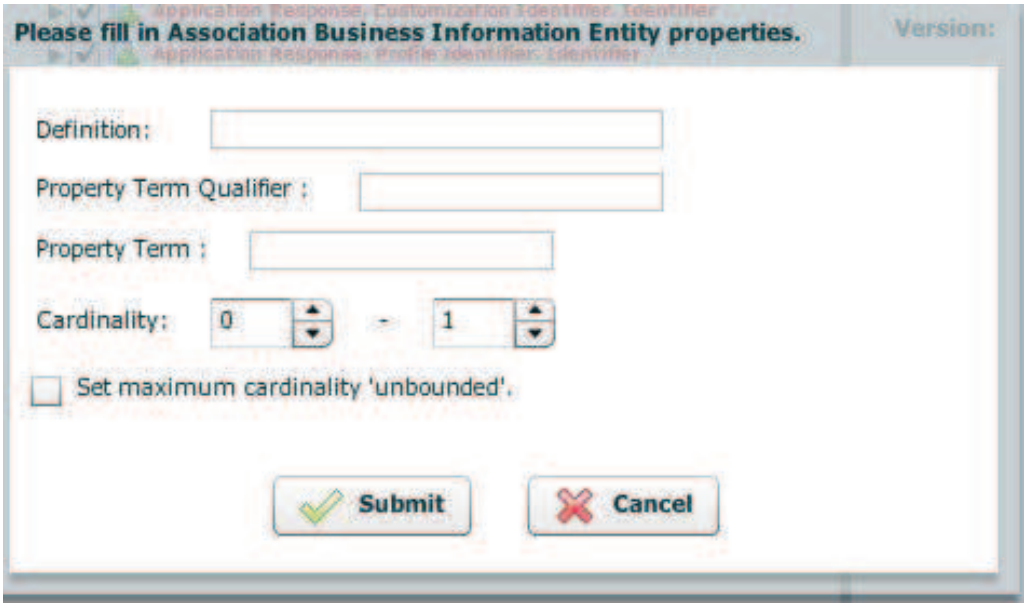

Figure 3.10: An Input Panel for requesting the Information that is required to assemble a document building blocks

Furthermore, the tool automatically locates the dragged component according to conventional ordering of UN/CEFACT CCTS: Basic Business Information Entities are listed before Associations Business Entities in the Message Assembly. They are located at the end of their list as shown in Step 4 of 3.9; however, users are enabled to change the order of the assembled document building blocks through up and down buttons after selecting the component to be moved. These buttons are at the bottom side of the middle panel of 3.9.

3.2.2.3 Creating a New Document Schema

To create a new document building block already existing finer document building blocks can be aggregated to form a larger components. For example, since there is no defined CPFR message in UBL, a new Message Assembly is required to be formed by aggregating available document building blocks such as Item. Details, Location. Details, Party. Details and so on for creating an Item Information Request message.

After the user examined all available document building blocks and was not able to find any document building block of which content is satisfying, users may create new document building block and commit it to the repository. However, users are strictly recommended to check

available ones from the repository before creating a new component. The options are provided in the main screen as shown in Figure 3.4 under ”Create Document Artifact” menu. The provided options are ”Create Document”, ”Create Aggregate Business Information Entity”, and ”Create Business Information Entity”. If users try to create a new Message Assembly, (s)he shall need to click on ”Create Message Assembly”.

First, a panel is initiated to request Object Class Term, Definition, Usage Rules, Namespaces etc. as shown in Figure 3.11. After the user enters the information, ”Start to Create” Button shall be pressed. After this button is pressed, the user can assemble building blocks from the list as described in Section 3.2.2. When the user thinks that the Message Assembly is completed according to his/her business requirements, at any time- any step- the user can finish creating the Message Assembly and save it to the registry/repository architecture. Until the user presses save button, all the changes done for creating a new document building block are memory-resident and not submitted to the repository.

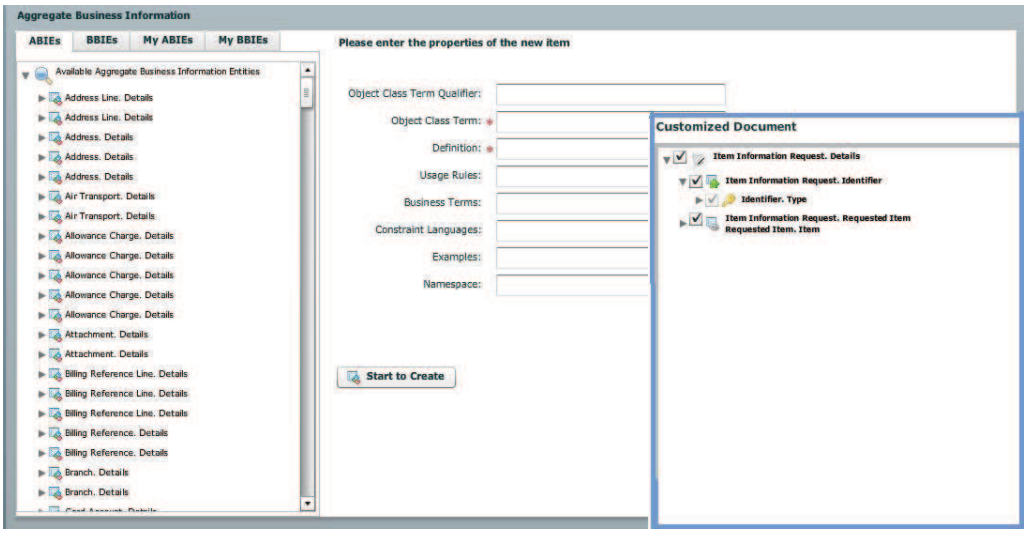

Figure 3.11: An Input Panel for requesting the Information that is required to assemble a document building blocks

The process is completely the same with the creation of an Aggregate Business Information Entity process. Creating a new Basic Business Information Entity is similar to the above process. However, this time the information shown in Figure 3.12 is requested.

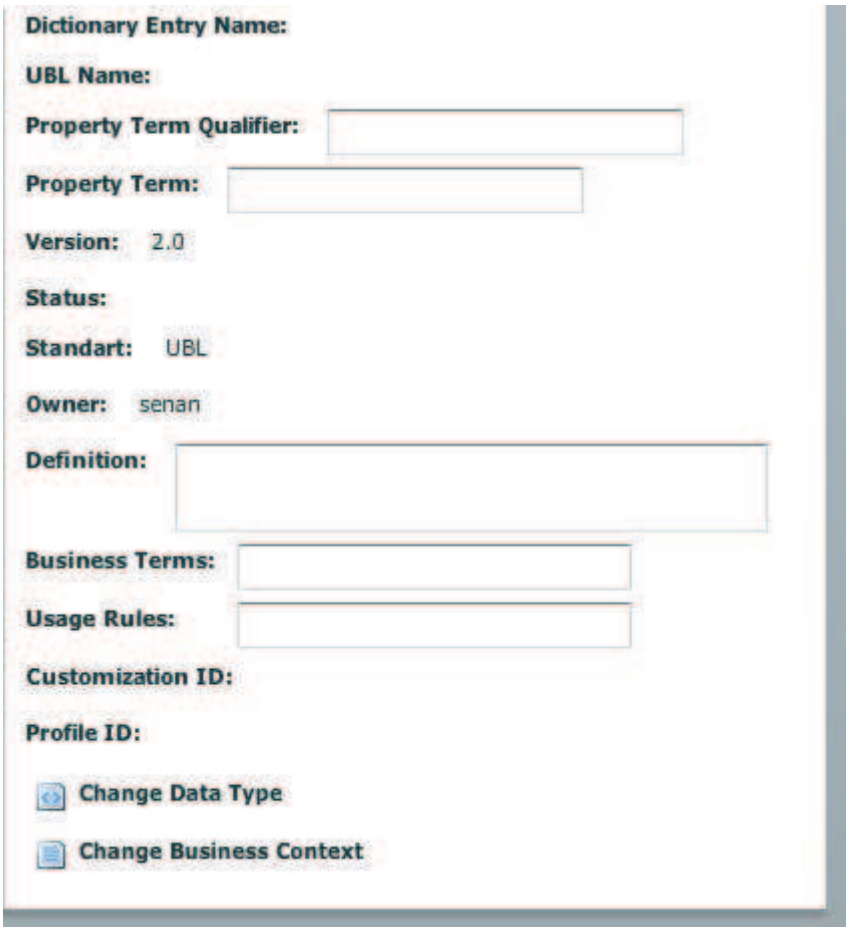

Figure 3.12: A Panel for Creating Basic Business Information Entity

3.2.2.4 Creating a New Document Schema Based on Existing Models

One way of creating new models is to reuse already existing document building block with some minor modifications. This can be realized by using customization features of the iSURF eDoCreator. This feature will be detailed in Section 3.2.3. Another possibility to create a new building block from the existing model is to refine a Data Type to generate a new Qualified Data Type. iSURF eDoCreator has support for refining Data Types. By using iSURF eDoCreator users can either qualify an Unqualified Data Type or further qualify a Qualified Data Type. In the tool, no modification is allowed on Unqualified Data Types to create a new one since UBL provides all basic data types that can be used.

For either qualifying an Unqualified Data Type or qualifying further a Qualified Data Type, a user needs to initiate the process by a right click. For qualifying an Unqualified Data Type,

a user needs to right click on an Unqualified Data Type, and select ”Qualify”, whereas for qualifying further a Qualified Data Type, a user needs to select an item from Qualified Data Type list and select ”Customize” from the context menu that is initialized. After that, the initialized panel is the same in both processes, but Qualified Data Type has some prior information or has already been customized with respect to ”Unqualified Data Type” since it is already ”qualified”.

According to the type of Data Type document building block such as Text, Code, specifiable restrictions that can be realized on Content Component varies. The environment provides some built-in restriction functions for this purpose. The Figure 3.13 shows how Content Component of Amount. Type is qualified. On the left panel, the Data Type is visualized in a check tree graphical view so that the users can subset the supplementary components of the Data Type. If the selected tree node is a Content Component, the right panel is initialized and allows users to add restrictions to selected Content Components. Available built-in restriction functions for Amount. Type and Quantity. Type are Total Digits, Fractional Digits, Maximum inclusive, minimum inclusive, Maximum exclusive, Minimum exclusive. For Code. Type, DateTime. Type, Date. Type, Identifier. Type, Indicator. Type, Name. Type, Numeric. Type, Time.Type, Value. Type the built-in functions are Expression, Minimum Length, Maximum Length, Length, Enumeration.

In addition to Content Component, Supplementary Components may also be edited in the environment via subsetting available Supplementary Components, in other words, adding or excluding some of the Supplementary Components.

3.2.2.5 Deleting Document Building Blocks

From the context menu, which is initialized when the right mouse is clicked, users may select ”Delete” option. This operation causes the deletion process for the selected document building block. Users are enabled only to delete the components that are not committed to the repository since committed document building blocks may be used by other users.

With the trigger of delete operation selection, the graphical user interface invokes ”deleteByUniqueID” operation of the Guideline Execution Web Service, which detects the type of the selected document building blocks and calls its internal functions implemented specifi-

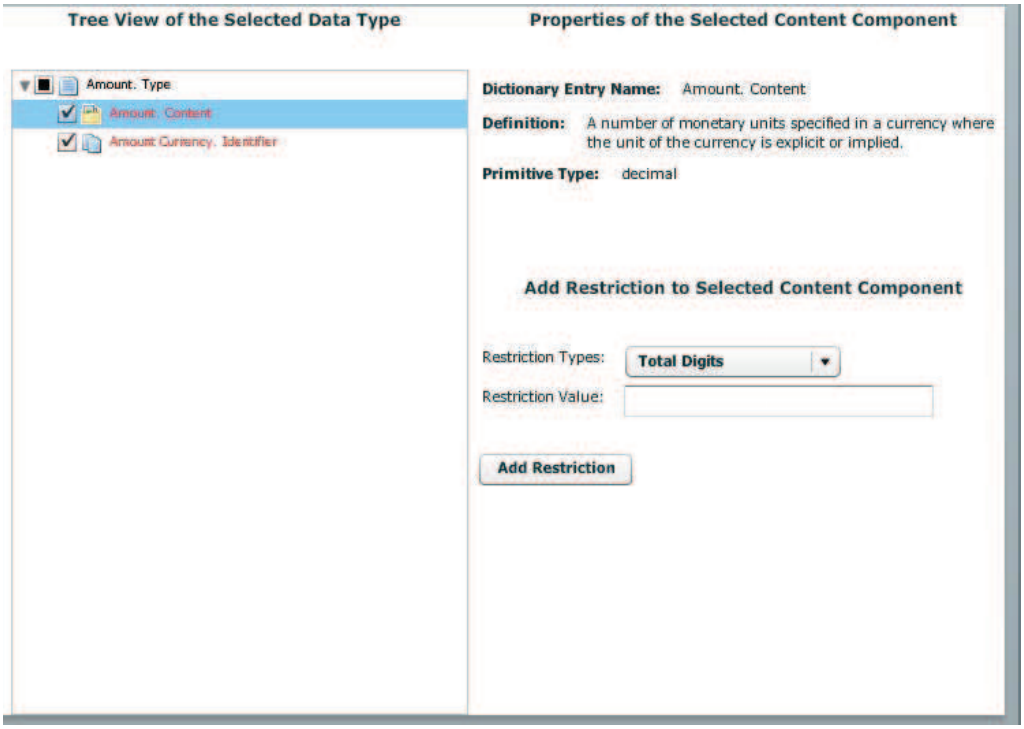

Figure 3.13: A Panel for Qualifying Amount. Type Content Component

cally for the document building block types.

3.2.2.6 Storing a Document Building Block into the Repository

The document design generated within the tool is memory resident until it is submitted to the repository. Therefore, before switching between functionalities of the tool a user has to save it and start up the other functionality to not to lose the work that was carried out. For example, while creating an Aggregate Business Information Entity if a Basic Business Information Entity is needed, which is not available for now, a user shall save the created Aggregate Business Information Entity as it is now, then he/she shall initiate the creation process of the new Basic Business Information Entity, and finally continue to modify the saved Aggregate Business Information Entity and add the newly created Basic Business Information Entity.

A new item is submitted to the Registry/Repository Architecture when user finalizes the creation or customization of an artifact. There are two different functions provided in the tool for storing document building blocks into the repository: ”Save” and ”Commit”. These are

two-leveled functions. Save function marshals the graphical data structure to the repository and makes its status ”In Preparation”, which means that the document building block is only visible to the user, i.e. the owner, and it is still being prepared. Then to make it available by other users and to finalize its creation, users need to commit it. Commit functionality is available for the document building blocks of which status are ”In preparation”. saveToRegistryRepository operation of the Guideline Execution Web Service is invoked for saving the document building block and setStatusCommitted operation of the Guideline Execution Web Service is invoked for committing the document building block.

For saving document building block the client side converts graphical model objects to XML format and submits the XML message to the server via web service invocation. At the server side the objects can be submitted to the web server with the invocation of any of three available storing operations, namely saveBMT2Registry, saveABIE2Registry, saveBBIE2Registry according to the type of submitted artifact.

The server parses the coming message and alters the content of underlying database tables. The query functionality is realized over these persisted objects.

XML Serialization of Document Building Blocks

The Core Components Technical Specification artifacts are exchanged between the Registry/Repository architecture and Flex Graphical User Interface in XML format. Therefore, some methodologies are required for conversion of the representation from graphical data model objects to XML format and XML format to database table rows in order to retrieve or store these artifacts. The XML representations of the artifacts are as follows:

The Figure 3.14 shows the XML representation of an Aggregate Business Information Entity. In the representation unique ID, entry type, dictionary entry name, definition, qualifier, object class term, business terms and context values are listed. Entry Type indicates the type of Business Information Entity: Its value shall be one of the ABIE, ASBIE, or BBIE. The contents of the Aggregate Business Information such as Basic Business Information Entities and Association Business Information Entities are listed below of the Aggregate Business Information Element in BusinessInformationEntity XML Elements as analogous to spreadsheet file representation of UBL artifacts. All of the BusinessInformationEntity XML Elements between one BusinessInformationEntity Element with ABIE EntryType and the other one with

```
<BusinessInformationEntity>
    <UniqueID>UN01002092</UniqueID>
    <EntryType>ABIE</EntryType>
    <DictionaryEntryName>Unstructured_ Address. Details</DictionaryEntryName>
    <Definition>The location, using an unstructured address format, at which a particular
            organization or person may be found or reached.</Definition>
    <ObjectClassTermQualifier>Unstructured</ObjectClassTermQualifier>
    <ObjectClassTerm>Address</ObjectClassTerm>
    <BusinessTerms/>
    <BusinessProcessContext>In All Contexts</BusinessProcessContext>
    <ProductContext>In All Contexts</ProductContext>
    <IndustryContext>In All Contexts</IndustryContext>
    <GeopoliticalContext>In All Contexts</GeopoliticalContext>
    <OfficialConstraintsContext>None</OfficialConstraintsContext>
    <RoleContext>In All Contexts</RoleContext>
    <SupportingRoleContext>In All Contexts</SupportingRoleContext>
    <SystemConstraintsContext>In All Contexts</SystemConstraintsContext>
</BusinessInformationEntity>
```

Figure 3.14: XML Representation of Business Information Entity

ABIE EntryType belong to the first BusinessInformationEntity with ABIE Entry Type; in other words, all the Entities defined under ABIE are the properties of that ABIE until a new ABIE is defined.

The Association Business Information Entity and Basic Business Information Entity XML representations are presented in Figure 3.15 and Figure 3.16 , respectively. The representation of associated Aggregate Business Information Entity of Association Business Information Entity can be found in file by looking at an Aggregate Business Information Entity of which Dictionary Entry name is equal to <Associated Object Class Term Qualifier>_ <Associated Object Class Term>. Details.

The XML representation of Data Type is shown in Figure 3.17. Similar to above artifacts, associated Complementary and Supplementary Content Components are listed below of that Data Type until a new Data Type is defined. The Data Types of Basic Business Information Entity are declared in the Data Type field of their representation.

For presenting these types at the graphical side these XML representations are parsed and converted to the Objects. UN/CEFACT CCTS methodology has defined a number of entity classes that should be used as building blocks of business document standards. In this subsystem, we modeled these entities as Entity Classes. They enable us to re-use common building blocks while creating interoperable business document content standards.

```
<BusinessInformationEntity>
    <UniqueID>UN01001394</UniqueID>
    <EntryType>ASBIE</EntryType>
    <DictionaryEntryName>Buyer_ Party. Postal. Unstructured_ Address</DictionaryEntryName>
    <Definition>The unstructured postal address for this buyer party.</Definition>
    <ObjectClassTermQualifier>Buyer</ObjectClassTermQualifier>
    <ObjectClassTerm>Party</ObjectClassTerm>
    <PropertyTermQualifier/>
    <PropertyTerm>Postal</PropertyTerm>
    <AssociatedObjectClassTermQualifier>Unstructured</AssociatedObjectClassTermQualifier>
    <AssociatedObjectClassTerm>Address</AssociatedObjectClassTerm>
    <BusinessTerms/>
    <MinOccurence>0</MinOccurence>
    <MaxOccurence>1</MaxOccurence>
    <BusinessProcessContext>Trade</BusinessProcessContext>
    <ProductContext>In All Contexts</ProductContext>
    <IndustryContext>In All Contexts</IndustryContext>
    <GeopoliticalContext>In All Contexts</GeopoliticalContext>
    <OfficialConstraintsContext>None</OfficialConstraintsContext>
    <RoleContext>In All Contexts</RoleContext>
    <SupportingRoleContext>In All Contexts</SupportingRoleContext>
    <SystemConstraintsContext>In All Contexts</SystemConstraintsContext>
</BusinessInformationEntity>
```

Figure 3.15: XML Representation of Association Business Information Entity

```
<BusinessInformationEntity>
    <UniqueID>UN01002096</UniqueID>
    <EntryType>BBIE</EntryType>
    <DictionaryEntryName>Unstructured_ Address. Line Two. Text</DictionaryEntryName>
    <Definition>The second free form line, expressed as text,
                of the unstructured address.</Definition>
    <ObjectClassTermQualifier>Unstructured</ObjectClassTermQualifier>
    <ObjectClassTerm>Address</ObjectClassTerm>
    <PropertyTermQualifier/>
    <PropertyTerm>Line Two</PropertyTerm>
    <RepresentationTerm>Text</RepresentationTerm>
    <DataTypeQualifier/>
    <DataType>Text. Type</DataType>
    <QualifiedDataTypeUniqueID/>
    <BusinessTerms/>
    <MinOccurence>0</MinOccurence>
    <MaxOccurence>1</MaxOccurence>
    <BusinessProcessContext>In All Contexts</BusinessProcessContext>
    <ProductContext>In All Contexts</ProductContext>
    <IndustryContext>In All Contexts</IndustryContext>
    <GeopoliticalContext>In All Contexts</GeopoliticalContext>
    <OfficialConstraintsContext>None</OfficialConstraintsContext>
    <RoleContext>In All Contexts</RoleContext>
    <SupportingRoleContext>In All Contexts</SupportingRoleContext>
    <SystemConstraintsContext>In All Contexts</SystemConstraintsContext>
</BusinessInformationEntity>
```

Figure 3.16: XML Representation of Basic Business Information Entity

The class diagram of the client side objects is in Figure 3.18. Furthermore, in addition these object models for graphical visualization, property view for editing values of properties and

```
<DataType>
    <UniqueID>UN6</UniqueID>
    <EntryType>DT</EntryType>
    <DictionaryEntryName>Amount. Type</DictionaryEntryName>
    <Definition>An amount with the corresponding currency unit.</Definition>
    <CoreComponentType>Amount</CoreComponentType>
    <RepresentationTerm>Amount</RepresentationTerm>
    <DataTypeQualifier>Unqualified</DataTypeQualifier>
</DataType>
<DataType>
    <EntryType>SC</EntryType>
    <DictionaryEntryName>Amount. Currency. Code</DictionaryEntryName>
    <Definition>Represents the currency unit.</Definition>
    <PrimitiveType>CurrencyCodeList</PrimitiveType>
    <Enumeration>ISO 4217 2001 5</Enumeration>
</DataType>
<DataType>
    <EntryType>CC</EntryType>
    <DictionaryEntryName>Amount. Content</DictionaryEntryName>
    <Definition>An amount with the corresponding currency unit.</Definition>
    <PrimitiveType>Decimal</PrimitiveType>
    <Enumeration/>
</DataType>
```

Figure 3.17: XML Representation of Data Type

tree node view are implemented for each class object model.

3.2.3 Customizing a Document Model

The UBL was designed on an 80/20 principle according to Pareto's Law[1], which states 20% of features accommodates 80% of requirements [27]. In order to satisfy the remaining 20% of the eBusiness requirements the business document schemas are needed to be customized. Therefore, nearly most of the large and small enterprises need to generate their own business document customizations from standard document schemas or design their property solution specific business document schema or component by following UBL Customization methodology [6]

As described in Section 2.2.1, UBL provides guidelines for two types of customization, which are conformant and compatible. These types describe the extent to which generated documents schemas are interoperable with UBL standard document schemas. To summarize,

[1] The Pareto Law (also known as the 80-20 rule, the law of the vital few, Haddad's Theorem and the principle of factor sparsity) states that, for many events, roughly 80% of the effects come from 20% of the causes.

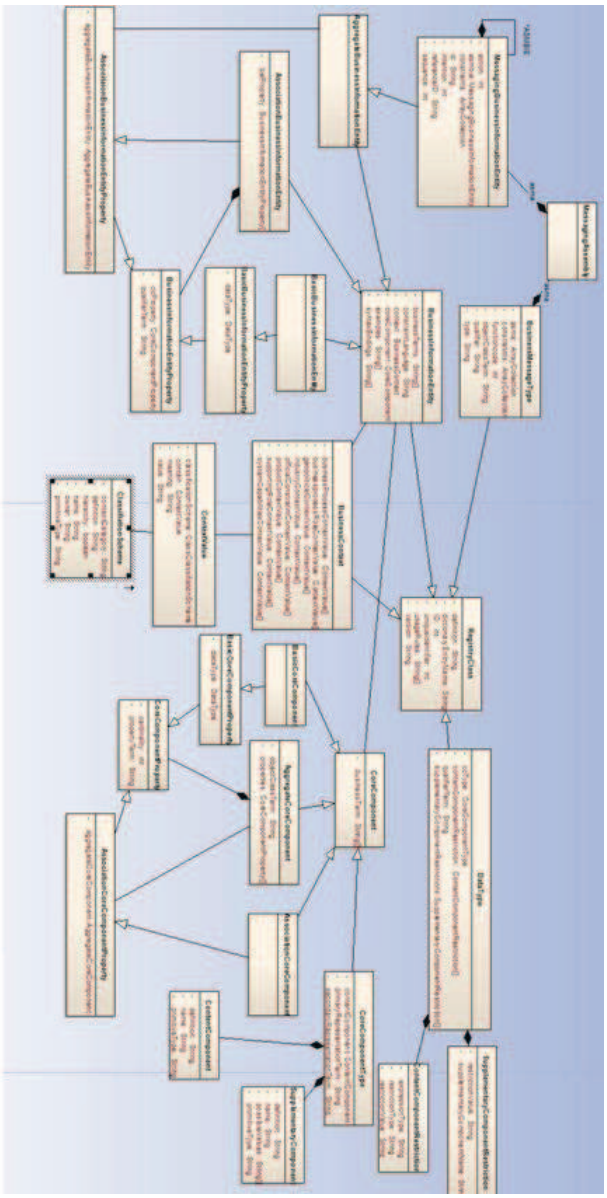

Figure 3.18: Class Diagram of UN/CEFACT CCTS Entities

UBL Conformant Customization Guidelines states that if the output customized schema is still valid for a UBL standard schema, whereas UBL Compatible Customization Guideline allows users to customize the UBL standard schema with exceptions, but it states that it follows the methodology that is behind the UBL. The modeling environment supports both types of customization recommended by UBL standard: Conformant Customization and Compatible Customization. iSURF eDoCreator allows (1) subsetting source document model, (2) extending source document model, (3) constraining document artifacts and (4) creation of

new document artifacts from scratch (5) constraining the schema with Schematron rules. All the customization features - except 5th one - is provided under Customization menu, which appears in the context menu which pops up when a right click is triggered on a document building block. The Schematron Editor is available under the Tools menu. The difference between types of customization is not explicitly differentiated in the tool with separate functionalities; instead users are warned as they operate an action which will result in compatible design. For example, if a mandatory element is discarded from the document model or if the minimum cardinality of the document artifact is decreased, this action violates the UBL conformance design rules. Accordingly, the warning reminds users that the design will be compatible if the action is realized and let the user withdraw the action. Once a user accepts the compatible design warning, and continues to execute the action, warning mechanisms deactivates it and no warning is displayed later since the document is conformant for that time being.

The customization process for document building blocks is started with the selection of a document building block from the list of available document building blocks. User is expected to right click on the selected component and select ”Customize” from the initialized context menu.

3.2.3.1 Subsetting or Extending a Document Content Model

For the customization process, the tree graphical view is presented as a check tree in order to let the user subset the document model according to their needs. Document schema can either be extended or subset. These modifications can be realized in two ways: either by deleting or adding Association Business Information Entity Properties or Basic Business Information Entity Properties. The deletion is done via right clicking on the tree node component that will be deleted and choosing the option of deletion, respectively. The addition is done via dragging building blocks representations from the left panel to the right panel as presented in Section 3.2.2.2.

Furthermore, the user can customize Business Information Entities encapsulated in selected document building block. Figure 3.19 illustrates how to subset document artifact schemas from tree interface. In the interface, when it is initialized, the document artifacts which are mandatory, i.e. minimum cardinality is equal to at least one, are presented as selected and in

red text. When user decides to exclude it from the document schema, a warning is dispatched to recall that eliminating mandatory element is against to conformant design rules and this may cause interoperability problems. Furthermore, the user may also include some optional elements, i.e. elements of which minimum cardinality is equal to zero, to its document design. When user sets one of the optional document building blocks as mandatory, the inclusion of new information entities in its ancestral path, the optional elements in an ancestral path are also included as mandatory in the design. Alternatively, when an element from the upper levels is excluded, encapsulated elements of that excluded elements are automatically excluded from the document schema. This could be regarded as a ripple effect.

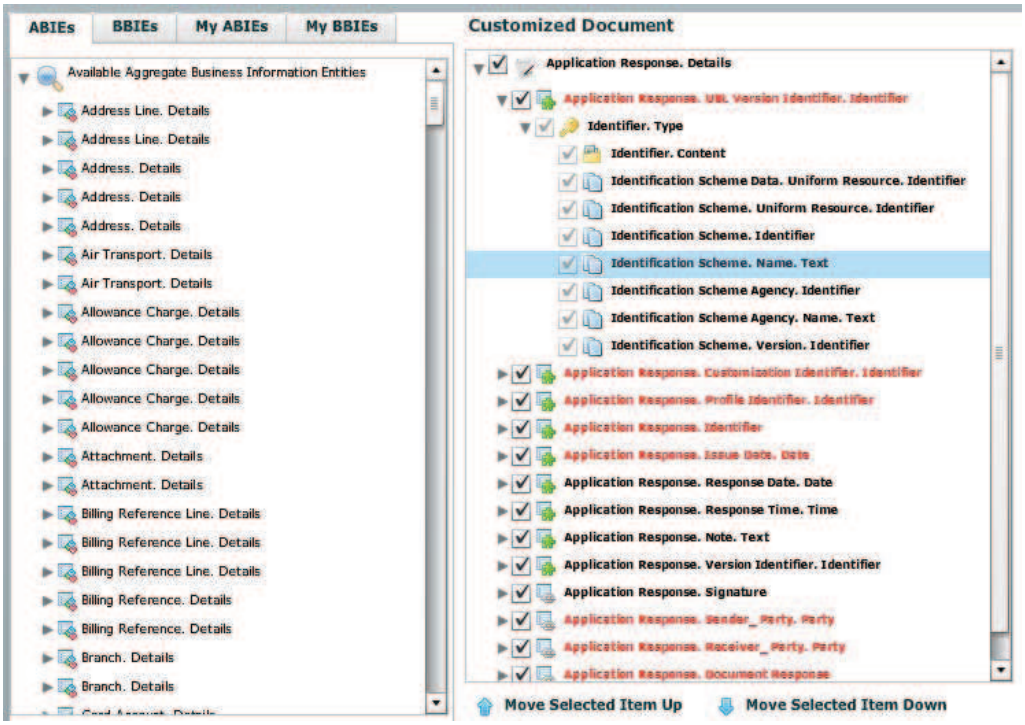

Figure 3.19: Subsetting the document model

3.2.3.2 Changing Cardinality Values of Document Building Blocks

In addition to subsetting, the occurrence number of the document building blocks may be changed through User Interface via spinners as in Figure 3.20. The ”Set maximum cardinality to ’unbounded’” checkbox sets the maximum cardinality to unbounded value without altering the spinners’ value. The warning mechanism also checks cardinality values. Mini-

mum cardinality can be increased to its maximum and maximum cardinality can be decreased to minimum, otherwise a warning alert is dispatched. Furthermore, in the tool since minimum cannot be higher than maximum cardinality, or vice versa, it automatically adjusts the cardinality values with respect to the other.

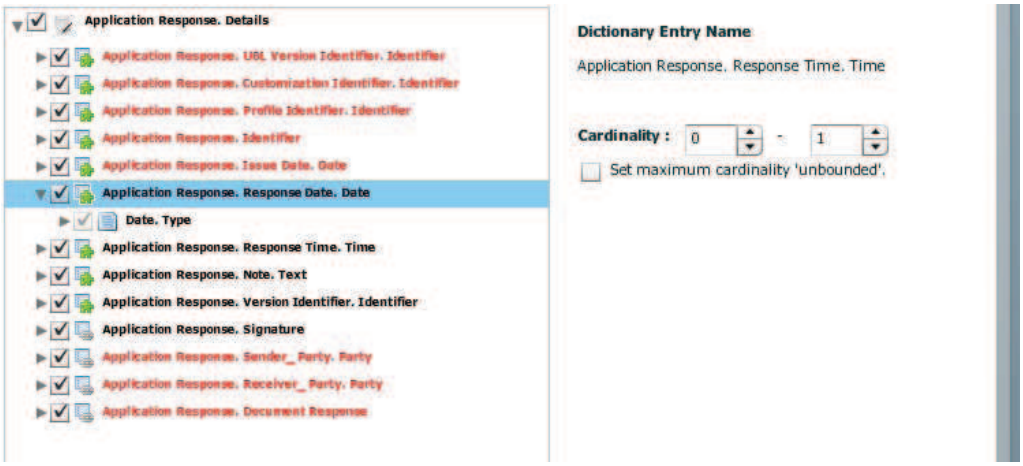

Figure 3.20: Changing Cardinality

3.2.3.3 Changing Business Context Values

One way of constraining document artifacts is editing Context Value of document artifact according to context in which business process take places. Therefore, in the tool, user can change the context values of Business Information Entities for a customization process. In order to define appropriate document schema, which reflects your industry, product, geopolitical etc. requirements, the context in which business process takes place shall be analyzed: so the user interface asks for the context values of the target business document schema, since context is a key driver to precisely determine business meaning and intent of information definition and exchange. When user clicks on the root node of the being customized document building block, on the right most panel a link button is shown with ”View/ Change Business Context”. By clicking on this link button, user can initiate the context value assignment process. There are eight different Context Categories introduced by the UN/CEFACT CCTS: Business Process Context, Product Classification Context, Industry Classification Context, Geopolitical Context, Official Constraints Context, Business Process Role Context, Supporting Role Context, and System Capabilities Context. These context categories are introduced

in Core Components Technical Specification as follows [4]:

- Business Process Context - The primary Context Category, and provides many useful distinctions in the analysis of Core Components.
- Product Classification Context -specifies the goods or services involved in the collaboration.
- Industry Classification Context -specifies a particular industry from which parties are involved in collaboration.
- Geopolitical Context - specifies where the business process is to be conducted.
- Official Constraints Context - specifies any legal restrictions or requirements on the business process.
- Business Process Role Context - Every partner in a Business Process data exchange has a particular role - buyer, seller, etc. It specifies the role played by the user and their trading partners.
- Supporting Role Context - specifies the significant parties other than receiver or sender that will be using the data
- System Capabilities Context - specifies any major restrictions that can be caused capability of the legacy systems.

By using the panel presented in the Figure 3.21, user can assign some values to context categories in order to specify main characteristics of a business process for which the document schema is designed. In the panel, each Context Category is shown in a separate tab. To add a search value for a specific Context Category, the user shall click on corresponding tab of that specific Context Category and then click on ”Add” button after selected the Context Value, or drop the selected value onto the context category table. A context category may use more than one Classification Schema. For example, the Industry Context values can be assigned from Universal Standard Product and Service Specification or International Standard Industry Classification Schemes. Some defined context values are loaded to the tool as declared in UN/CEFACT CCTS guidelines. For example, UN/CEFACT Catalogue of Common business processes is loaded for adding Context Values for business process category. UN/CEFACT

CCTS guideline also recommends using following recognized code lists that provide authoritative sources for different context categories:

- Business Process Context
  - UN/CEFACT Catalogue of Common business processes
- Product Classification Context
  - Universal Standard Product and Service Specification (UNSPSC)
    - Custodian: GS1
  - Standard International Trade Classification (SITC Rev .3)
    - Custodian: United Nations Statistics Division (UNSD)
  - Harmonized Commodity Description and Coding System (HS)
    - Custodian: World Customs Organization (WCO)
  - Classification Of the purposes of non Profit Institutions serving households (COPI)
    - Custodian: UNSD
- Industry Classification Context
  - o International Standard Industrial Classification (ISIC)
    - Custodian: UNSD
  - Universal Standard Product and Service Specification (UNSPSC) Top-level Segment [digits 1 and 2] used to define industry
    - Custodian: ECCMA
- Geopolitical Context
  - Country - ISO 3166.1

To delete a previously added Context Value from the list of context values, user shall select a value from the context category table and drop it to the basket icon; the deletion of Context Value may also be realized by clicking on the basket icon after selecting a value from the table.

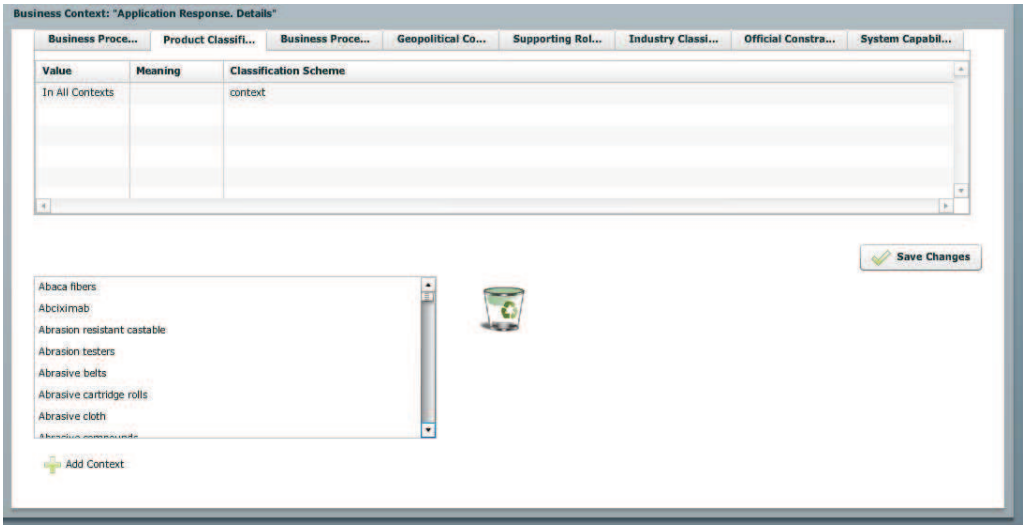

Figure 3.21: A Panel for Constraining Context Values

Furthermore, for ”Official Constraint” and ”System Capabilities Context”, users are allowed to create its own Custom Context Value by declaring its Classification Scheme, Value and Meaning after checking the ’Create custom Context Value’ box (Figure 3.22).

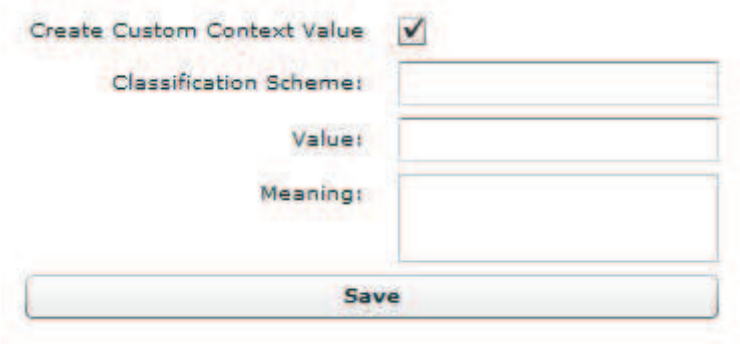

Figure 3.22: A Panel for Custom Context Value

By clicking the ”Save Changes” button user can finalize the constraining document context values.

3.2.3.4 Schematron Editor for Constraining a Document Model

An XML schema is used in most of the applications to model the structure of an XML document. It can specify the valid elements that can occur in a document, the order in which they can occur, and express constraints on certain aspects of these elements. However, business organizations may need to define some business rules or constrain the values used in UBL documents in a way that XSD validation cannot specify easily. These kinds of assertions can be expressed with Schematron [22] language, which is a small language for making assertions about the presence or absence of patterns in XML documents. For example, Denmark and Sweden define subsets of UBL 2.0 for customization by layering on business rules implemented in Schematron.

In the iSURF eDoCreator a document specific Schematron Editor is implemented to handle creation of Schematron files through graphical interfaces. This functionality is available under the Tools menu.

When the main screen is initialized, the list of all visible document models is presented to the user in the left-most panel. This list enables the user examine the document models and select the one on which they will work on.

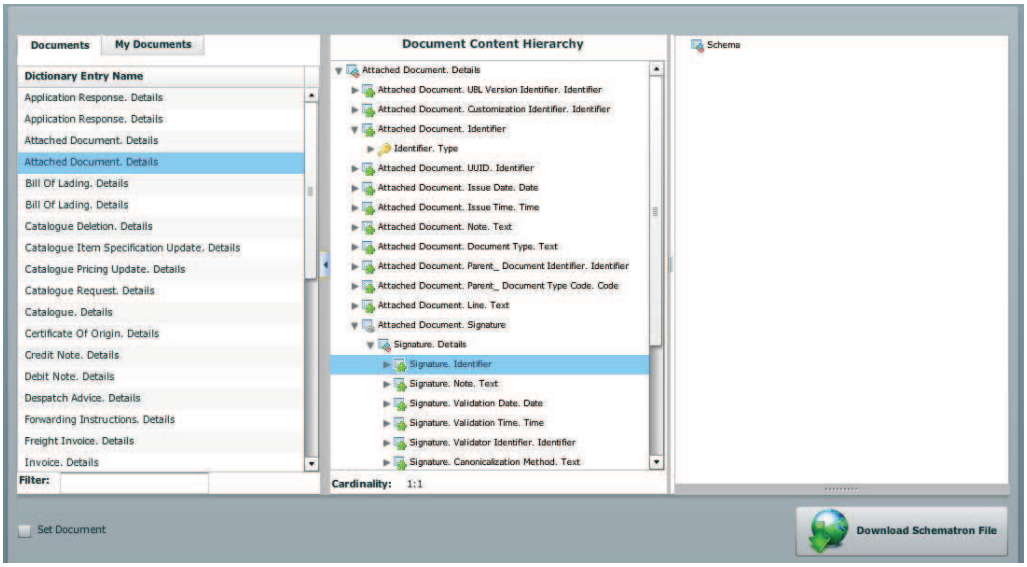

Figure 3.23: Schematron Editor

When a document model is clicked on, on the middle panel graphical presentation of the selected document model is initialized as described in Section 3.2.1. After the document model on which constraints will be defined is decided, user can set the document model by checking the 'Set Document' check box. Then left-most panel becomes invisible, but user can redo it by unchecking the checkbox. As seen in 3.23, on the right-most panel XML structure of schematron file is presented with Schematron root node. In the XML structure, there are title element, namespace element and pattern elements, which are represented as children of this root node. The schematron tree is populated with pattern nodes as user defines new schematron patterns as shown in Figure 3.24.

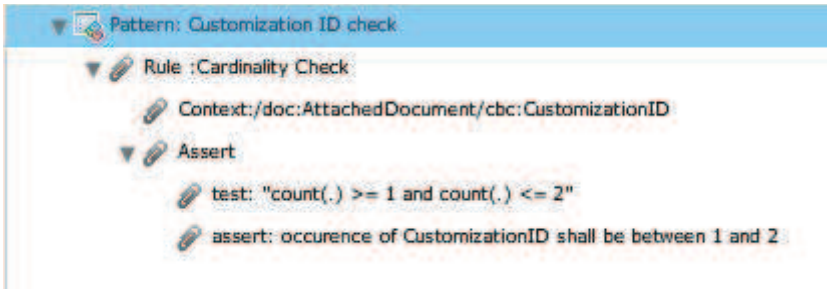

Figure 3.24: Schematron Pattern Representation

Furthermore, in the tree namespace and title are also presented as nodes as shown in Figure 3.25.

Figure 3.25: Schematron Title and Namespace Element Representation

While forming schematron file all processes are handled graphically from the user interface.

Setting Title Element

Title element is used in schematron files for a human readable title. Users can add a title either

by right clicking on root node and selecting ”Add Title” option or clicking the upper menu. When user selects adding title option, user is requested to enter a title and press ”Set” button, which appears on the right-down corner as seen in the Figure 3.26.

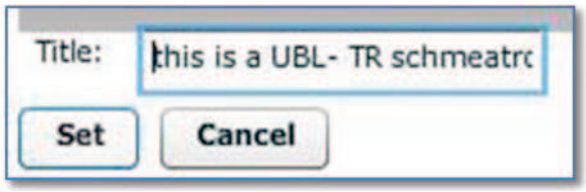

Figure 3.26: A Panel for Setting Title Element

Setting Namespace and Prefix Elements

Schematron can also be used to validate XML instance documents that use namespaces. Each namespace used in the XML instance document should be declared in the Schematron schema. The element used to declare namespaces are the ns element which should appear as a child of the schema element. The ns element has two attributes, Uri and prefix, which are used to define the namespace URI and the namespace prefix [22]. Setting this element is similar to Adding Title and this time you need to select ”Add Namespace” and enter URI and prefix values as seen in the Figure 3.27.

Furthermore, in the tool prefix and URIs automatically assigned when a document building block is selected from the document model. It retrieves namespace and its prefix from the document building blocks and uses these namespace values while constructing the schematron file.

Figure 3.27: A Panel for Setting Namespace Element

Adding Pattern Elements

Schematron is as flexible as to define a variety of different constraints on a document model. In this document specific Schematron Editor we address the common schema needs. Available patterns that can be defined by using the editor are available under a menu item.

- ***Validating mandatory elements*** Users can validate that certain elements are present. For setting an element as mandatory users need to select the element and assign a name for this pattern. Furthermore, they can set the text that will be displayed when validation does not perform.

  The panel is shown in Figure 3.28. Node Path is automatically set according to the selected node. 'View Pattern' link button pops a bottom panel up. The bottom panel visualizes XML excerpt for the generated pattern. "Cancel" button closes this pop-up view and "Save" button adds this pattern to the Schematron File tree as shown in Figure 3.28. If all the input fields required to build the pattern is not complete, a warning is displayed in a red text at the bottom corner.

- ***Validating non-applicable elements*** Similar to mandatory elements it is sometimes needed to exclude some elements from the document model. For this pattern, a panel is also developed. The excluded document check panel can be shown in Figure 3.29.

- ***Validating cardinality of elements***

  A constraint which specifies whether a specific number of a particular element needs to be included in the message can be checked with a schematron pattern. To check the cardinality value user can select any comparison function, i.e. =, !=, <, <=, >, >= from the menu. For comparison value user can enter an input value or choose to compare it with the cardinality of another document building block by clicking on "..." button. User can constrain both minimum cardinality and maximum cardinality. Furthermore, user can set a comparison relation between minimum cardinality and maximum cardinality with "and" or "or" Boolean clauses. The panel is as in the Figure 3.30.

- ***Validating presence of a value***

  User can validate that an element appears in the instance, or it has a value. For example, an <author/> element is not valid while <author>Tom Robbins</author> is a valid element for the validation of presence of a value. For this kind of nullity checks, user can

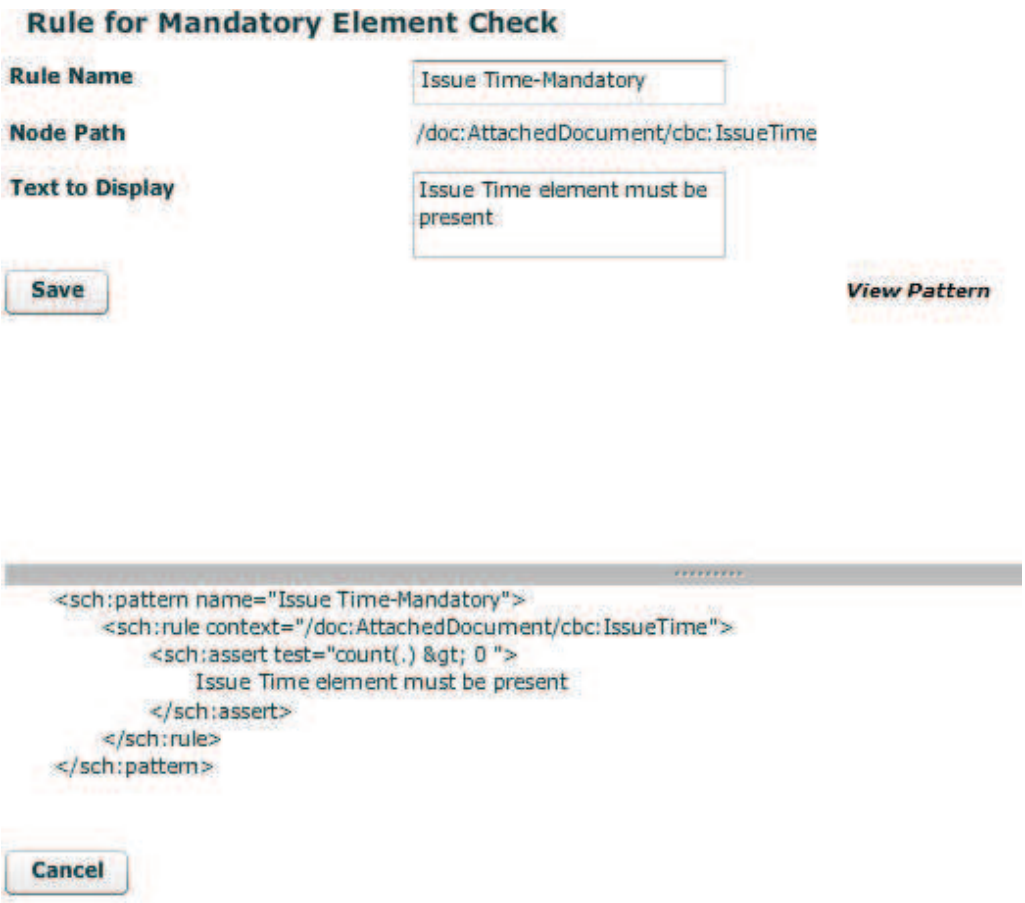

Figure 3.28: A Panel for Creating a Pattern for Checking Mandatory Elements

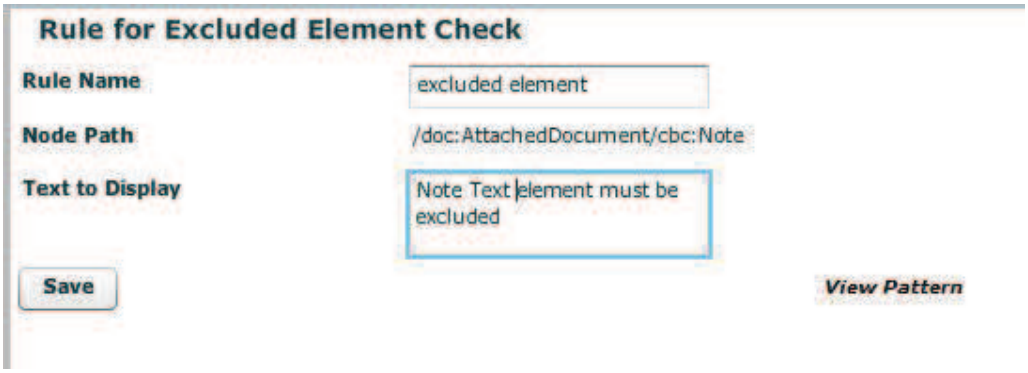

Figure 3.29: A Panel for Creating a Pattern for Checking Excluded Elements

specify a schematron pattern. The implemented panel is the same with the mandatory element check, but the pattern generated by the tool is different. User needs to select a

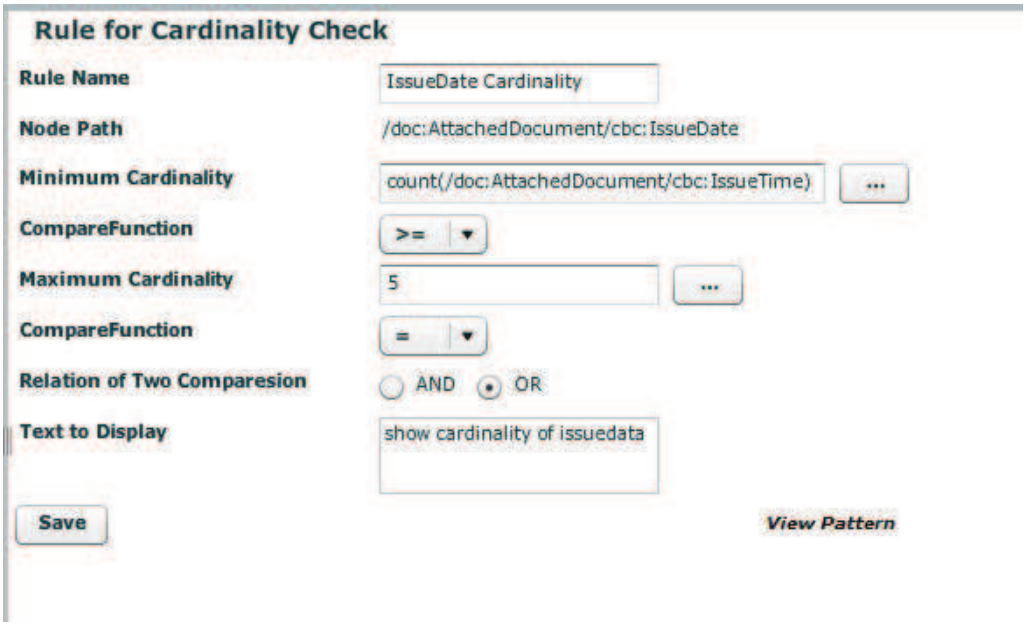

Figure 3.30: A Panel for Creating a Pattern for Cardinality Check

node from the document model, set the rule name and enter a text message.

- ***Validating element content***

  To validate that an element has a certain value, user can select the ”Validating matching value of an element” menu item. For this comparison, user can add a value from text input, compare the value with value of another node or upload a genericode file as seen in the Figure 3.31. Furthermore, user can also realize any combination of these features.

  For example, in UBL-TR schemeID attribute of cac:PartyIdentification/cbc:ID shall be one of TCKN VKN HIZMETNO MUSTERINO TESISATNO TELEFONNO DISTRIBUTORNO TICARETSICILNO TAPDKNO BAYINO ABONENO SAYACNO. This kind of constraints can be declared through element content validation. For implementing such a constraint, user can upload a genericode file which specifies possible values, and then schematron pattern can be automatically generated with the tool. Genericode file is uploaded to the server side and then parsed to retrieve the code values. Then coded values are passed to the client side to formalize the test value.

- ***Validating conditional presence or exclusion***

  A co-occurrence constraint constrains one or more components of document content

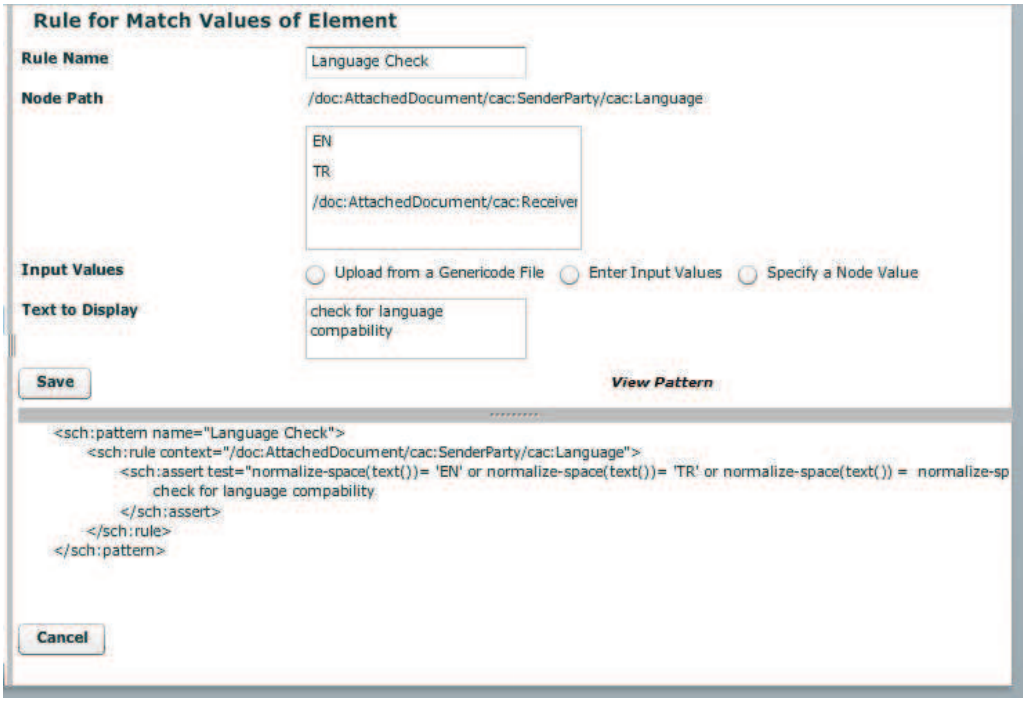

Figure 3.31: A Panel for Creating a Pattern for Element Content Check

based on one or more other components of document content. The basis can be the presence or absence of content or particular values of content. For example, one could assert that for each itemized information entity that is based on the UBL party, cac:PartyIdentification/cbc:ID must be present. For this kind of restrictions a conditional presence or exclusion panel can be used as seen in Figure 3.32.

There are four types of options:

- ***Mutual Presence:***

  This means that the second specified element can exist depending on the existence of the first specified element. In other words, if first selected element exits then the second selected element must exist or if first one does not exist, then second one must not exist.

- ***Mutual Exclusion:***

  This means that the second specified element can exist depending on the exclusion of the first specified element. In other words, if first selected element exits then the second selected element must not exist or if first one does not exist, then second one must not exist.

– ***Conditional Presence:*** This means that the first specified element implies the existence of the second specified element. In other words, if first selected element exits then the second selected element must exist otherwise second one may or may not exist.

– ***Conditional Exclusion:*** This means that the first specified element implies the non-existence of the second specified element. In other words, if first selected element exits then the second selected element must not exist otherwise second one may or may not exist.

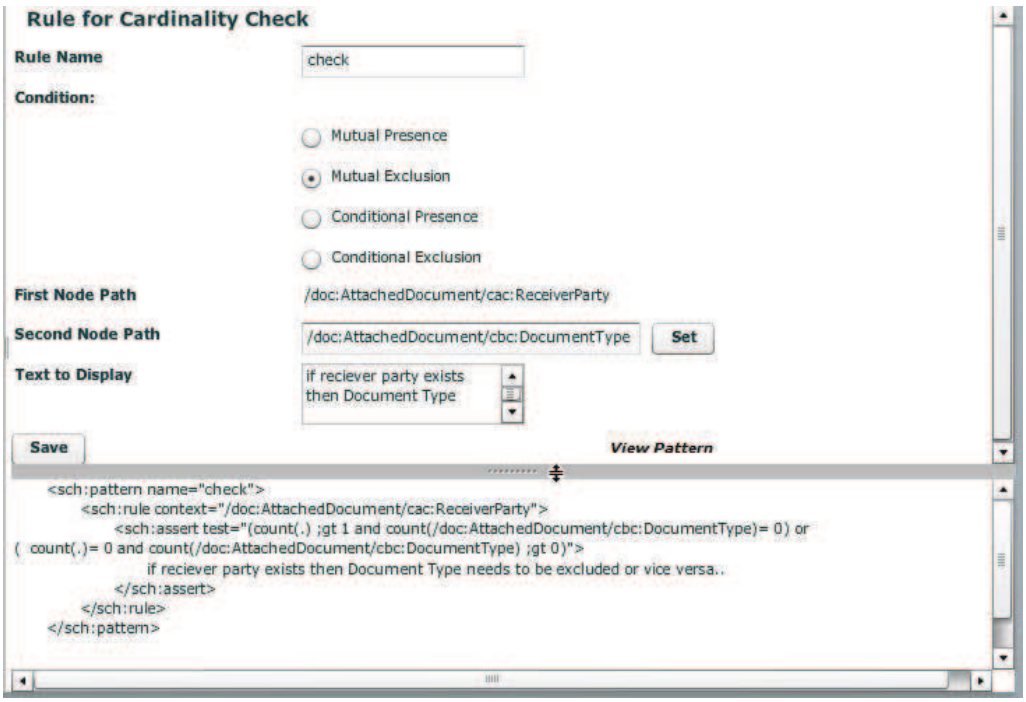

Figure 3.32: A Panel for Creating a Pattern for Conditional Presence or Exclusion Check

Similar to other patterns users needs to enter a rule name, set the first element and the second element and enter a text message to be displayed for completing the creation of the schematron pattern. The first element is the element that is selected from the document model displayed at the left-hand side, while the second element is set from a pop-up window which is opened when ”Set” button is clicked.

Retrieving Schematron File

After saving all constraints, a tree model representing the schematron file is formed as in Figure 3.33. The tree can be updated by adding new patterns via menu items or deleting the title element, namespaces elements or pattern elements by right click on the nodes.

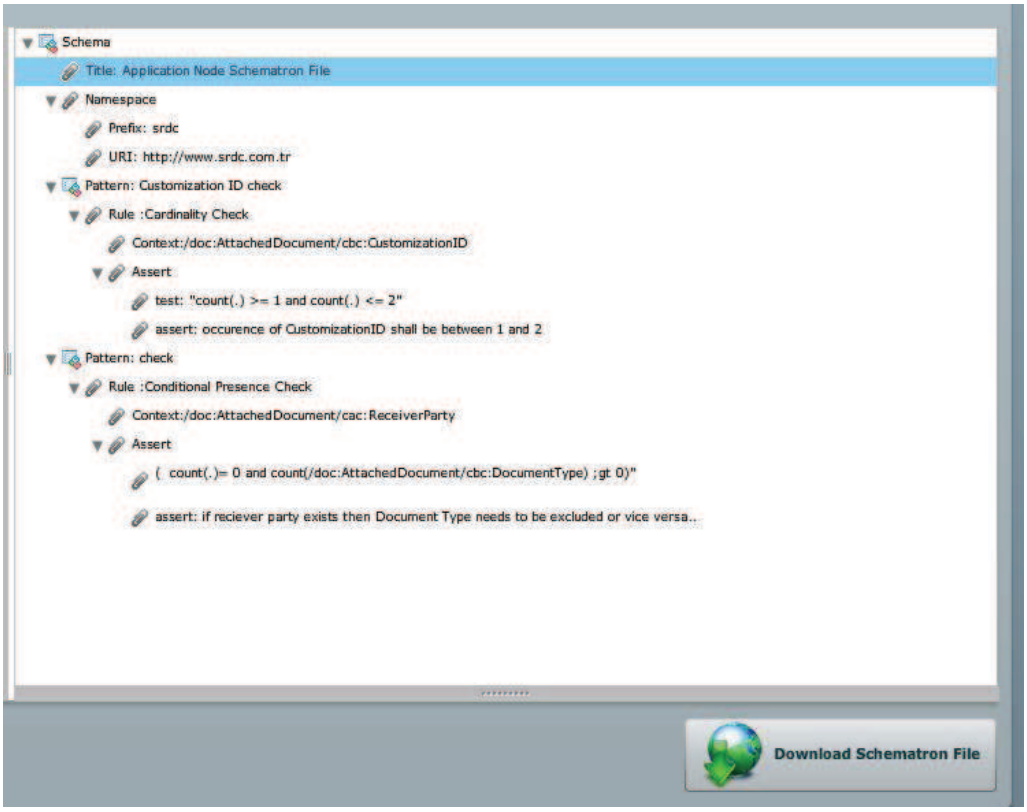

Figure 3.33: The Representation of a Schematron File

Finally, user can download the schematron file named with the document model Dictionary Entry Name by clicking Download Schematron File button. This file processing mechanism is handled at the server side and the file retrieved by the client side via its URL. A sample generated schematron file can be shown in Figure 3.34.

3.2.4 Enriching the Repository, Uploading a New Document Model

As new document models are generated by standard bodies or users, the generated spreadsheet models which are created externally without using the tool can be uploaded to the repository and features of the iSURF eDoCreator can be utilized and the models can be shared with a

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:schema xmlns="http://www.ascc.net/xml/schematron"
    xmlns:sch="http://www.ascc.net/xml/schematron"
    xmlns:doc="urn:oasis:names:specification:ubl:schema:xsd:Order-2"
    xmlns:cbc="urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2"
    xmlns:cac="urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2">
<sch:ns prefix="doc" uri="urn:oasis:names:specification:ubl:schema:xsd:Order-2" />
<sch:ns prefix="cbc" uri="urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2" />
<sch:ns prefix="cac" uri="urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2" />
    <sch:pattern name="check for Order">
        <sch:rule context="/doc:Order/cbc:ID">
            <sch:assert test="count(.) &gt; 0 ">
                mandatory element must be present
            </sch:assert>
        </sch:rule>
    </sch:pattern>
    <sch:pattern name="fdsfdsfs">
        <sch:rule context="/doc:Order/cac:TaxTotal">
            <sch:assert test="count(.) = 0 ">
                Exclude element must be excluded
            </sch:assert>
        </sch:rule>
    </sch:pattern>
    <sch:pattern name="nodevalues">
        <sch:rule context="/doc:Order/cbc:SalesOrderID">
            <sch:assert test="count(.) &gt;= count(/doc:Order/cac:AllowanceCharge)
                    and
                        count(.) &lt;= count(/doc:Order/cac:OrderLine/cac:LineItem)">
                test
            </sch:assert>
        </sch:rule>
    </sch:pattern>
    <sch:pattern name="presence">
        <sch:rule context="/doc:Order/cac:OrderLine/cac:LineItem/cbc:ID">
            <sch:assert test="text() != '' ">
```

Figure 3.34: An Excerpt from a Sample Schematron File

wider community.

UBL uses spreadsheet files for the representation of an assembly model in tabular form; a spreadsheet file is available for each document, which describes the assembly of components into specific types of documents. The spreadsheet models use rows to define Basic Business Information Entities and Association Business Information Entities as shown in Figure 3.35. Each business information entity (BIE) is defined in a single row. Row background color distinguishes between BBIE (white), ABIE (pink), and ASBIE (green). The details of associated Business Information Entities are found in different spreadsheets: Basic Business Information Entities are stored in CommonBasicComponents spreadsheet and Aggregate Business Information Entities are stored in CommonAggregateComponents spreadsheet. Columns of the spreadsheets define the metadata associated with each component type. Many of the spreadsheet columns are determined by UN/CEFACT CCTS requirements.

From the main menu panel, user can select uploading a document model from either Microsoft

```
<?xml version="1.0" encoding="UTF-8"?>
<sch:schema xmlns="http://www.ascc.net/xml/schematron"
    xmlns:sch="http://www.ascc.net/xml/schematron"
    xmlns:doc="urn:oasis:names:specification:ubl:schema:xsd:Order-2"
    xmlns:cbc="urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2"
    xmlns:cac="urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2">
<sch:ns prefix="doc" uri="urn:oasis:names:specification:ubl:schema:xsd:Order-2" />
<sch:ns prefix="cbc" uri="urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2" />
<sch:ns prefix="cac" uri="urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2" />
    <sch:pattern name="check for Order">
        <sch:rule context="/doc:Order/cbc:ID">
            <sch:assert test="count(.) &gt; 0 ">
                mandatory element must be present
            </sch:assert>
        </sch:rule>
    </sch:pattern>
    <sch:pattern name="fdsfdsfs">
        <sch:rule context="/doc:Order/cac:TaxTotal">
            <sch:assert test="count(.) = 0 ">
                Exclude element must be excluded
            </sch:assert>
        </sch:rule>
    </sch:pattern>
    <sch:pattern name="nodevalues">
        <sch:rule context="/doc:Order/cbc:SalesOrderID">
            <sch:assert test="count(.) &gt;= count(/doc:Order/cac:AllowanceCharge)
                    and
                        count(.) &lt;= count(/doc:Order/cac:OrderLine/cac:LineItem)">
                test
            </sch:assert>
        </sch:rule>
    </sch:pattern>
    <sch:pattern name="presence">
        <sch:rule context="/doc:Order/cac:OrderLine/cac:LineItem/cbc:ID">
            <sch:assert test="text() != '' ">
```

Figure 3.35: An Excerpt from a Sample Spreadsheet Document

Excel .xls file or OpenOffice Calc .ods file, which is formatted according to UBL Spreadsheet Design Rules. Furthermore, user can validate the spreadsheet to check whether it is consistent with the UBL Spreadsheet Design Rules and provides feedback for letting the user update the spreadsheet model.

3.2.4.1 Validating a Document Spreadsheet Model

The spreadsheet models generated outside of the tool can be uploaded to the iSURF eDoCreator environment as mentioned previously. However, in order to check its consistency with UBL spreadsheet design rules, a validation mechanism shall need to be implemented. Furthermore, providing feedback to user is vital for the users, since there is no mechanism which helps users to correctly format their spreadsheet model.

For this purpose, iSURF eDoCreator provides validation and feedback mechanisms which check for following guideline rules:

1. Column Header Check: In the UBL Spreadsheet, there are thirty one headers in a row, which describe document building blocks attributes found in a spreadsheet. They are as UBL Name used in XSD file, Dictionary Entry Name, Object Class Qualifier, Object Class, Property Term Qualifier, Property Term, Representation Term Index, Data Type Qualifier, Data Type, Associated Object Class Qualifier, Associated Object Class, Business Terms, Cardinality, Component Type, Definition, Examples, Version, Business Process Context, Geopolitical Context, Official Constraints Context, Product Context, Industry Context, Role Context, Supporting Role Context, System Constraints Context in their column index order. One feature of the validation mechanism is to check whether the headers are well formed and in their presence order. Furthermore, it checks if there exists any column header value, which is not recognized.

2. UBL Name Syntax Check: UBL provides Naming and Design Rules, which regulates how to transform the conceptual model to a physical model. According to these regulations, every document building block and every attribute has a unique representation at a syntactic level. In the Spreadsheet model UBL formalized these with a spreadsheet formula UBL Names according to UBL Naming and Design Rules. The formula is as follows:

   - =SUBSTITUTE(SUBSTITUTE(CONCATENATE(IF(E12=”Universally Unique”-;”UU”;E12);IF(G12¡¿I12;H12;F12);CONCATENATE(IF(I12=”Identifier”;”ID”;-IF(I12=”Text”;””;I12))));” ”;””);”’”;””)

   This formula reimplemented in Guideline Execution Engine as UBLNameGenerator which retrieves the data from rows and columns to find out correct UBLName and then compares it with the user input to see whether they are consistent or not . If not, it provides a feedback to the user declaring that ”UBL Name is not formed correctly”.

3. Existence check of Associated Aggregate Business Information Entity of Association Business Information Entity: As mentioned in Section 2.1.1 every Association Business Information Entity encapsulates an Aggregate Business Information Entity and relates an Aggregate Business Information Entity with another one. In the spreadsheet model, by retrieving Object Class Qualifier and Object Class, Dictionary Entry Name of the associated Aggregate Business Information Entity can be obtained. This validation mechanism checks for if that Aggregate Business Information Entity is available in the

repository. Otherwise, it declares that it refers to an Aggregate Business Information Entity that does not exist.

4. Existence check of Data Type of Basic Business Information Entity: As mentioned in Section 2.1.1 every Basic Business Information Entity is represented by a Data Type. This validation mechanism checks for whether the Data Type exists or not in the repository and provides feedback to the user according to result. For this purpose, first of all, it obtains Dictionary Entry Name of Data Type from Data Type Qualifier and Representation Term of Basic Business Information Entity and executes a query on the repository for inquiring Dictionary Entry Name of Data Type.

5. Dictionary Entry Name Check: Every document building block has a Dictionary Entry Name, which is formed according to UBL Naming and Design guidelines. They basically formalized from the concatenation of Object Class Term, Property Term and Representation term. This mechanism checks every row of the spreadsheet to see whether they are well-formed according to these guidelines.

6. Cardinality Check: This mechanism controls if the cardinality cell of the spreadsheet contains either an integer value or unbounded cardinality symbol.

7. Entity Check: In the document spreadsheet model Aggregate Business Information Entity, Basic Business Information Entity and Association Business Information Entity are allowed to be declared. This mechanism checks for if the document consists of any document building blocks other than allowed ones.

3.2.4.2 Submitting a Spreadsheet Model to the Repository

The uploaded spreadsheet model is saved to the repository if the uploaded spreadsheet is valid. For this purpose, it parses the uploaded document file, which is either .ods file or .xls file and submit them to the repository through SQL queries. It parses the file row by row and updates the tables in the repository accordingly and creates a new entity for the loaded document model. Thus, later they can be retrieved from the repository and graphical presentation can be formed.

3.2.5 Producing the Documentation of Designed Document Model

The documentation of the conceptual model graphically represented in the environment can be exported to the outside of the tool in a zip archive package format which includes XML Schema (XSD) documents of the UBL format, its spreadsheet models and genericode files. The zip archive folder has a structure as shown in Figure 3.36 for a downloaded Bill of Lading document model.

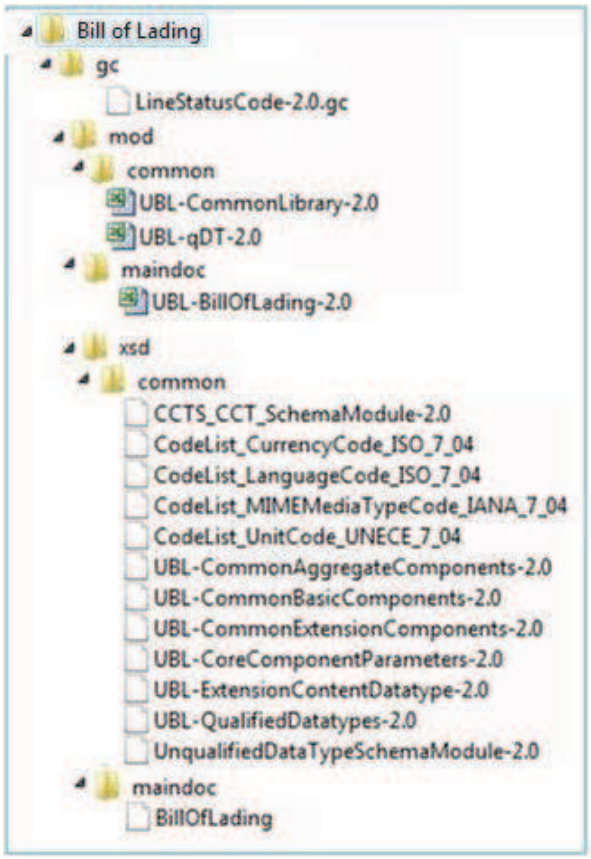

Figure 3.36: Folder Structure

UBL uses the W3C XML Schema Definition Language, which is experiencing the most widespread adoption. Although other schema languages exist that offer their own advantages and disadvantages, UBL has determined that the best approach for developing an international XML business standard is to base its work on W3C XSD.

The conceptual models represented in the graphical view can form the basis for XML schema for related business documents. The process of deriving these schemas is very rigorous and

there should be a unique way of representing these data schemas. The UBL TC has a subcommittee for UBL Naming and Design Rules (NDR), which created a comprehensive document with rules for naming and overall design in UBL XML components in order to provide the uniqueness. For example, it says that ”Names for XML constructs must use camel-case capitalization, such that each internal word in the name begins with an initial capital followed by lowercase letters (example: AmountContentType).” All the documentation generated within the iSURF eDoCreator is consistent with these guidelines.

The xsd folder found in the zip archive file contains maindoc and common directories, which contain XSD schema of the Message Assembly and common XSD documents of common components such as Aggregate Core Components and Basic Core Components, respectively.

The models persisted in the repository are automatically serialized to XSD files according to Naming and Design Rules of UBL. In the persistence layer, conceptual models of document building blocks are stored and documentation engines derive XML schema artifacts from them to form XSD files when user initiates the transformation of core components to the appropriate XML schema representation. The XML schema generator automatically detects dependencies in the document model and generates additional XMLschema for Common Basic Components and Common Aggregate Components. The XML artifacts form the logical level business document model to which every document instance exchanged between two B2B systems must comply with.

The schema of the Message Assembly begins with the definition of some namespaces and some import elements. Then the Message Assembly is defined as XML element and its type is declared as a complex type. The XML schema generator iterates over every Association Business Information Entity in the Message Assembly and constructs a complexType with a sequence for each. As shown in Figure 3.37 Cross Invoice Message Assembly includes an Association Business Information Entity and two Basic Business Information Entities. Additionally, in a definition of a complex type cardinality information of the included elements is also given with minOccurs and maxOccurs attributes. The used element namespaces such as cac, cbc are used according to UBL convention; they represent common an aggregate component and a common basic component, respectively. The location of these namespaces addresses the content of the second downloaded directory.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema
        targetNamespace="urn:oasis:names:specification:ubl:schema:xsd:Cross_Invoice"
        elementFormDefault="qualified" attributeFormDefault="unqualified" version="2.0"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:cac="urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2"
        xmlns="urn:oasis:names:specification:ubl:schema:xsd:Cross_Invoice"
        xmlns:cbc="urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2"
        xmlns:udt="urn:un:unece:uncefact:data:specification:UnqualifiedDataTypesSchemaModule:2"
        xmlns:ccts="urn:un:unece:uncefact:documentation:2"
        xmlns:ext="urn:oasis:names:specification:ubl:schema:xsd:CommonExtensionComponents-2"
        xmlns:qdt="urn:oasis:names:specification:ubl:schema:xsd:QualifiedDatatypes-2">
        <xsd:import
                namespace="urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2"
                schemaLocation="../common/UBL-CommonAggregateComponents-2.0.xsd"/>
        <xsd:import
                namespace="urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2"
                schemaLocation="../common/UBL-CommonBasicComponents-2.0.xsd"/>
        <xsd:import
                namespace="urn:un:unece:uncefact:data:specification:UnqualifiedDataTypesSchemaModule:2"
                schemaLocation="../common/UnqualifiedDataTypeSchemaModule-2.0.xsd"/>
        <xsd:import
                namespace="urn:oasis:names:specification:ubl:schema:xsd:QualifiedDatatypes-2"
                schemaLocation="../common/UBL-QualifiedDatatypes-2.0.xsd"/>
        <xsd:element name="CrossInvoice" type="CrossInvoiceType"/>
        <xsd:complexType name="CrossInvoiceType">
                <xsd:sequence>
                        <xsd:element ref="cac:IssuerInvoiceIssuerParty" minOccurs="0" maxOccurs="1"/>
                        <xsd:element ref="cbc:IdentificationIdentifier" minOccurs="1" maxOccurs="1"/>
                        <xsd:element ref="cbc:ContainmentProcedureText" minOccurs="0" maxOccurs="1"/>
                </xsd:sequence>
        </xsd:complexType>
</xsd:schema>
```

Figure 3.37: XSD of Message Assembly

The second directory, common, includes XSD files of Common Aggregate Components, Common Basic Components, Unqualified Data Types, Qualified Data types and Code Lists. Similar to Business Message Type, Aggregate Business Information Entities are also represented in XSD files. In Figure 3.38, a schema definition of Aggregate Business Information Entity is shown.

```
<xsd:element name="DefinedTradeContact" type="TradeContactType"/>
<xsd:complexType name="TradeContactType">
  <xsd:sequence>
    <xsd:element ref="DirectTelephoneSpecifiedCommunication" minOccurs="0" maxOccurs="1"/>
    <xsd:element ref="TelephoneSpecifiedCommunication" minOccurs="0" maxOccurs="1"/>
    <xsd:element ref="TelexSpecifiedCommunication" minOccurs="0" maxOccurs="1"/>
    <xsd:element ref="FaxSpecifiedCommunication" minOccurs="0" maxOccurs="1"/>
    <xsd:element ref="MobileTelephoneSpecifiedCommunication" minOccurs="0" maxOccurs="1"/>
    <xsd:element ref="EmailURISpecifiedCommunication" minOccurs="0" maxOccurs="1"/>
    <xsd:element ref="cbc:DepartmentNameText" minOccurs="0" maxOccurs="1"/>
    <xsd:element ref="cbc:TypeCode" minOccurs="0" maxOccurs="1"/>
    <xsd:element ref="cbc:PersonNameText" minOccurs="0" maxOccurs="1"/>
    <xsd:element ref="cbc:IdentificationIdentifier" minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>
</xsd:complexType>
```

Figure 3.38: An excerpt from Common Aggregate Components

As shown in Figure 3.38 if the complex type refers another Common Aggregate Component defined in that file, it does not declare namespaces, but if the element refers to a Common Basic Component, then it uses cbc namespace, since their schema definition are represented

in a different file.

Figure 3.39 shows an excerpt from Common Basic Components schema file and defines a complex type which includes Text type which is an unqualified data type.

```
<xsd:complexType name="CompleteNumberTextType">
  <xsd:simpleContent>
    <xsd:extension base="udt:TextType"/>
  </xsd:simpleContent>
</xsd:complexType>
```

Figure 3.39: An excerpt from Common Basic Components

Figure 3.40 shows Code Type, unqualified data type schema. In XSD file, the attributes of the type are declared with their names, types and use. For example, in Figure 3.40, all of the attributes are defined as optional.

```
<xsd:complexType name="CodeType">
   <xsd:simpleContent>
      <xsd:extension base="xsd:normalizedString">
                              <xsd:attribute name="listID" type="xsd:normalizedString" use="optional"/>
                              <xsd:attribute name="listAgencyID" type="xsd:normalizedString" use="optional"/>
                              <xsd:attribute name="listAgencyName" type="xsd:string" use="optional"/>
                              <xsd:attribute name="listName" type="xsd:string" use="optional"/>
         <xsd:attribute name="listVersionID" type="xsd:normalizedString" use="optional"/>
         <xsd:attribute name="name" type="xsd:string" use="optional"/>
         <xsd:attribute name="languageID" type="xsd:language" use="optional"/>
         <xsd:attribute name="listURI" type="xsd:anyURI" use="optional"/>
         <xsd:attribute name="listSchemeURI" type="xsd:anyURI" use="optional"/>
      </xsd:extension>
   </xsd:simpleContent>
</xsd:complexType>
```

Figure 3.40: An excerpt from Unqualified Data Type

Furthermore, the tool generates spreadsheet model of the graphical model for enabling versatile document modeling. The document models generated within the tool can be exported to the outside of the tool and can be shared and worked on at the outside of the tool by using simple office software such as Microsoft Excel or on-line Google Doc application.

For this purpose, the conceptual model persisted to the repository is retrieved, and they are marshaled to spreadsheet files according to UBL spreadsheet style and design. Similar to XSD folder hierarchy, spreadsheet folder is available under mod file and it consists of two other folders: maindoc and common. Maindoc folder includes a spreadsheet model describing a conceptual model of the retrieved document model and common includes the models of

common library elements.

Finally tool presents genericode files for defining code lists aka. Enumerations or controlled vocabularies for Coded Values. In UBL 2.0, only three code lists are enumerated in the schemas: (1) The CurrencyCodeContentType for internationally standardized currency codes, (2) The BinaryObjectMimeCodeContentType for MIME encoding identifiers and (3) The UnitCodeContentType for unit codes. In fact, the other code lists used in UBL are not enumerated in the schema expressions. As shown in Figure 3.41 UBL has a special formatted XML file for encoding these controlled vocabularies. ColumnSet describes the meta-data and Rows defines codes and their names. iSURF eDoCreator also forms these genericode files.

```
<ColumnSet>
    <Column Id="code" Use="required">
        <Data Type="normalizedString"/>
        <ShortName>Code</ShortName>
    </Column>
    <Key Id="codeKey">
        <ColumnRef Ref="code"/>
        <ShortName>CodeKey</ShortName>
    </Key>
    <Column Id="name" Use="optional">
        <Data Type="string"/>
        <ShortName>Name</ShortName>
    </Column>
</ColumnSet>
<SimpleCodeList>
    <Row>
        <Value ColumnRef="code">
            <SimpleValue>AED</SimpleValue>
        </Value>
        <Value ColumnRef="name">
            <SimpleValue>Dirham</SimpleValue>
        </Value>
    </Row>
```

Figure 3.41: An excerpt from Genericode file

The documentation process is done at the server side since the Flex based User Interface is not able to handle file management or processing. The Guideline execution engine interacts with the persistence layer and forms the XSD files, spreadsheet files and genericode file by examining relations. After that, URLs of formed files are passed to the interface side to let it

retrieve the files by accessing their URLs.

3.2.6 Protecting Privacy of Users and Document Models

iSURF eDoCreator is a tool which let its users collaboratively work thanks to its web based implementation. An on-line web tool is shown as new ways to work collaboratively and promote your research on-line. In the iSURF eDoCreator the generated document building blocks become available to the other users when creator commits the component. However, there is an important issue in this sharing mechanism that needs to be addressed to preserve privacy, i.e. access rights. There are two questions to be answered:

- Who can view my documents?
- Which documents are I allowed to view?

All of these access rights are adjusted via grouping property. In the iSURF eDoCreator grouping option is provided for both users and documents building blocks.

Like to all web based application, iSURF eDoCreator requires singing up and signing in the environment in order to initiate the use of tool features. All users have a unique username, which identifies the users. Users can subscribe to the groups and create a new group by using the identification of their usernames. For creating a group a user needs to click on the ”Manage Groups” menu item and then select ”Create Group” option. After that the creation of group is completed by setting a group name. The admin of the group is set as the username of the creator.

Furthermore, to subscribe to a group user again needs to click on ”Manage Groups” menu item and then select ”Open Groups Panel” option. At this time a panel which shows all available groups is initialized as shown in Figure 3.42. In the group list, the text color of the group name identifies three different categories: light green indicates the groups that the user owns, dark green indicates the groups that the user is a member of and finally the red text indicates that the user is neither subscribed nor owned.

There are two types of groups in iSURF eDoCreator: Public and Private Group. The public groups are the groups which do not need any authorization from the group owner for joining.

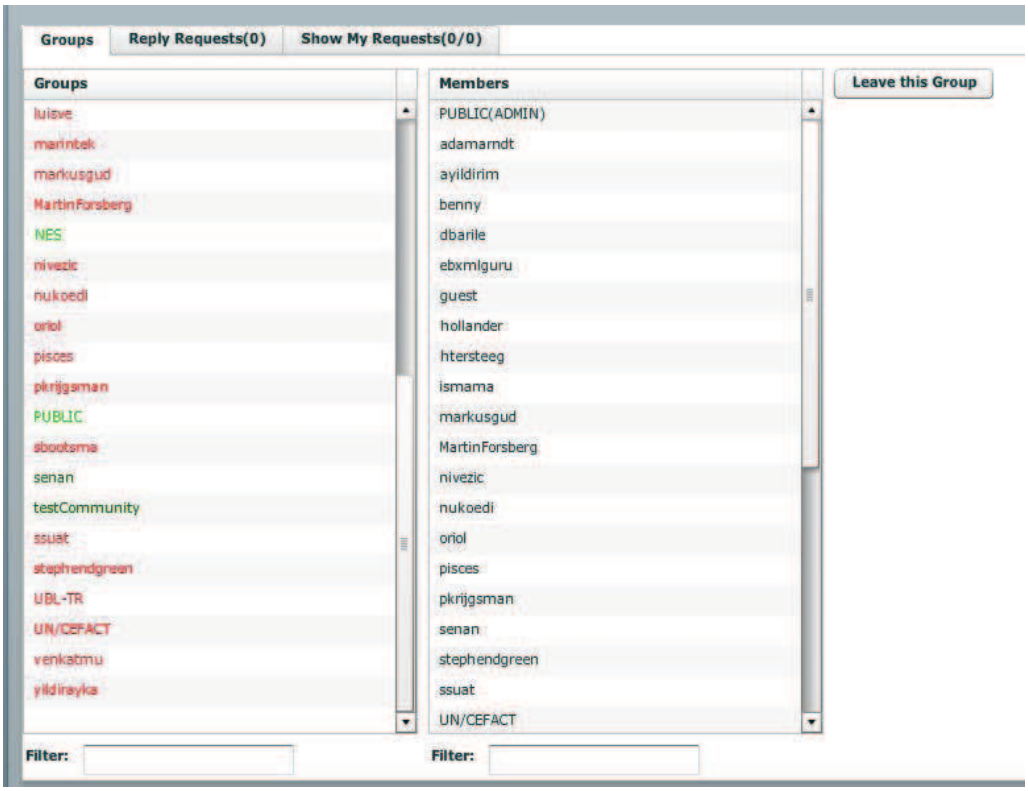

Figure 3.42: Managing Group Options

The works of standardization organizations are hold in public groups. Users join to the public groups just by clicking on ”Join this group” button without waiting for the approval of the group owner. Unlike to public groups, to enter to a private group user needs to click on ”Join this group” Button and write a request text message to the group owner since user needs the owners’ approval before joining a private group. Hence, this makes document building block only to be available to group members and preserve privacy.

To leave a group, users need to first select the group name from the list and click on ”Leave this group” button. This way, users can manage the initial list shown in Figure 3.42, which show all available document building blocks which are visible to the user.

Furthermore, every document building block has a group list which identifies the groups that are allowed to see the created document building block. Therefore, creators need to initialize the object’s group before committing it to the repository. In other words, users need to select groups which are able to visualize the selected document building block. User can only share the document with the groups which he/she is a member of.

Figure 3.43: Managing document building blocks' groups

As shown in Figure the group panel has three sections: In the left hand side the list of groups that user has subscribed is shown, in the middle the document building block details and selected group details are presented and at the right hand side the document building blocks' groups are shown. As users click on "Add" Button, the group selected on the left is added to the right panel. For sharing a document with a group, the owner of the document building block shall be the member of that group.

Finally as a miscellaneous feature of the iSURF eDoCreator, we can say that iSURF eDoCreator is a multilingual tool which let users change the interface language from a drop-down menu list. Currently the tool supports both Turkish and English, however it is implemented in an expandable manner and the language options can be increased whenever the profile file is translated to any other language.

CHAPTER 4

Use Cases: UBL-TR Customization and Gap Analysis between UBL-TR and PEPPOL Project

iSURF eDoCreator provides a document modeling environment for users to assemble their documents from common components according to UN/CEFACT CCTS document modeling methodology and customize these documents using UBL 2.0 customization methodology. When the document building blocks are derived from a common semantic specification with well-defined rules, it becomes possible to achieve electronic business document interoperability.

iSURF eDoCreator tool has been used to derive the UBL 2.0 conformant eInvoice, Turkey [15] from the standard UBL 2.0 eInvoice. Such customizations are becoming popular recently, especially within the scope of the large scale integration project, PEPPOL (Pan-European Public Procurement Online) currently being implemented in Europe. PEPPOL will be producing UBL 2.0 conformant invoice, order, virtual company dossier and catalog schemas to be customized to the Member States and we believe that the publicly accessible iSURF eDoCreator tool provides an opportunity to help with these customizations. Furthermore, the tool proved to be very useful in performing the gap analysis between NES/UBL Invoice [11] and eInvoice, Turkey [29]. In this section, the details of how iSURF eDoCreator is used in UBL Turkish Localization Subcommittee and how gap analysis between NES/UBL Invoice and eInvoice, Turkey is produced are given.

4.1 UBL-TR eInvoice Interoperability Profile

Turkey's eInvoice Interoperability Profile is being realized by the Revenue Administration (Gelir İdaresi Başkanlığı, GIB) [30]. The GIB eInvoice Interoperability Profile addresses all the layers in the interoperability stack, namely, the content of the eInvoice documents, the transport and communication interoperability and the business process layer interoperability. For the Document Content Layer, the UBL 2.0 Invoice documents are localized to Turkey according to the "conformant" customization guidelines of UBL. For this purpose iSURF eDoCreator is utilized and local documents are generated.

The "interoperability profiling" in the Information Technology means fixing the roles, the business processes and the interactions in a given use-case scenario and then determining the standards to be used at each layer of the interoperability stack, possibly by further restricting them. The interoperability stack involves the document content layer, the transport and the communication layer and the business process layer. The basic e-business requirement of authentication, confidentiality, integrity and non-repudiation must also be handled by the profile.

In this section, we mainly focus on document content of the eInvoice Interoperability Profile of the Revenue Administration (Gelir İdaresi Başkanlığı, GIB) of Turkey. GIB [30] is in charge of revenue management in Turkey including implementing the state revenue policy; ensuring to collect the governmental claims; measuring the costs of all exceptions, exemption and discounts in the tax laws or other fiscal laws and carrying out tax inspection and audit at the direction of main policies and strategies determined by the Ministry of Finance.

In order increase its efficiency and effectiveness towards accomplishing its tasks, the Revenue Administration has realized an eInvoice Interoperability Profile. In this first phase of the implementation, only the invoicing process is considered and the other procurement processes such as ordering and payment are left as future work. Hence the matching of the Invoice to other electronic documents like order is not considered.

4.1.1 The GIB eInvoice Interoperability Profile: Document Content Layer

For the GIB eInvoice Interoperability Profile Document Content Layer, the UBL 2.0 Invoice documents are localized to Turkey through the work of OASIS UBL Turkish Localization Subcommittee [15] according to the ”conformant” customization guidelines of UBL.

The UBL documents can be customized to specific needs in two ways as described in Section 2.2.1. The ”conformant” customization which is used in Turkey, allows the XML instances in the customized implementation to also conform to the original standard UBL 2.0 schemas hence providing interoperability with other conformant schemas. The UBL Invoice document model in spreadsheet can be shown in Figure 4.1.

UBL Name	Dictionary Entry Name	Object Class	Property Term	Property Term Possessive	Property Term Primary Noun	Property Term	Representation Term
Invoice	Invoice. Details	Invoice					
UBLVersionID	Invoice. UBL Version Identifier. Identifier	Invoice		UBL Version	Identifier	UBL Version Identifier	Identifier
CustomizationID	Invoice. Customization Identifier. Identifier	Invoice		Customization	Identifier	Customization Identifier	Identifier
ProfileID	Invoice. Profile Identifier. Identifier	Invoice		Profile	Identifier	Profile Identifier	Identifier
ID	Invoice. Identifier	Invoice			Identifier	Identifier	Identifier
CopyIndicator	Invoice. Copy_ Indicator. Indicator	Invoice	Copy		Indicator	Indicator	Indicator
UUID	Invoice. UUID. Identifier	Invoice			UUID	UUID	Identifier
IssueDate	Invoice. Issue Date. Date	Invoice		Issue	Date	Issue Date	Date
IssueTime	Invoice. Issue Time. Time	Invoice		Issue	Time	Issue Time	Time
InvoiceTypeCode	Invoice. Invoice Type Code. Code	Invoice		Invoice Type	Code	Invoice Type Code	Code
Note	Invoice. Note. Text	Invoice			Note	Note	Text
TaxPointDate	Invoice. Tax Point Date. Date	Invoice		Tax Point	Date	Tax Point Date	Date
DocumentCurrencyCode	Invoice. Document_ Currency Code. Code	Invoice	Document	Currency	Code	Currency Code	Code
TaxCurrencyCode	Invoice. Tax_ Currency Code. Code	Invoice	Tax	Currency	Code	Currency Code	Code
PricingCurrencyCode	Invoice. Pricing_ Currency Code. Code	Invoice	Pricing	Currency	Code	Currency Code	Code
PaymentCurrencyCode	Invoice. Payment_ Currency Code. Code	Invoice	Payment	Currency	Code	Currency Code	Code
PaymentAlternativeCurrency	Invoice. Payment Alternative_ Currency Code. Code	Invoice	Payment	Currency	Code	Currency Code	Code
AccountingCostCode	Invoice. Accounting Cost Code. Code	Invoice		Accounting Cost	Code	Accounting Cost Code	Code
AccountingCost	Invoice. Accounting Cost. Text	Invoice			Accounting Cost	Accounting Cost	Text
LineCountNumeric	Invoice. Line Count. Numeric	Invoice		Line	Count	Line Count	Numeric
InvoicePeriod	Invoice. Invoice_ Period. Period	Invoice	Invoice			Period	Period
OrderReference	Invoice. Order Reference	Invoice				Order Reference	Order Reference
BillingReference	Invoice. Billing Reference	Invoice				Billing Reference	Billing Reference
DespatchDocumentReferenc	Invoice. Despatch_ Document Reference. Document Reference	Invoice	Despatch			Document Reference	Document Reference
ReceiptDocumentReference	Invoice. Receipt_ Document Reference. Document Reference	Invoice	Receipt			Document Reference	Document Reference
OriginatorDocumentReferenc	Invoice. Originator_ Document Reference. Document Reference	Invoice	Originator			Document Reference	Document Reference
ContractDocumentReference	Invoice. Contract_ Document Reference. Document Reference	Invoice	Contract			Document Reference	Document Reference
AdditionalDocumentReferen	Invoice. Additional_ Document Reference. Document Reference	Invoice	Additional			Document Reference	Document Reference
Signature	Invoice. Signature	Invoice				Signature	Signature
AccountingSupplierParty	Invoice. Accounting_ Supplier Party. Supplier Party	Invoice	Accounting			Supplier Party	Supplier Party
AccountingCustomerParty	Invoice. Accounting_ Customer Party. Customer Party	Invoice	Accounting			Customer Party	Customer Party
PayeeParty	Invoice. Payee_ Party. Party	Invoice	Payee			Party	Party
BuyerCustomerParty	Invoice. Buyer_ Customer Party. Customer Party	Invoice	Buyer			Customer Party	Customer Party
SellerSupplierParty	Invoice. Seller_ Supplier Party. Supplier Party	Invoice	Seller			Supplier Party	Supplier Party
TaxRepresentativeParty	Invoice. Tax Representative_ Party. Party	Invoice	Tax			Party	Party
Delivery	Invoice. Delivery	Invoice				Delivery	Delivery
DeliveryTerms	Invoice. Delivery Terms	Invoice				Delivery Terms	Delivery Terms
PaymentMeans	Invoice. Payment Means	Invoice				Payment Means	Payment Means
PaymentTerms	Invoice. Payment Terms	Invoice				Payment Terms	Payment Terms
PrepaidPayment	Invoice. Prepaid_ Payment. Payment	Invoice	Prepaid			Payment	Payment
AllowanceCharge	Invoice. Allowance Charge	Invoice				Allowance Charge	Allowance Charge
TaxExchangeRate	Invoice. Tax_ Exchange Rate. Exchange Rate	Invoice	Tax			Exchange Rate	Exchange Rate
PricingExchangeRate	Invoice. Pricing_ Exchange Rate. Exchange Rate	Invoice	Pricing			Exchange Rate	Exchange Rate
PaymentExchangeRate	Invoice. Payment_ Exchange Rate. Exchange Rate	Invoice	Payment			Exchange Rate	Exchange Rate
PaymentAlternativeExchange	Invoice. Payment Alternative_ Exchange Rate. Exchange Rate	Invoice	Payment			Exchange Rate	Exchange Rate
TaxTotal	Invoice. Tax Total	Invoice				Tax Total	Tax Total
LegalMonetaryTotal	Invoice. Legal_ Monetary Total. Monetary Total	Invoice	Legal			Monetary Total	Monetary Total
InvoiceLine	Invoice. Invoice Line	Invoice				Invoice Line	Invoice Line

Figure 4.1: UBL Invoice Document Model

The following changes are made to UBL 2.0 eInvoice documents while localizing them to the GIB Profile. The requirement of business changes among counties, industries and business processes. The UBL schemas allow the communities to extend the document schemas by adding new elements, which are not described in the abstract business model of UBL. These kinds of customized artifacts are positioned under a reserved element named UBLExtensions found at the beginning of all UBL documents. Indeed, the ”UBLExtensions” element allows

for conformant customizations by restricting the use of non-UBL elements inside these tags.

The optional ”UBLExtensions” element is used to include non-UBL data elements specific to the intended use in Turkey. For example, the XSL files that are used to visualize the Invoice documents are embedded in the ”UBLExtensions” element.

Furthermore, the optional information entities in the original UBL 2.0 Invoice documents that are not necessary for the GIB Profile are removed. For example, ”TaxPointDate” entity is removed from the Invoice document, since in Turkey the ”IssueDate” is used to indicate the point at which the tax becomes applicable. Clearly, removing the optional elements does not violate the conformance to the original UBL schema. This is done through check tree option of iSURF eDoCreator.

There are additional constraints on the value space of the information entities in the GIB Profile. For example, a constraint is introduced to check whether the sum of ”TaxAmount” items of the ”TaxSubtotal” elements in a ”TaxTotal” entity is equal to the ”TaxAmount” item of the respective ”TaxTotal” entity. Such requirements are reflected in the UBL schemas through Schematron rules. iSURF eDoCreator has a document special Schematron Editor, which produces Schematron files from graphically generated assertions. The details of Schematron editor is described in Section 3.2.3.4.

Finally, the customization of the code lists is realized. The code lists are used to convey the meaning of the values in the data elements. In UBL 2.0, only three code lists are enumerated in the schemas: (1) The CurrencyCodeContentType for internationally standardized currency codes, (2) The BinaryObjectMimeCodeContentType for MIME encoding identifiers and (3) The UnitCodeContentType for unit codes. The other code lists used in UBL are not enumerated in the schema expressions. Instead, UBL uses a common base type called CodeType, which is an extension of ”xsd:normalizedString” for all elements expressing values from the code lists. This constraint is also added to documents through Schematron Editor as it lets the users to upload Code Type enumerations either through a text based input field or a genericode file.

The resulting document tree is in Figure 4.3.

After these modifications, the documentation of the UBL-TR Invoice document is retrieved from the iSURF eDoCreator. The UBL 2.0 package includes files for every code list. These

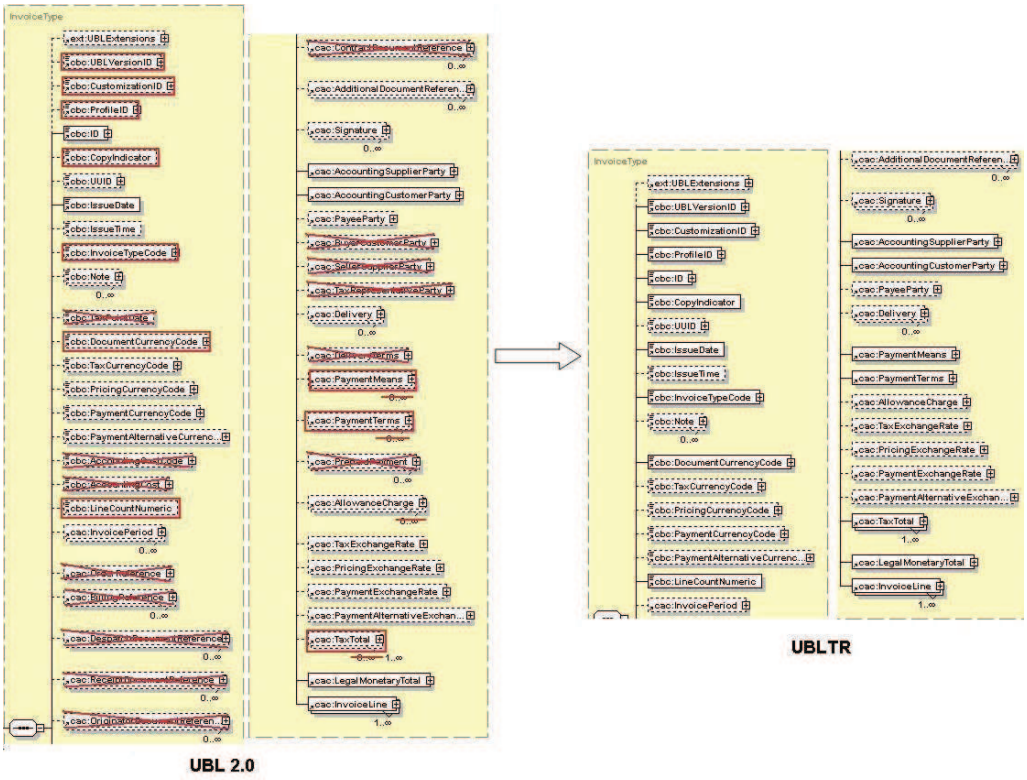

Figure 4.2: Customizing UBL 2.0 Invoice to UBLTR Invoice

files are separate from the provided XSD schemas and they are in a standard format. For the GIB Profile, these files are generated for the codes used in Turkey. For example, a value set for ”TaxTypeCode” basic business information entity is created. Some example values for this value set are: Income Tax, Value Added Tax (VAT), and Stamp Tax.

Validation of the eInvoice, Turkey: UBL 2.0 recommends a two-phase validation technique since the specification of the default values directly in the schemas makes it difficult to modify the code lists to meet customization requirements. In the GIB implementation, the two-phase validation technique is used: in the first phase, an incoming invoice document is validated against UBL 2.0 GIB eInvoice XSD schemas.

If the instance passes the first phase, in the second phase it is checked against the rules, which specify GIB business constraints on the values of the elements in the instance. These rules are specified through Schematron language. If the instance passes both phases successfully, it is delivered to the processing business application.

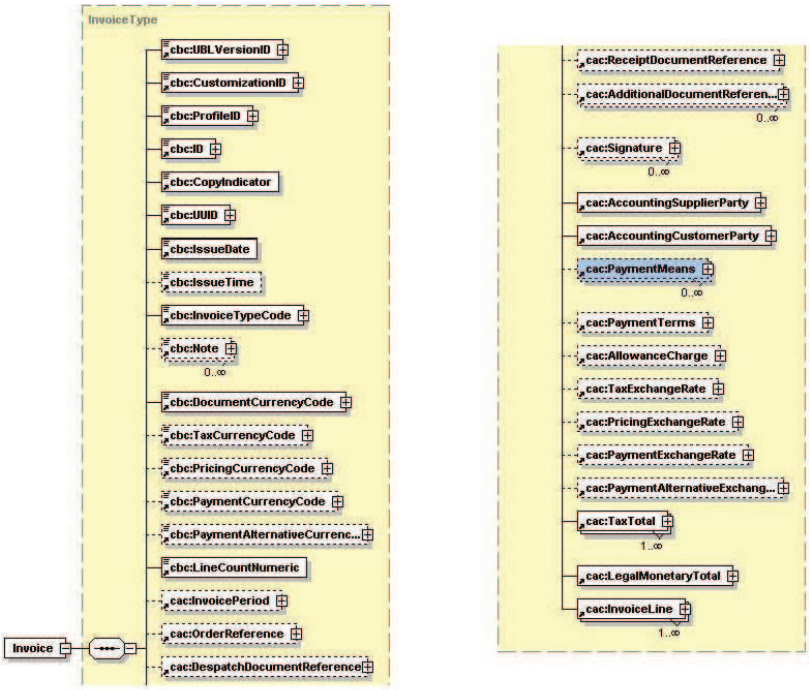

Figure 4.3: Visualization of UBL TR Invoice Document Schema

4.2 Pan-European Public Procurement Online - PEPPOL Project

UBL is being adopted by several communities around the world, especially in electronic government applications. The first government to use UBL Invoice is Denmark. The use of UBL Invoice is realized through the ”Offentlig Information Online UBL (OIOUBL) [8]” Project and has been mandated by law for all public-sector businesses in Denmark. Furthermore, in Sweden, the National Financial Management Authority recommended UBL Invoice customized to Sweden, namely, Svefaktura [9] for all government use. Following the success of Danish and Swedish examples, representatives from Denmark, Norway, Sweden, UK, Finland and Iceland have created a Northern European Subset (NES) [11] for UBL to ensure interoperability among these countries. Furthermore, Revenue Administration of Turkey chose UBL 2.0 as the electronic document standard to be used in the Turkish National eInvoicing system and generated Turkish UBL 2.0 customization (UBLTR) [15]. These kinds of standardized electronic business processes have already realized the high impact of collaborations, generating savings on administrative and transaction costs by eliminating invoices and orders by fax or email, the reduction of data entered manually and the time businesses spend queuing,

filling out forms and sorting out paper work.

The large scale integration project, PEPPOL (Pan-European Public Procurement Online) will be producing UBL 2.0 conformant invoice, order, virtual company dossier and catalog schemas to be customized to the Member States. The objective of the PEPPOL-project is to set up an integrated pilot solution across borders, capitalizing each country's strength/advantage in existing national systems that jointly facilitate the enabling of an EU-wide interoperable solution for public eProcurement.

PEPPOL aims to demonstrate an integrated operational solution that builds upon national systems and provides cross border access to public eProcurement supporting the full cycle of e-procurement activities. In other words, any economic operator in the EU and the EEA can respond to and enact upon any published public tender throughout the EU and the EEA community from their own national infrastructure to another national infrastructure throughout the framework provided by PEPPOL-project.

Since this pilot project focuses on cross-border activities within the procurement process, UBL TR Localization will also participate into the initiative to follow the work and influence the definition of specifications as they will be developed. By making sure that their systems will work together later, as in inline with PEPPOL objective, economic operators in Turkey can attend cross-border published public tenders and can win public sector contracts anywhere in the EU. This is a crucial step towards completion of the Single European Market.

The project will not replace but rather build on existing national e-procurement systems using information and communication technologies to enable them to communicate with each other. However, in any case this requires national solutions from both suppliers and governments need to be aligned with common European standards, which will require some investment.

For this requirement, a new tool is developed to perform the gap analysis between different customizations of UBL to show the interoperability level of the compared document models.

4.3 A New feature: Gap Analysis Reporting

Although UBL Conformant schemas are valid for UBL standard schema, there is still to do to interoperate two systems adopting different UBL Conformant message definitions. For ex-

ample, although both NES and UBLTR are conformant subsets of UBL 2.0 standard, there are some incompatibilities between them. For example, ”/Invoice/LineCountNumeric” is required as a mandatory element in UBLTR, whereas this element is excluded from NES. Therefore, first of all, it is needed to find how problems can arise while mapping two exchanged messages among the system in order to handle these problems. Therefore, first of all, requirements of the gap analysis tool are elicited and some problem levels are identified according to UBL Conformant Customization Guidelines.

4.3.1 Interoperability Problem Levels

There are four types of issues that may cause interoperability problems between two conformant customization of UBL from document content perspective according to cardinality values of Basic Business Information Entity and Aggregate Business Information Entity. Cardinality of Basic Business Information Entity indicates whether the Basic Business Information Entity Property represents a not-applicable, optional, mandatory and/or repetitive characteristic of the Aggregate Business Information Entity, whereas Cardinality of Aggregate Business Information Entity indicates whether the Association Business Information Entity Property represents a not-applicable, optional, mandatory and/or repetitive characteristic of the Aggregate Business Information Entity. The identified problems are as follows:

- Multiple Cardinality versus Optional Cardinality - Issue 1:
  - There are elements with incompatible cardinalities. For example, a ”0..1” cardinality in the ”/Invoice/Note” element in UBLTR is set as ”0..n” in NES.
- Optional Cardinality versus Excluded Element - Issue 2:
  - An element is set as optional in one of them, whereas the same element is excluded from the other. . For example, ”/Invoice/TaxPointDate” element is excluded from UBLTR, but it is optional in NES.
- Optional Cardinality versus Mandatory Cardinality - Issue 3:
  - An element is set as mandatory in one of them, whereas the same element is set as optional in another. For example, ”/Invoice/CopyIndicator” element is mandatory in UBLTR, but it is optional in NES.

- Mandatory Cardinality versus Excluded Element - Issue 4:
  - An element is set as mandatory in one of them, whereas the same element is excluded from the other. For example, ”/Invoice/LineCountNumeric” element is mandatory in UBLTR; however, it is excluded from NES.

Considering the severity, the issue 1 is the least important and issue 4 is the most important issue which may cause interoperability problems in the message exchange.

4.3.2 Document Content Gap Analysis Tool with Implementation Details

An additional feature of iSURF eDoCreator tool is the Gap Analysis Reporting Tool, which compares two customized e-business documents from document content perspective. This tool automates the manual comparison of all document building blocks to identify the elicited problem levels. Gap Analysis Tool is available under ”Tools” menu in iSURF eDoCreator main panel.

As shown in Figure 4.4 in the first panel of the tool, it visualizes all available e-business document schemas in two separate lists in order to let the user select two documents to be compared. In the list, e-business document schemas are identified via their Dictionary Entry Names, Standards on which they are based on and Customization identifiers. Users are requested to select a document schema from each list and click to ”View” Button to initiate a gap analysis process. There is one restriction in the selection; the selected documents should be based on the same standard. In other words, user cannot compare two different documents standards based on UN/CEFACT CCTS methodology such as GS1 and UBL.

The tool graphically presents two selected document schema details in a tree view in a separated panel as shown in Figure 4.5. The document models are presented in Russian Doll view and the encapsulation details of each document building block is available, when nodes are expanded. Through this expandable hierarchical tree view of document schemas, users are enabled to see the whole data content of a component at a glance by opening nodes of the tree. Furthermore, the cardinality values of selected document building block are presented at the bottom of each panel for let users to manually identify differences.

When user click on ”Compare”, the tool navigates over first level document elements and

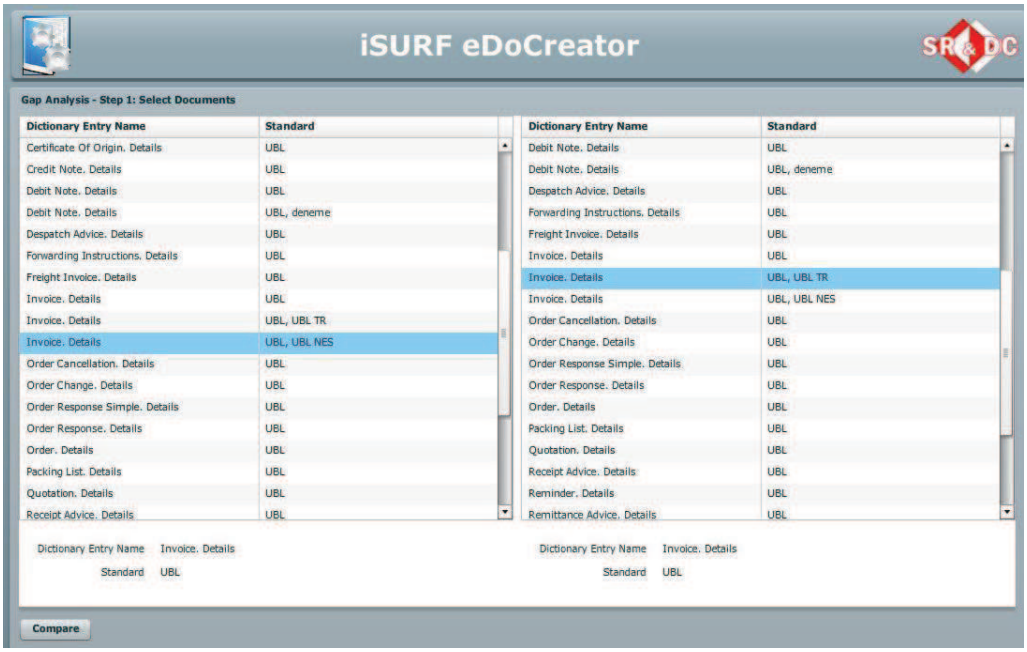

Figure 4.4: Gap Analysis Tool- Document Selection

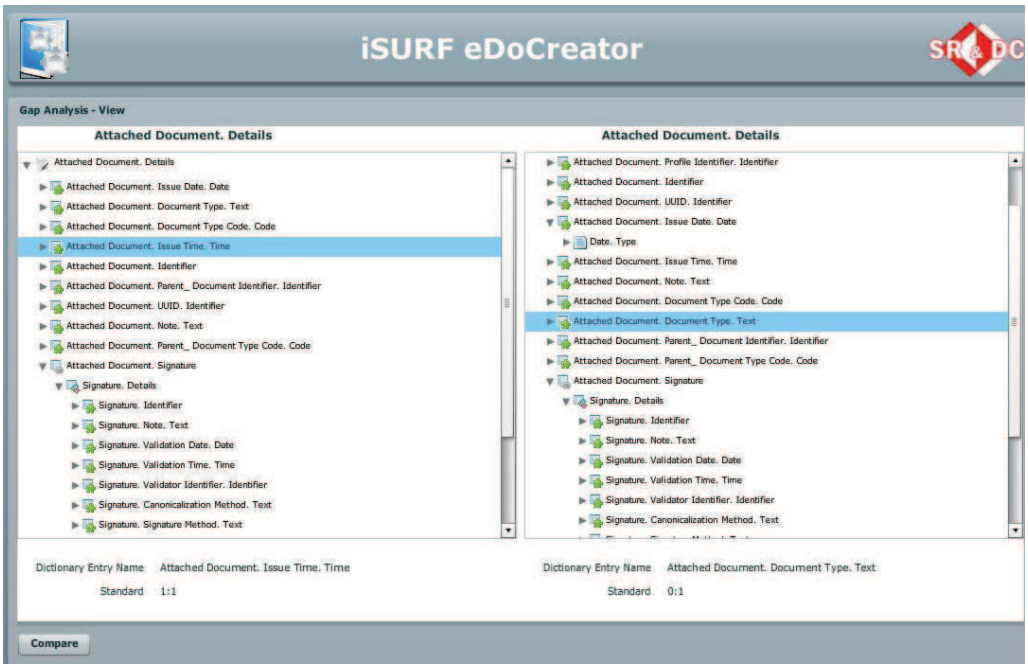

Figure 4.5: Gap Analysis View

identifies differences between two e-business document schemas based on their cardinality values. As mentioned previously, there are four identified problem levels for cardinality val-

ues, which may cause interoperability problems among document schemas. All four issue levels are indicated through colors in the Document Level Compare List as presented at the Figure 4.6.

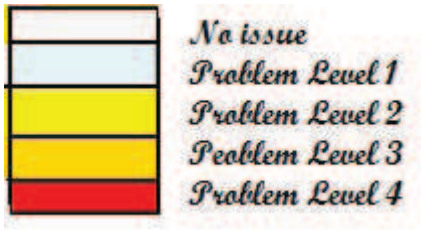

Figure 4.6: Interoperability Problem Levels- Color List

When comparing an Aggregate Business Information Entity, first of all Basic Business Information Entity Properties and Association Business Information Entity Properties are retrieved from the first selected document, then it checks for each document building block representing these properties. For each Basic Business Information Entity and Association Business Information Entity it navigates over the second document model in order to find the same entity with the same Dictionary Entry Name. If it founds it, it compares their cardinality values and assigns their problem levels. Otherwise, it says that the document building block is excluded from the second document model. After that it compares the unvisited second document building blocks in the same manner. Finally, the row colors are automatically assigned by the visualization table according to its problem level.

In addition to possible issues indicated by row colors, the grid also presents Dictionary Entry Names of document building blocks and their cardinality values.

Furthermore, in a similar approach, the tool can visualize component level issues (in Figure 4.8), which go to one level deeper and compare the properties of each Aggregate Business Information Entity. Furthermore, it can compare all document Aggregate Business Information Entities -whatever their depths are- with the second model elements and produce the gap analysis report. For this purpose, it navigates over the document model tree in a breadth-first manner and forms a spreadsheet file with the same format with the shown graphical user interfaces and problem level color codes.

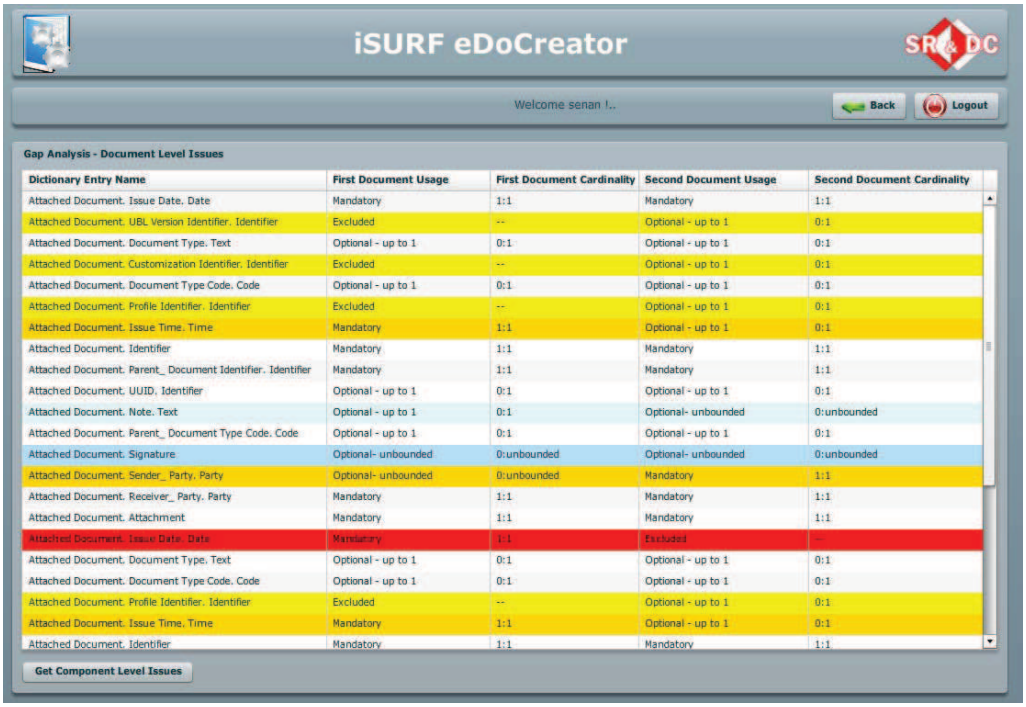

Figure 4.7: Gap Analysis Results at the Document Level

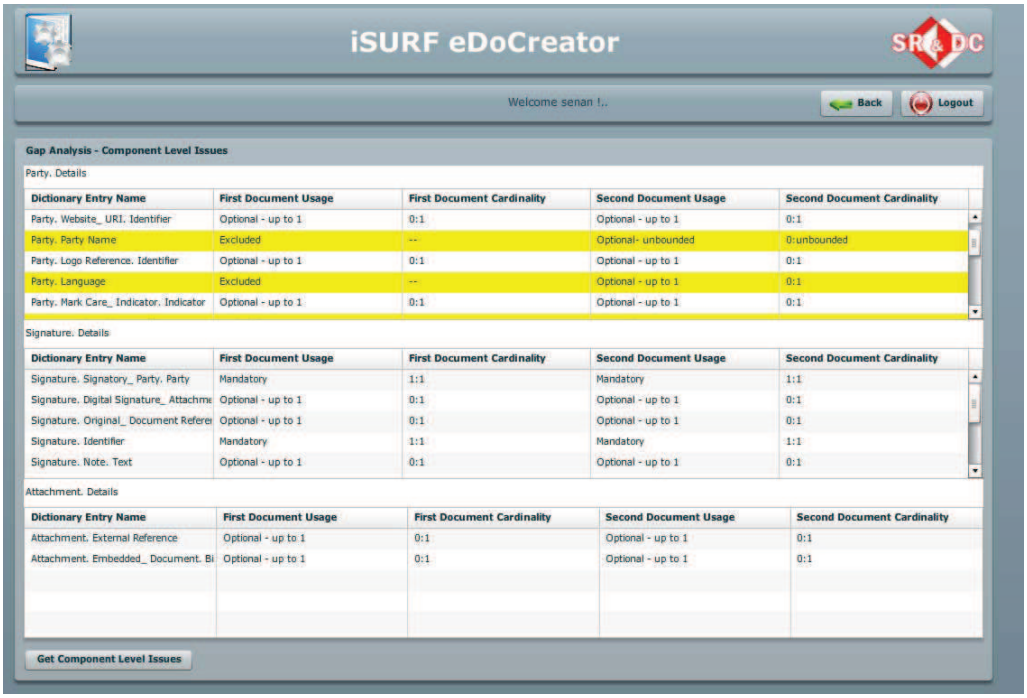

Figure 4.8: Gap Analysis Results at Component Level

4.3.3 Gap Analysis between Northern European Subset (NES) Invoice and Turkish UBL eInvoice Customization (UBLTR)

We prepared a report for the PEPPOL Project WP5 to present the results of the gap analysis between Northern European Subset (NES) Invoice and Turkish UBL eInvoice Customization (UBLTR) [15]. This document presents the gap analysis between two conformant customization of UBL 2.0, namely, (1) Northern European Subset (NES) and (2) the Turkish eInvoice customization (UBLTR). The analysis is performed from two perspectives: The Business Process and the Document Content. Since this thesis focuses on the document content perspective, in this section, the details of document content compression is presented.

Considering the severity, the issue 1 is the least important and issue 4 is the most important. In the following tables, the elements that may cause interoperability problems are presented. In the tables, the following font styles are used for the interoperability issues defined: Issue 1-Normal, issue 2-Italic, issue 3-Bold, issue 4-Underline. In Figure 4.9, the gaps are indicated in the Invoice document. In Figure 4.10, the gaps in the common components are presented using the font styles.

Invoice	UBLTR-Usage	UBLTR-Cardinality	NES-Usage	NES-Cardinality
CopyIndicator	**USED**	**1**	**USED**	**0..1**
UUID	*USED*	*0..1*	*EXCLUDED*	
IssueTime	*USED*	*0..1*	*EXCLUDED*	
Note	USED	0..n	USED	0..1
TaxPointDate	*EXCLUDED*		*USED*	*0..1*
PricingCurrencyCode	*USED*	*0..1*	*EXCLUDED*	
PaymentCurrencyCode	*USED*	*0..1*	*EXCLUDED*	
PaymentAlternativeCurrencyCode	*USED*	*0..1*	*EXCLUDED*	
AccountingCost	*EXCLUDED*		*USED*	*0..1*
LineCountNumeric	**USED**	**1**	**EXCLUDED**	
OrderReference	*EXCLUDED*		*USED*	*0..1*
BillingReference	*EXCLUDED*		*USED*	*0..1*
Delivery	USED	0..n	USED	0..1
DeliveryTerms	*EXCLUDED*		*USED*	*0..1*
PaymentMeans	**USED**	**1**	**USED**	**0..n**
PaymentTerms	**USED**	**1**	**USED**	**0..1**
AllowanceCharge	USED	0..1	USED	0..n
PricingExchangeRate	*USED*	*0..1*	*EXCLUDED*	
PaymentExchangeRate	*USED*	*0..1*	*EXCLUDED*	
PaymentAlternativeExchangeRate	*USED*	*0..1*	*EXCLUDED*	

Figure 4.9: Invoice Document Level Issues

Address	UBLTR-Usage	UBLTR-Cardinality	NES-Usage	NES-Cardinality
ID	*EXCLUDED*		*USED*	*0..1*
AddressFormatCode	*EXCLUDED*		*USED*	*0..1*
Postbox	*EXCLUDED*		*USED*	*0..1*
StreetName	*EXCLUDED*		*USED*	*0..1*
AdditionalStreetName	*EXCLUDED*		*USED*	*0..1*
BuildingName	*EXCLUDED*		*USED*	*0..1*
BuildingNumber	*EXCLUDED*		*USED*	*0..1*
Department	*EXCLUDED*		*USED*	*0..1*
CityName	*EXCLUDED*		*USED*	*0..1*
PostalZone	*EXCLUDED*		*USED*	*0..1*
Region	*EXCLUDED*		*USED*	*0..1*
AddressLine	**USED**	**1**	**USED**	**0..n**
Country	**USED**	**1**	**USED**	**0..1**
AllowanceCharge	USED		USED	
ChargeIndicator	USED	1	USED	1
AllowanceChargeReasonCode	*EXCLUDED*		*USED*	*0..1*
TaxCategory	*EXCLUDED*		*USED*	*0..1*
Attachment	USED		USED	
EmbeddedDocumentBinaryObject	**USED**	**1**	**USED**	**0..1**
ExternalReference	*EXCLUDED*		*USED*	*0..1*
BillingReference	EXCLUDED		USED	
InvoiceDocumentReference	*EXCLUDED*		*USED*	*0..1*
CreditNoteDocumentReference	*EXCLUDED*		*USED*	*0..1*
BillingReferenceLine	*EXCLUDED*		*USED*	*0..1*
BillingReferenceLine	EXCLUDED		USED	

ID	**EXCLUDED**		**USED**	**1**
Branch	EXCLUDED		USED	
ID	*EXCLUDED*		*USED*	*0..1*
FinancialInstitution	*EXCLUDED*		*USED*	*0..1*
CommodityClassification	USED		USED	
CommodityCode	*EXCLUDED*		*USED*	*0..1*
ItemClassificationCode	**USED**	**1**	**USED**	**0..1**
Communication	USED		EXCLUDED	
ChannelCode	**USED**	**1**	**EXCLUDED**	
Channel	*USED*	*0..1*	*EXCLUDED*	
Value	**USED**	**1**	**EXCLUDED**	
Contact	USED		USED	
ID	*EXCLUDED*		*USED*	*0..1*
Name	*EXCLUDED*		*USED*	*0..1*
OtherCommunication	*USED*	*0..n*	*EXCLUDED*	
Country	USED		USED	
IdentificationCode	**USED**	**0..1**	**USED**	**1**
Name	**USED**	**1**	**EXCLUDED**	
CreditAccount	EXCLUDED		USED	
AccountID	**EXCLUDED**		**USED**	**1**
Delivery	USED		USED	
ActualDeliveryDate	*EXCLUDED*		*USED*	*0..1*
DeliveryLocation	*EXCLUDED*		*USED*	*0..1*
Despatch	**USED**	**1**	**EXCLUDED**	
DeliveryTerms	EXCLUDED		USED	
ID	*EXCLUDED*		*USED*	*0..1*
SpecialTerms	*EXCLUDED*		*USED*	*0..1*

Despatch	USED		EXCLUDED	
ID	**USED**	**1**	**EXCLUDED**	
ActualDespatchDate	**USED**	**1**	**EXCLUDED**	
DocumentReference	USED		EXCLUDED	
IssueDate	*EXCLUDED*		*USED*	*0..1*
DocumentTypeCode	*USED*	*0..1*	*EXCLUDED*	
ExchangeRate	USED		USED	
SourceCurrencyBaseRate	*EXCLUDED*		*USED*	*0..1*
TargetCurrencyBaseRate	*EXCLUDED*		*USED*	*0..1*
CalculationRate	**USED**	**1**	**USED**	**0..1**
ExternalReference	EXCLUDED		USED	
URI	**EXCLUDED**		**USED**	**1**
FinancialAccount	USED		USED	
ID	**USED**	**1**	**USED**	**0..1**
AccountTypeCode	*EXCLUDED*		*USED*	*0..1*
FinancialInstitutionBranch	*EXCLUDED*		*USED*	*0..1*
FinancialInstitution	EXCLUDED		USED	
ID	**EXCLUDED**		**USED**	**1**
Name	*EXCLUDED*		*USED*	*0..1*
InvoiceLine	USED		USED	
InvoicedQuantity	**USED**	**1**	**USED**	**0..1**
AccountingCost	*EXCLUDED*		*USED*	*0..1*
OrderLineReference	*EXCLUDED*		*USED*	*0..1*
Delivery	*EXCLUDED*		*USED*	*0..1*
AllowanceCharge	USED	0..1	USED	0..n
TaxTotal	USED	0..1	USED	0..n
Price	**USED**	**1**	**USED**	**0..1**

Item	USED		USED	
BrandName	*USED*	*0..1*	*EXCLUDED*	
ModelName	*USED*	*0..1*	*EXCLUDED*	
ManufacturersItemIdentification	*USED*	*0..1*	*EXCLUDED*	
StandardItemIdentification	*EXCLUDED*		*USED*	*0..1*
OriginCountry	*EXCLUDED*		*USED*	*0..1*
ClassifiedTaxCategory	*EXCLUDED*		*USED*	*0..n*
AdditionalItemProperty	*EXCLUDED*		*USED*	*0..n*
ItemInstance	*EXCLUDED*		*USED*	*0..n*
ItemInstance	EXCLUDED		USED	
ProductTraceID	*EXCLUDED*		*USED*	*0..1*
ManufactureDate	*EXCLUDED*		*USED*	*0..1*
ManufactureTime	*EXCLUDED*		*USED*	*0..1*
RegistrationID	*EXCLUDED*		*USED*	*0..1*
SerialID	*EXCLUDED*		*USED*	*0..1*
LotIdentification	*EXCLUDED*		*USED*	*0..1*
ItemProperty	EXCLUDED		USED	
Name	**EXCLUDED**	**.**	**USED**	**1**
Value	**EXCLUDED**	**.**	**USED**	**1**
UsabilityPeriod	*EXCLUDED*		*USED*	*0..1*
ItemPropertyGroup	*EXCLUDED*		*USED*	*0..n*
ItemPropertyGroup	EXCLUDED		USED	
ID	**EXCLUDED**	**.**	**USED**	**1**
Name	*EXCLUDED*		*USED*	*0..1*
Location	EXCLUDED		USED	
Address	*EXCLUDED*		*USED*	*0..1*
LotIdentification	EXCLUDED		USED	
LotNumberID	*EXCLUDED*		*USED*	*0..1*
ExpiryDate	*EXCLUDED*		*USED*	*0..1*

MonetaryTotal	USED		USED	
TaxExclusiveAmount	**USED**	**1**	**USED**	**0..1**
TaxInclusiveAmount	**USED**	**1**	**USED**	**0..1**
PayableRoundingAmount	**USED**	**1**	**USED**	**0..1**
OrderLineReference	EXCLUDED		USED	
LineID	**EXCLUDED**	**.**	**USED**	**1**
OrderReference	*EXCLUDED*		*USED*	*0..1*
OrderReference	EXCLUDED		USED	
ID	**EXCLUDED**	**.**	**USED**	**1**
IssueDate	*EXCLUDED*		*USED*	*0..1*
Party	USED		USED	
EndpointID	*EXCLUDED*		*USED*	*0..1*
PartyIdentification	**USED**	**1**	**USED**	**0..1**
PartyName	**USED**	**0..1**	**USED**	**1**
PostalAddress	**USED**	**1**	**USED**	**0..1**
PartyTaxScheme	USED	0..1	USED	0..n
PartyLegalEntity	*EXCLUDED*		*USED*	*0..1*
Person	*USED*	*0..1*	*EXCLUDED*	
PartyLegalEntity	EXCLUDED		USED	
RegistrationName	*EXCLUDED*		*USED*	*0..1*
CompanyID	**EXCLUDED**	**.**	**USED**	**1**
RegistrationAddress	*EXCLUDED*		*USED*	*0..1*
PartyTaxScheme	USED		USED	
RegistrationName	*EXCLUDED*		*USED*	*0..1*
CompanyID	*EXCLUDED*		*USED*	*0..1*
ExemptionReason	*EXCLUDED*		*USED*	*0..1*

PaymentMeans	USED		USED	
PaymentDueDate	**USED**	**0..1**	**USED**	**1**
InstructionID	*EXCLUDED*		*USED*	*0..1*
PaymentID	*EXCLUDED*		*USED*	*0..1*
CreditAccount	*EXCLUDED*		*USED*	*0..1*
PaymentTerms	USED		USED	
ReferenceEventCode	*EXCLUDED*		*USED*	*0..1*
SettlementPeriod	*EXCLUDED*		*USED*	*0..1*
PenaltyPeriod	*EXCLUDED*		*USED*	*0..1*
Period	USED		USED	
DurationMeasure	*USED*	*0..1*	*EXCLUDED*	
Description	*USED*	*0..1*	*EXCLUDED*	
Person	USED		EXCLUDED	
FirstName	**USED**	**1**	**EXCLUDED**	**.**
FamilyName	**USED**	**1**	**EXCLUDED**	**.**
Title	*USED*	*0..1*	*EXCLUDED*	
MiddleName	*USED*	*0..1*	*EXCLUDED*	
NameSuffix	*USED*	*0..1*	*EXCLUDED*	
Price	USED		USED	
BaseQuantity	*EXCLUDED*		*USED*	*0..1*
AllowanceCharge	*EXCLUDED*		*USED*	*0..1*
TaxCategory	USED		USED	
ID	**EXCLUDED**	**.**	**USED**	**1**
Percent	*EXCLUDED*		*USED*	*0..1*
TaxExemptionReasonCode	*EXCLUDED*		*USED*	*0..1*
TaxExemptionReason	USED	0..1	USED	0..1
TaxScheme	USED		USED	
ID	**EXCLUDED**	**.**	**USED**	**1**
JurisdictionRegionAddress	*USED*	*0..1*	*EXCLUDED*	

TaxSubtotal	USED		USED	
TaxableAmount	**USED**	**0..1**	**USED**	**1**
CalculationSequenceNumeric	*USED*	*0..1*	*EXCLUDED*	
Percent	*USED*	*0..1*	*EXCLUDED*	
BaseUnitMeasure	*USED*	*0..1*	*EXCLUDED*	
PerUnitAmount	*USED*	*0..1*	*EXCLUDED*	
TaxTotal	USED		USED	
TaxSubtotal	USED	1..n	USED	0..n

Figure 4.10: Gap Analysis Results at Component Level

At the business document content level, the elements that may cause a problem are presented. In order to better identify the problematic elements, we tried to validate an example Invoice document that conforms to UBLTR with ”NES Profile 4 - Basic Invoice Only” schemas. In order to successfully validate the invoice instance with NES, some elements are commented out and some new elements are inserted as shown in Figure 4.11. The newly added elements are shown with bold characters, whereas commented out elements are within <!– –> characters. As it is clear from Figure 4.11, with only minor changes, interoperability can be established.

```
<Invoice xsi:schemaLocation="urn:oasis:names:specification:ubl:schema:xsd:Invoice-2 UBL-Invoice-2.0.xsd"
xmlns="urn:oasis:names:specification:ubl:schema:xsd:Invoice-2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:cac="urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2"
xmlns:cbc="urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2">
    <cbc:UBLVersionID>2.0</cbc:UBLVersionID>
    <cbc:CustomizationID>UBL-TR</cbc:CustomizationID>
    <cbc:ProfileID>UBL-TR-Profile-1</cbc:ProfileID>
    <cbc:ID>A123456</cbc:ID>
    <cbc:CopyIndicator>true</cbc:CopyIndicator>
    <!--cbc:UUID></cbc:UUID-->
    <cbc:IssueDate>2008-01-02</cbc:IssueDate>
    <cbc:InvoiceTypeCode>SatisFaturasi</cbc:InvoiceTypeCode>
    <cbc:DocumentCurrencyCode>TRL</cbc:DocumentCurrencyCode>
    <!--cbc:LineCountNumeric>1.0</cbc:LineCountNumeric-->
    <cac:Signature>
        <cbc:ID/>
        <cac:SignatoryParty>
            <cbc:WebsiteURI/>
            <cac:PartyIdentification>
                <cbc:ID/>
            </cac:PartyIdentification>
            <cac:PartyName>
                <cbc:Name></cbc:Name>
            </cac:PartyName>
            <!--cac:PostalAddress>
                <cac:AddressLine>
```

```
                                        <cbc:Line/>
                                    </cac:AddressLine>
                                    <cac:Country>
                                        <cbc:Name/>
                                    </cac:Country>
                                </cac:PostalAddress-->
                            </cac:SignatoryParty>
                        </cac:Signature>
                        <cac:AccountingSupplierParty>
                            <cac:Party>
                                <cbc:WebsiteURI>www.satici.com</cbc:WebsiteURI>
                                <cac:PartyIdentification>
                                    <cbc:ID schemeID="VKN">1234567890</cbc:ID>
                                </cac:PartyIdentification>
                                <cac:PartyName>
                                    <cbc:Name>SATICI A.Ş.</cbc:Name>
                                </cac:PartyName>
                                <cac:PostalAddress>
                                    <cac:AddressLine>
                                        <cbc:Line>Atatürk Cad. 06000 ANKARA</cbc:Line>
                                    </cac:AddressLine>
                                    <cac:Country>
                                        <cbc:IdentificationCode>TR</cbc:IdentificationCode>
                                        <!--cbc:Name>Türkiye</cbc:Name-->
                                    </cac:Country>
                                </cac:PostalAddress>
                                <cac:PartyTaxScheme>
                                    <cac:TaxScheme>
                                        <cbc:ID></cbc:ID>
```

```
                                        <!--cac:JurisdictionRegionAddress>
                                            <cac:AddressLine>
<cbc:Line>BüyükMükellefler</cbc:Line>
                                            </cac:AddressLine>
                                            <cac:Country>
<cbc:IdentificationCode>TR</cbc:IdentificationCode>
                                                <cbc:Name>Türkiye</cbc:Name>
                                            </cac:Country>
                                        </cac:JurisdictionRegionAddress-->
                                    </cac:TaxScheme>
                                </cac:PartyTaxScheme>
                                <cac:Contact>
                                    <cbc:Telephone>(0312)1234567</cbc:Telephone>
                                    <cbc:Telefax>(0312)1234568</cbc:Telefax>
                                    <cbc:ElectronicMail>satici@satici.com</cbc:ElectronicMail>
                                </cac:Contact>
                            </cac:Party>
                        </cac:AccountingSupplierParty>
                        <cac:AccountingCustomerParty>
                            <cac:Party>
                                <cbc:WebsiteURI>www.alici.com</cbc:WebsiteURI>
                                <cac:PartyIdentification>
                                    <cbc:ID schemeID="VKN">1234567891</cbc:ID>
                                </cac:PartyIdentification>
                                <cac:PartyName>
                                    <cbc:Name>ALICI LTD. ŞTİ.</cbc:Name>
                                </cac:PartyName>
                                <cac:PostalAddress>
```

```
                        <cac:AddressLine>
                            <cbc:Line>Mustafa Kemal Cad. 06000 ANKARA</cbc:Line>
                        </cac:AddressLine>
                        <cac:Country>
                            <cbc:IdentificationCode>TR</cbc:IdentificationCode>
                            <!--cbc:Name>Türkiye</cbc:Name-->
                        </cac:Country>
                    </cac:PostalAddress>
                    <cac:PartyTaxScheme>
                        <cac:TaxScheme>
                            <cbc:ID></cbc:ID>
                            <!--cac:JurisdictionRegionAddress>
                                <cac:AddressLine>
                                    <cbc:Line>BüyükMükellefler</cbc:Line>
                                </cac:AddressLine>
                                <cac:Country>
                                    <cbc:IdentificationCode>TR</cbc:IdentificationCode>
                                    <cbc:Name>Türkiye</cbc:Name>
                                </cac:Country>
                            </cac:JurisdictionRegionAddress-->
                        </cac:TaxScheme>
                    </cac:PartyTaxScheme>
                    <cac:Contact>
                        <cbc:Telephone>(0312)1234569</cbc:Telephone>
                        <cbc:Telefax>(0312)1234560</cbc:Telefax>
                        <cbc:ElectronicMail>alici@alici.com</cbc:ElectronicMail>
                    </cac:Contact>
                </cac:Party>
```

```
        </cac:AccountingCustomerParty>
        <cac:PaymentMeans>
                <cbc:PaymentMeansCode>42</cbc:PaymentMeansCode>
                <cbc:PaymentDueDate>2009-08-12</cbc:PaymentDueDate>
                <!--cbc:PaymentChannelCode>BANKA</cbc:PaymentChannelCode-->
                <!--cac:PayeeFinancialAccount>
                        <cbc:ID>BBB Bankası Ank. CCC Şb. 00000001-0001</cbc:ID>
                        <cbc:CurrencyCode>TRL</cbc:CurrencyCode>
                </cac:PayeeFinancialAccount-->
        </cac:PaymentMeans>
        <cac:PaymentTerms>
                <cbc:Note>30 GÜN VADELİ</cbc:Note>
        </cac:PaymentTerms>
        <cac:TaxTotal>
                <cbc:TaxAmount currencyID="TRL">329.24</cbc:TaxAmount>
                <cac:TaxSubtotal>
                        <cbc:TaxableAmount currencyID="TRL">1829.10</cbc:TaxableAmount>
                        <cbc:TaxAmount currencyID="TRL">329.24</cbc:TaxAmount>
                        <cbc:TransactionCurrencyTaxAmount
currencyID="TRL">0.0</cbc:TransactionCurrencyTaxAmount>
                        <!--cbc:Percent>18.0</cbc:Percent-->
                        <cac:TaxCategory>
                                <cbc:ID></cbc:ID>
                                <cac:TaxScheme>
                                        <cbc:ID></cbc:ID>
                                        <cbc:TaxTypeCode>KDV</cbc:TaxTypeCode>
                                </cac:TaxScheme>
                        </cac:TaxCategory>
                </cac:TaxSubtotal>
        </cac:TaxTotal>
```

```
        <cac:LegalMonetaryTotal>

                <cbc:LineExtensionAmount
currencyID="TRL">1829.10</cbc:LineExtensionAmount>

                <cbc:TaxExclusiveAmount
currencyID="TRL">1829.10</cbc:TaxExclusiveAmount>

                <cbc:TaxInclusiveAmount
currencyID="TRL">2158.34</cbc:TaxInclusiveAmount>

                <cbc:PayableRoundingAmount
currencyID="TRL">0</cbc:PayableRoundingAmount>

                <cbc:PayableAmount currencyID="TRL">2158.34</cbc:PayableAmount>

        </cac:LegalMonetaryTotal>

        <cac:InvoiceLine>

                <cbc:ID>1</cbc:ID>

                <cbc:InvoicedQuantity unitCode="NIU">30</cbc:InvoicedQuantity>

                <cbc:LineExtensionAmount
currencyID="TRL">1829.10</cbc:LineExtensionAmount>

                <cac:Item>

<cbc:Name>Ürün 1</cbc:Name>

                        <cac:SellersItemIdentification>

                                <cbc:ID>P/N:000000000001</cbc:ID>

                        </cac:SellersItemIdentification>

                </cac:Item>

                <cac:Price>

                        <cbc:PriceAmount currencyID="TRL">60.97</cbc:PriceAmount>

                </cac:Price>

        </cac:InvoiceLine>

</Invoice>
```

Figure 4.11: UBLTR Invoice Instance Validated with NES Profile 4 - Basic Invoice Only

CHAPTER 5

RELATED WORK

Providing an environment or a tool for designing new document building blocks has only been addressed by a number of initiatives but there is no available solution on the market, which is as comprehensive as iSURF eDoCreator. In this section, the details and comparison of these tools will be provided.

One example is the GENESIS project (Integration for SMEs, Governmental Organizations and Intermediaries in the New European Union) that is supported by the European Commission. In the course of the GENESIS project, a living platform that grows over time and provides users with the possibility to seamlessly model and set-up business relations with other users and execute these based on a collaboratively designed standard is presented [33]. To achieve interoperability on both a business process and data level, an integrated modeling concept based on ISO 15000-5 CCTS specification (developed by UN/CEFACT) is produced. Similar to iSURF eDoCreator it has a common repository of data and process components [34]. The GENESIS document modeling environment is integrated with business process definition and provides basic customization mechanisms. Furthermore, it is a desktop application which lacks a collaborative feature.

Another important environment for modeling is GEFEG. FX [35], which is a more comprehensive tool than a modeling environment including validation tools and message generation capability. Furthermore, GEFEG also supports Electronic Data Interchange Format (EDI), XML and any property format as document syntaxes. However, it is not free and the details of the tool are not detailed in any of the available resources.

The most similar effort to a part of our work was announced by System Applications and

Product in Data Processing (SAP AG). SAP introduced a tool called ”Warp 10” as a tool that can be used via the web in a wiki-based environment, is rich in semantic support, handles data integration, modeling, and mapping, and leverages CCTS to maximize reuse and sharing of core component artefacts in real time. The envisioned functionalities of WARP are as follows [37]:

> ”Warp 10 is a Semantic Web ontology-based data integration, modeling and mapping tool that leverages the semantics of meta data by implementing the semantic-based approach described in ISO 15000-5 Core Component Technical Specification (CCTS). The Warp 10 modeling environment is wiki-based, collaborative, evolutionary, and autonomous. Warp 10 uses CCTS, the Core Component Library of UN/CEFACT, and the SAP GDT catalogue to create and maintain data content ontologies for further reuse. The Wiki nature of the tool for the first time enables CCTS users to view the design time modeling activities of their counterparts to maximize reuse and minimize duplicative efforts. Warp 10 is the first step to the future Web 3.0 - a combination of Web 2.0 and Semantic Web technologies. Warp 10 is based on SAP NetWeaver tools and offers both, the smooth integration and extension of SAP GDTs as well as the smooth integration and consolidation of existing investments in data integration and mapping through its semi automatic mapping approach”.

This tool is not yet available in the market, and will be a commercial tool. iSURF eDoCreator addresses the modeling part of this tool and it seems they will be competitive when Warp 10 is available on the market. UBLish is another aid for UBL adopters, which mostly address the needs of UBL implementers rather than UBL document designers. It provides the user-friendly Excel spreadsheet forms, which handles unsophisticated human data entry for users to fill up data just once to generate electronic documents. UBLish takes the spreadsheet and transforms it to the final XML instance according to XPath-look-alike specifications specified in the cells’ comment blocks. Another tool aiding UBL implementers is UBLer [37], which is originally developed to produce and test the CODICE’s conceptual model [10]. UBLer is a Java tool able to produce W3C XSD Schemes from a UBL-2.0 compliant model. Given an Open Document Spreadsheet with UBL NDR compliant entries, UBLer builds as many schemes as specified by its configuration file.

Additionally, this tool also generates Plain Old Java Objects (POJOS) and Open Document Text documentation based on the above mentioned model.

CHAPTER 6

CONCLUSION AND FUTURE WORK

Today, the application of information and communication technologies to business and governmental domain is on the agenda of many countries and most of them have already accomplished some developments for establishing complete national/regional electronic collaborations. The main aim of these efforts is to make collaborations interoperable and standard-based. This is even valid in the cross-border case. Today an enterprise's competitiveness is to a large extent determined by its ability to seamlessly interoperate with others.

For example, the European Commission has published the i2010 Strategy Framework and it has explicitly identified interoperability as a key bottleneck that should be tackled. iSURF eDoCreator will address this objective by letting organizations form their documents from common components constituting a common denominator for horizontal standards. In this way the companies, even the SMEs will be able to collaborate seamlessly with partners across a wide variety of business domains: this will promote their competitiveness, they will be more agile and will be able to expand possible business partnerships.

iSURF eDoCreator provides a document modeling environment for users to assemble their documents from common components according to UN/CEFACT CCTS document modeling methodology and customize these documents using UBL 2.0 customization methodology. When the document building blocks are derived from a common semantic specification with well-defined rules, it becomes possible to achieve electronic business document interoperability.

iSURF eDoCreator tool has been used to derive the UBL 2.0 conformant eInvoice Turkey [29] from the standard UBL 2.0 eInvoice. Such customizations are becoming popular recently, es-

pecially within the scope of the large scale integration project, PEPPOL (Pan-European Public Procurement Online) [28] currently being implemented in EU. PEPPOL will be producing UBL 2.0 conformant invoice, order, virtual company dossier and catalog schemas to be customized to the Member States and we believe that the publicly accessible iSURF eDoCreator tool provides an opportunity to help with these customizations. Additionally, the tool proved to be very useful in performing the gap analysis between NES/UBL Invoice [11] and eInvoice, Turkey [29].

The contributions of this thesis are as follows:

- Creating, extending, customizing document schemas conforming to UN/CEFACT CCTS methodology are tedious, labor intensive and time-consuming processes. In this thesis, the guidelines are converted to a machine processable format which let users to graphically design or customize their document models and follow the guidelines under controlled way.
- The document building blocks are stored in a spreadsheet format which does not let collaborative working and causes version inconsistencies. We designed a machine processable and graphical document models for each business document building block.
- We developed an on-line repository which provides collaborative working and discovery mechanisms.
- We have developed a Gap Analysis reporting tool, which shows the gap between two UBL 2.0 customizations and gives clues on how to implement wrappers to convert two messages.

A further impact of this work lies in the fact that UBL and UN/CEFACT are attempting to converge with two efforts into a single library. In this thesis, UN/CEFACT CCTS Item Discovery and New item submission guidelines are integrated with UBL Customization Guidelines to leverage the power of both concepts.

This work has been presented to several standardization bodies in UBL Rome Meeting, UBL TC Atlantic Tele-Conference and OSLO Meeting of European Committee for Standardization Workshop on Business Interoperability Interfaces for Public Procurement (CEN ISSS WS BII).

UBL TC Schema Generation Task Group plans to use iSURF eDoCreator for generating UBL version 2.1 document schemas. Furthermore, World Customs Organization group also plans to recommend iSURF eDoCreator to regional/national Custom Organizations to use them while creating their customized document schemas based on the global model that they produced. Furthermore, we plan to collaborate with these committees and provide the features that they need. The implementation process still continues as the feedback comes from users.

As a future work, the tool will be improved with an XSLT mapping tool which can recommend mapping options for two customized versions of UBL. The tool will be built on top of Gap Analysis Tool.

REFERENCES

[1] Chen, Q. et al., ”Peer-to-Peer Collaborative Internet Business Servers”, HP-Labs Technical Working Paper HPL-2001-14.

[2] Glushko R.J., McGrath T. ”Document Engineering for e-Business”DocEng ’02: Proceedings of the 2002 ACM Symposium on Document Engineering McLean, Virginia, USA, ACM Press, 2002, 42-48. http://doi.acm.org/10.1145/585058.585067

[3] IEEE Dictionary. Institute of Electrical and Electronics Engineers. IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries. New York, NY: 1990.

[4] UN/CEFACT. (2003, November) Core Components Technical Specification, Part 8 of the ebXML Framework, Version 2.01, [Online], Available: http://www.unece.org/-cefact/ebxml/CCTS/V2-01/Final.pdf., last visited on June 2009.

[5] OASIS. (2006, December) Universal Business Language (UBL) Version 2.0, Standard, [Online], Available: http://docs.oasis-open.org/ubl/os-UBL-2.0.zip., last visited on June 2009.

[6] UBL Guidelines for Customization Version 1.0, 30 September 2008, http://xml.coverpages.org/ni2007-04-04-a.html, last visited on June 2009.

[7] ”Conceptual Business Document Modeling using UN/CEFACT’s Core Components”; Talk: Sixth Asia-Pacific Conference on Conceptual Modeling (APCCM2009), Wellington, New Zealand; 01-20-2009 - 01-23-2009; in: ”Proceedings of the Sixth Asia-Pacific Conference on Conceptual Modeling (APCCM2009), Wellington, New Zealand, January 2009”, Australian Computer Society, 96 (2009), ISBN: 978-1-920682-77-4; Paper ID 4, 12 pages

[8] Offentlig Information Online UBL (OIOUBL), http://www.oioubl.info/classes/en/-index.html, last visited on June 2009.

[9] Svefaktura. Swedish Invoice. http://www.svefaktura.se/, last visited on June 2009

[10] CODICE Project COmponentes y Documentos Interoperables para la Contratacion Electronica (Interoperable Components and Documents for Electronic Procurement), http://www.meh.es/en-GB/Servicios/Contratacion/Junta%20Consultiva%20de-%20Contratacion%20Administrativa/Paginas/CODICE.aspx, last visited on June 2009.

[11] NES. UBL Northern European Subset.[Online], Available: http://www.nesubl.eu/, last visited on June 2009.

[12] Electronic Freight Management. [Online], Available: http://ops.fhwa.dot.gov/freight/-intermodal/efmmanifest/index.htm, last visited on June 2009.

[13] iSURF eDoCreator: e-Business Document Design and Customization Environment, [Online], Available: http://www.srdc.com.tr/eDoCreator, last visited on June 2009.

[14] UN/CEFACT Releases XML Schema for Cross Industry Electronic Invoice (CII), http://www.UNCEFACT.org, last visited on June 2009.

[15] OASIS UBL Turkish Localization Subcommittee, [Online], Available: http://www.oasis-open.org/committees/sc_home.php?wg_abbrev=ubl-trlsc, last visited on June 2009.

[16] UN/CEFACT Core Component Library, [Online], Available: http://www.unece.org/cefact/codesfortrade/unccl/CCL08B.xls, last visited on June 2009.

[17] An Interoperability Service Utility for Collaborative Supply Chain Planning across Multiple Domains Supported by RFID Devices, the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 213031, http://www.srdc.com.tr/iSURF/, last visited on June 2009.

[18] OASIS SET TC Deliverable: Semantic Representations of the UN/CEFACT CCTS-based Electronic Business Document Artifacts (Editors: Dogac, Kabak), [Online], Available: http://www.oasis-open.org/apps/org/workgroup/set/, last visited on June 2009.

[19] Electronic Product Code, [Online], Available: http://www.epcglobalinc.org/home, last visited on June 2009.

[20] Gunther Stuhec and Mark Crawford, "Getting Started with UN/CEFACT XML NDR for CCTS." SAP Developer Network (SDN) Contribution, August 04, 2006.

[21] XML Common Business Library, [Online], Available: http://www.xcbl.org/index.shtml, last visited on June 2009.

[22] Schematron, [Online], Available: http://www.schematron.com/overview.html, last visited on June 2009.

[23] The Extensible Stylesheet Language Family, [Online], Available: www.w3.org/Style/-XSL/, last visited on June 2009.

[24] commerce eXtensible Markup Language, [Online], Available: http://www.cxml.org/, last visited on June 2009..

[25] RosettaNet, [Online], Available: http://www.rosettanet.org/cms/sites/RosettaNet/, last visited on June 2009.

[26] The Open Applications Group Integration Specification, [Online], Available: www.oagi.org, last visited on June 2009..

[27] UBL Overall Design Principles, [Online], Available: www.oasis-open.org/committees/-ubl/ndrsc/archive/design-principles.doc, last visited on June 2009.

[28] PEPPOL (Pan-European Public Procurement Online), [Online], Available: http://-www.peppol.eu/, last visited on June 2008.

[29] UBL eInvoice TR. [Online], Available: http://www.efatura.gov.tr/, last visited on June 2009.

[30] Gelir İdaresi Başkanlığı (GIB), Türkiye, [Online], Available: http://www.gib.gov.tr/-index.php?id=466, last visited on June 2009.

[31] Electronic Procurement in the EU opens doors to cross-border business, last visited on June 2009.

[32] Integration for SMEs, Governmental Organizations and Intermediaries in the New European Union (GENESIS Project) http://www.genesis-ist.eu/, last visited on June 2009.

[33] Yannis Charalabidis , Fenareti Lampathaki , Dimitris Askounis, Unified data modeling and document standardization using core components technical specification for electronic government applications, Journal of Theoretical and Applied Electronic Commerce Research, v.3 n.3, p.38-51, December 2008

[34] C. Schroth, G. Pemptroad, T. Janner, CCTS-based Business Information Modeling for Increasing Cross-Organizational Interoperability, in Proceedings of the 3rd International Conference on Interoperability for Enterprise Software and Applications (I-ESA 2007) and: Enterprise Interoperability II. New Challenges and Approaches (R. J. Gonçalves, J. Müller, K. Mertins, M. Zelm, Editors), Springer, 2007.

[35] GEFEG.FX: High Data Quality in Electronic Business Documents, [Online], Available: http://www.gefeg.com/en/index.htm, last visited on June 2009.

[36] Crawford, Mark; Stuhec, Gunther - SAP Network Blog: CCTS Modeler Warp 10 - The Speed of Data Integration and B2B - 20 November 2007

[37] UBLer, [Online], Available: http://www.staro.es/home/doku.php?id=ubler, last visited on June 2009.