A Survey and Analysis of Electronic Business Document Standards

YILDIRAY KABAK and ASUMAN DOGAC

Middle East Technical University

No document standard is sufficient for all purposes because the requirements significantly differ among businesses, industries, and geopolitical regions. On the other hand, the ultimate aim of business document interoperability is to exchange business data among partners without any prior agreements related to the document syntax and semantics. Therefore, an important characteristic of a document standard is its ability to adapt to different contexts, its extensibility, and its customization. The UN/CEFACT Core Component Technical Specification (CCTS) is an important landmark in this direction.

In this article, we present a survey and an analysis of some of the prominent UN/CEFACT CCTS-based electronic document standards. We describe their document design principles and discuss how they handle customization and extensibility. We address their industry relevance and the recent efforts for their harmonization and convergence. We conclude by mentioning some emerging efforts for the semantic interoperability of different document standards.

Categories and Subject Descriptors: J.1 [**Administrative Data Processing**]: *Business; government*; H.2.4 [**Database Management**]: Systems—*Distributed databases*

General Terms: Standardization, Design

Additional Key Words and Phrases: eBusiness, Document Interoperability Standards, UN/CEFACT Core Component Technical Specification (CCTS), OASIS Universal Business Language (UBL), OAGIS Business Object Documents (BODs), Global Standards One (GS1) XML

ACM Reference Format:

Kabak, Y. and Dogac, A. 2010. A survey and analysis of electronic business document standards. ACM Comput. Surv. 42, 3, Article 11 (March 2010), 31 pages.
DOI = 10.1145/1670679.1670681 http://doi.acm.org/10.1145/1670679.1670681

1. INTRODUCTION

Interoperability is the ability of two or more systems or components to exchange information and to use the information that has been exchanged [IEEE Dictionary 1990]. In other words, interoperability is said to exist between two applications when one application can accept data (including data in the form of a service request) from the other

The research leading to these results received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 213031, the iSURF Project, and in part by the Scientific and Technical Research Council of Turkey (TÜBÍTAK), Project No: EEEAG 105E068.

Authors' addresses: Department of Computer Engineering, Software R&D Center, Middle East Technical University (METU), 06531, Ankara, Turkey; email: {yildiray, asuman}@srdc.metu.edu.tr.

and perform the task in an appropriate and satisfactory manner (as judged by the user of the receiving system) without the need for extra operator intervention [Brown and Reynolds 2000].

Interoperability of business applications can be investigated at three broad layers: the communication layer, the business processes layer, and the document layer. In this article we focus on the document layer, which addresses the interoperability of the document content exchanged.

Business document interoperability initiatives started in the 1970s before the invention of the Internet. The first standard developed was the Electronic Data Interchange (EDI) framework, where document exchange was realized through dialup connections using proprietary networks. (See Table I for Web addresses of all frameworks, languages, electronic libraries, tools, organizations, etc., discussed in this article.)

Starting with the late 1990s, eXtensible Markup Language (XML) became popular for describing data exchanged on the Internet. The relative human readability and the amount of XML tools available made XML a popular basis for a number of new document standards such as Common Business Library (CBL) and Commerce XML (cXML). This progress has been evolutionary because the later standards used the EDI experience. For example, CBL became XML Common Business Library [xCBL] after including EDI experience in CBL.

EDI, CBL, and xCBL are horizontal industry standards addressing several industry domains. There are also several vertical industry specific standard initiatives such as the ones from the North American Automotive Industry Action Group [AiAG], Health Level 7 (HL7) Standards Development Organization, Petroleum Industry Data Exchange (PIDX) Committee, Chemical Industry Data Exchange (CIDX) Organization, Open Travel Alliance (OTA), and RosettaNet Consortium (Rosetta Net), to name but a few.

The earlier standards focused on static message/document definitions which were inflexible to adapt to 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 defining flexible and adaptable business documents came from the UN/CEFACT Core Components Technical Specification (CCTS) in the early 2000s. UN/CEFACT CCTS provides a methodology to identify a set of reusable building blocks, called *Core Components* (CCs) to create electronic documents. Core Components represent the common data elements of everyday business documents such as "Address," "Amount," or "Line Item." These reusable building blocks are then assembled into business documents such as "Order" or "Invoice" by using the CCTS methodology. Core components are defined to be context-independent so that they can later be restricted to different contexts. Many core components defined by UN/CEFACT are available to users from the UN/CEFACT Core Component Library (UN/CCL).

This concept of defining context-free reusable building blocks, which are available from a single common repository, is an important innovation in business document interoperability for the following reasons:

—The incompatibility in electronic documents is incremental rather than wholesale. The users are expected to model their business documents by using the existing core components and by restricting them to their context with well-defined rules.
—The dynamic creation of interoperable documents becomes possible because if users cannot find proper components to model their documents, they can create and publish new core components.
—The horizontal interoperability among different industries is greatly facilitated by using a single common repository and by customizing the components to different industry contexts.

Table 1. Web Adresses

AAIA. Automotive Aftermarket Industry Association. http://www.aftermarket.org/Home.asp.
AiAG. Automotive Industry Action Group. http://www.aiag.org/.
ATG2-NDR. UN/CEFACT Applied Technology Group (ATG) XML Syntax, XML Naming and Design Rules.
http://www.uncefactforum.org/ATG/Documents/ATG/Downloads/XMLNamingAndDe%signRulesV2.0.pdf.
CBL. Common Business Library. http://xml.coverpages.org/cbl.html.
CCTS. UN/CEFACT Core Components Technical Specification.
http://www.unece.org/cefact/ebxml/CCTS_V2-01_Final.pdf.
CIDX. Chemical Industry Data Exchange. http://www.cidx.org/.
CLR TC. The OASIS Code List Representation Technical Committee.
http://www.oasis-open.org/committees/codelist.
Crawford. M. Crawford, Core Components Adoption on the Rise.
https://www.sdn.sap.com/irj/sdn/weblogs?blog=/pub/wlg/5395.
cXML. Commerce XML. http://cxml.org/.
EAN. European Article Number. http://en.wikipedia.org/wiki/European_Article_Number/.
EANCOM. European Article Number Communication.
http://www.gs1.org/productssolutions/ecom/eancom/.
ebBP. ebXML Business Process. http://docs.oasis-open.org/ebxml-bp/2.0.4/OS/.
EDI. Electronic Data Interchange. http://en.wikipedia.org/wiki/Electronic_Data_Interchange.
EDIINT-AS1. T. Harding, R. Drummond, C. Shih, MIME-based Secure Peer-to-Peer Business Data Interchange over the Internet, RFC 3335, Sept 2002. http://www.ietf.org/rfc/rfc3335.txt.
EDIINT-AS2. D. Moberg, R. Drummond, MIME-Based Secure Peer-to-Peer Business Data Interchange Using HTTP, Applicability Statement 2 (AS2), RFC 4130, July 2005.
http://www.ietf.org/rfc/rfc4130.txt.
EFT. Electronic Funds Transfer. http://en.wikipedia.org/wiki/Electronic_funds_transfer.
EPCglobal. Electronic Product Code Global. http://www.gs1.org/productssolutions/epcglobal/.
GDSN. GS1 Global Data Synchronisation Network. http://www.gs1.org/productssolutions/gdsn/.
GS1. Global Standard One. http://www.gs1.org/.
GS1 GDD. Global Standard One, Global Data Dictionary. http://gdd.gs1.org/.
GS1 XML. Global Standard One XML. http://www.gs1.org/productssolutions/ecom/xml/.
HL7. Health Level 7. http://www.hl7.org/.
IATA. International Air Transport Association. http://www.iata.org/index.htm.
ICH. ANSI ASC X12 ISA Interchange Control Header Segment.
http://www.rawlinsecconsulting.com/x12tutorial/x12syn.html.
ICHS. UN/EDIFACT UNB Interchange Header Segment.
http://www.unece.org/trade/edifact/untdid/d422_s.htm.
ISO Codes. International Standards Organization Codes.
http://www.unece.org/cefact/codesfortrade/codes_index.htm.
ISO11179. ISO/IEC 11179-5: Naming and identification principles.
http://standards.iso.org/ittf/PubliclyAvailableStandards/c035347_ISO_IE%
C_11179-5_2005(E).zip.
MIT-AutoID. Auto-ID Labs at MIT. http://autoid.mit.edu/cs/.
MoU. Memorandum of Understanding on electronic business between IEC, ISO, ITU, and UN/ECE.
http://www.itu.int/ITU-T/e-business/files/mou.pdf.
NES. UBL Northern European Subset. http://www.nesubl.eu/.
OAGi. Open Applications Group. http://www.openapplications.org/.
OAGIS. Open Applications Group Integration Specification 9.0.
http://www.openapplications.org/downloads/oagis/loadfrm9.htm.
OAGIS-Usage. Open Applications Group (OAGi) at 10 Years: A Look Back and Forward.
http://webservices.sys-con.com/read/47282.htm.
ODETTE. Organisation for Data Exchange by Tele Transmission in Europe.
http://www.odette.org/html/home.htm.
OIOUBL. Offentlig Information Online UBL.
http://www.oio.dk/dataudveksling/ehandel/hoeringer/oioubl.
Oracle. Oracle Corporation. http://www.oracle.com/products/middleware/docs/oracle_ebs_and_soa.pdf
http://www.oracle.com/technology/products/applications/integration/1147%_EBS_and_SOA.ppt.
OTA. OpenTravel Alliance. http://www.opentravel.org/.
OWL. Web Ontology Language. http://www.w3.org/2004/OWL/.
PIDX. Petroleum Industry Data Exchange. http://www.pidx.org/.
RosettaNet. http://www.rosettanet.org/.
Rowell, M. Rowell, The Open Applications Group Integration Specification.
http://www.ibm.com/developerworks/xml/library/x-oagis/.

(*Continued on next page*)

Table 1. *Continued*

SAP. SAP - Systemanalyse und Programmentwicklung. `http://www.sap.com/index.epx`.
Schematron. `http://www.ldodds.com/papers/schematron_xsltuk.html`.
STAR. Standards for Technology in Automotive Retail. `http://www.starstandard.org/`.
Stuhec. Gunther Stuhec, How to Solve the Business Standards Dilemma—The CCTS based Core Data Types. `https://www.sdn.sap.com/irj/sdn/go/portal/prtroot/docs/library/uuid/500\%db5c9-0e01-0010-81aa-d73cdd30df9a`.
Stuhec2. Gunther Stuhec, How to Solve the Business Standards Dilemma: The Context Driven Business Exchange. `https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uui%d/a6c5dce6-0701-0010-45b9-f6ca8c0c6474`.
Svefaktura. Swedish Invoice. `http://www.svefaktura.se/SFTI_Basic_Invoice20051130_EN/SFTI%20Basic%2%0Invoice_1.0/index.html`.
SWIFT. Society for Worldwide Interbank Financial Telecommunication. `http://www.swift.com/`.
UBL. Universal Business Language. `http://www.oasis-open.org/committees/ubl/`.
UBL NDR. Universal Business Language Naming and Design Rules. `http://docs.oasis-open.org/ubl/os-UBL-2.0/doc/ndr/NDR-checklist.pdf`.
UBL-SBS. Universal Business Language Small Business Subcommittee. `http://www.oasis-open.org/committees/sc_home.php?wg_abbrev=ubl-sbsc`.
UBLSchemas. Universal Business Language 2.0 Schemas. `http://docs.oasis-open.org/ubl/os-UBL-2.0/`.
UBP. Universal Business Process. `http://docs.oasis-open.org/ubl/cs-UBL-1.0-SBS-1.0/universal-business-pr%ocess-1.0-ebBP/`.
UCC. Uniform Code Council. `http://www.uc-council.org/`.
UN/CCL. United Nations Core Component Library. `http://www.unece.org/cefact/codesfortrade/unccl/CCL07A.xls`.
UN/EDIFACT. United Nations Directories for Electronic Data Interchange for Administration, Commerce and Transport. `http://www.unece.org/trade/untdid/welcome.htm`.
UN/EDIFACT 1131. UN/EDIFACT 1131 Data Element, Code list identification code. `http://www.unece.org/trade/untdid/d00a/tred/tred1131.htm`.
UN/EDIFACT 3055. UN/EDIFACT 3055 Data Element, Code list responsible agency code, note = `http://www.unece.org/trade/untdid/d00a/tred/tred3055.htm`.
UN/SBDH. UN/CEFACT Standard Business Document Header Technical Specification. `http://www.gs1.org/docs/gsmp/xml/sbdh/CEFACT_SBDH_TS_version1.3.pdf`.
US/DOT. US Department of Transportation UBL Implementation. `http://www.oasis-open.org/committees/ubl/faq.php`.
VOLLMER, KEN VOLLMER. B2B Integration Trends. Forrester, `http://www.forrester.com/Research/Document/Excerpt/0,7211,42735,00.html%`.
X12. EDI ANSI X12. `http://www.x12.org/`.
xCBL. XML Common Business Library. `http://www.xcbl.org/`.
XML. Extensible Markup Language. `http://www.w3.org/XML/`.
XSL. Extensible Stylesheet Language. `http://www.w3.org/Style/XSL/`.

CCTS is gaining widespread adoption by both the horizontal and the vertical standard groups. Universal Business Language (UBL) was the first implementation of the CCTS methodology. Some earlier horizontal standards such as *Global Standard One* (GS1) XML (GS1 XML) and *Open Applications Group Integration Specification* (OAGIS), and some vertical industry standards such as CIDX and RosettaNet have also taken up CCTS.

In this article, we survey some of the prominent horizontal business document standards, namely, EDI, UN/CEFACT CCL, UBL 2.0, OAGIS BOD 9.0 and GS1 XML. EDI is not only the earliest standard but the experience and the knowledge gained in its development also affected the other standards development efforts. UN/CEFACT CCL, which is based on UN/CEFACT CCTS, is a promising standard initiative to support dynamic electronic business requirements. The rest of the standards we cover are UBL 2.0, OAGIS BOD 9.0, and GS1 XML, which are horizontal standards all based on UN/CEFACT CCTS.

The surveyed standards are first analyzed based on their document design principles: the document design principles involve the document artifacts used in composing the documents, the code lists used to convey the meaning of the values in the elements, and the use of XML namespaces. Furthermore, since all the document standards surveyed are based on UN/CEFACT CCTS, how this methodology is used in the design of the documents is also discussed.

We then discuss how the standards handle extensibility and customization. The standards basically handle the customization and extensibility in two ways: either by introducing an "extension" element into the document schema or by allowing users to change the document schema. When an "extension" element is used, the document schema remains unchanged and the user can put any extra information in this element. When the document schemas are modified to accommodate extensions, the document interoperability is reduced.

Another important issue is whether the standards address the other layers in the interoperability stack, namely, the communication layer and the business process layer. The communication layer addresses the transport protocol and the message header. The business process layer involves the sequencing of the messages, and the business processes.

We also point out the industry relevance of these standards by providing some major usage examples. Most of the standards covered have very wide industry takeup. Finally we conclude by mentioning the harmonization efforts and an emerging trend for the semantic interoperability of document standards.

Before we proceed any further, we clarify the use of the terms *message* and *document*. Some standards call a *document* what other standards call a *message*. We use these terms to mean the following: the data that is exchanged between parties is called a *message*, which contains a *transport header* and a *payload*. The *payload* may consist of one or more *documents*. It is the *document* that contains the actual business data although, most of the time, the document standards also provide transport configuration information to be passed to the *transport header*.

The article is organized as follows: Section 2 summarizes the EDI initiative. Section 3 describes the UN/CEFACT Core Component Technical Specification. Section 4 introduces the Universal Business Language (UBL) 2.0 standard. In Section 5, Open Applications Group Integration Specification (OAGIS) 9.0 is presented. The Global Standard One (GS1) XML standard is covered in Section 6 after briefly introducing the set of standards proposed by GS1. Section 7 contains an analysis of the presented standards with respect to document design principles, customization and extensibility, coverage of other layers of interoperability, and the industry relevance. Finally, Section 8 concludes the article by describing harmonization efforts and the emerging semantic approach to document standards interoperability. Since a large number of acronyms are introduced throughout the article, a list of all acronyms and their meanings is provided in Table II.

2. ELECTRONIC DATA INTERCHANGE

EDI has been developed through two main branches: ANSI X12 and UN/EDIFACT. In the U.S., the American National Standards Institute (ANSI) developed ANSI X12 and internationally EDI was standardized as UN/EDIFACT (United Nations/Electronic Data Interchange For Administration, Commerce, and Transport). Through both of these initiatives, a large number of standard electronic documents in plain-text, quote-delimited formats have been specified for domains like procurement, logistics, and finance. EDIFACT has also been standardised by the International Standards Organization as ISO 9735 (UN/EDIFACT).

The basic EDI architecture is shown in Figure 1. The communications are through the Value-Added Networks (VANs), which are responsible for routing, storing, and

Table II. List of Acronyms and Abbreviations

ABIE	Aggregate Business Information Entity
ACC	Aggregate Core Component
AiAG	Automotive Industry Action Group
ANSI	American National Standards Institute
ASBIE	Aggregate Business Information Entity
ASCC	Association Core Component
ATG	UN/CEFACT Applied Technology Group
B2B	Business-to-business
BBIE	Basic Business Information Entity
BBC	Basic Core Component
BIE	Business Information Entity
BOD	Business object document
CBL	Common Business Library
CCL	Core Component Library
CCT	Core Component Type
CCTS	Core Components Technical Specification
CEFACT	Centre for Trade Facilitation and Electronic Business
CIDX	Chemical Industry Data Exchange
CLR TC	OASIS Code List Representation Technical Committee
cXML	Commerce XML
EAN	European Article Number
ebXML	Electronic Business eXtensible Markup Language
ebBP	ebXML Business Process
EDI	Electronic Data Interchange
EDIFACT	Electronic Data Interchange For Administration, Commerce and Transport
EFT	Electronic Funds Transfer
EPC	Electronic Product Code
GDD	Global Data Dictionary
GDSN	Global Data Synchronization Network
GDT	Global data type
GS1	Global Standards One
HL7	Health Level Seven
HTTP	HyperText Transfer Protocol
HTTPS	Secured HyperText Transfer Protocol
IATA	International Air Transport Association
IEC	International Electrotechnical Commission
IG	Implementation guide
ISO	International Organization for Standardization
IT	Information technology
ITU	International Telecommunication Union
MIME	Multipurpose Internet Mail Extension
NDR	Naming and design rule
OAGI	Open Applications Group, Inc.
OAGIS	Open Applications Group Integration Specification
OASIS	Organization for the Advancement of Structured Information Standards
OTA	Open Travel Alliance
OWL	Web Ontology Language
PIDX	Petroleum Industry Data Exchange
QDT	Qualified data type
RFID	Radio frequency identification
SBDH	StandardBusinessDocumentHeader
SMTP	Simple Mail Transfer Protocol
STAR	Standards in Automotive Retail
SWIFT	Society for Worldwide Interbank Financial Telecommunication
UBL	Universal Business Language
UBP	Universal Business Process
UCC	Uniform Commercial Code
UDT	Unqualified data type

Continued on next page

Table II. *Continued*

UMM	UN/CEFACT Modeling Methodology
UN	United Nations
UNECE	United Nations Economic Commission for Europe
xCBL	XML Common Business Library
XML	eXtensible Markup Language
XSD	XML Schema
XSL	Extensible Stylesheet Language
XSLT	Extensible Stylesheet Language Transformations

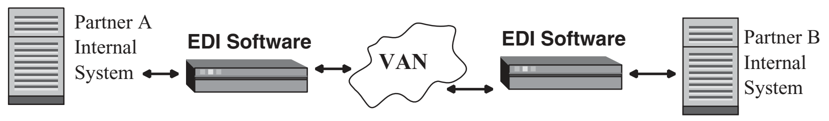
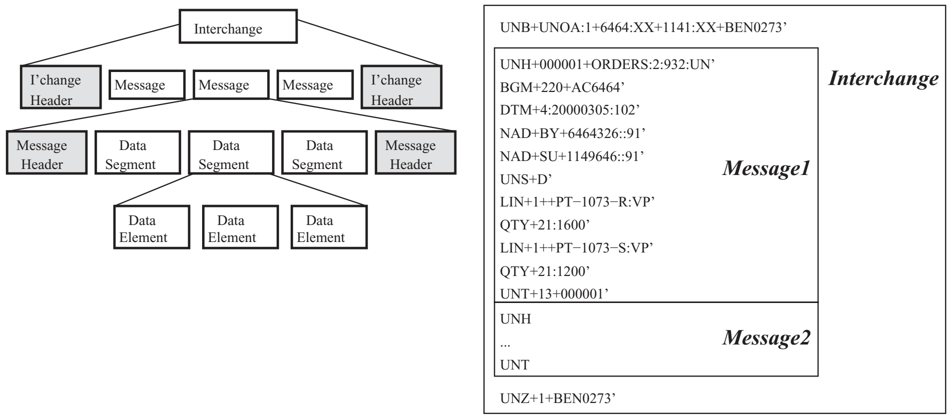


Fig. 1. The basic EDI architecture.



```
UNB+UNOA:1+6464:XX+1141:XX+BEN0273'
UNH+000001+ORDERS:2:932:UN'
BGM+220+AC6464'
DTM+4:20000305:102'
NAD+BY+6464326::91'
NAD+SU+1149646::91'
UNS+D'
LIN+1++PT−1073−R:VP'
QTY+21:1600'
LIN+1++PT−1073−S:VP'
QTY+21:1200'
UNT+13+000001'
UNH
...
UNT
UNZ+1+BEN0273'
```

(a) (b)

Fig. 2. (a) The basic EDI message structure; (b) an example EDI message.

delivering EDI messages. Special EDI adapters are implemented to interface the internal system of a partner to the value-added network. The particulars of the message syntax and interaction process are negotiated between partners in advance. Sometimes a dominant partner imposes its standards on smaller partners.

An EDI "interchange" document, as shown in Figure 2(a) consists of "messages," which are in turn composed of "data segments." The segments themselves consist of "data elements." Figure 2(b) shows an example EDI message.

When the Internet became an established networking environment starting in the mid-1990s, there were several updates to the EDI architecture. First, the Internet protocol for email, the Simple Mail Transfer Protocol (SMTP), and the File Transfer Protocol (FTP), came to be used to transfer EDI documents directly between parties connected to the Internet. Later, once the World Wide Web and its transfer protocol, the HyperText Transfer Protocol (HTTP), was popularized, this became another mechanism for EDI document transfer.

3. UN/CEFACT CORE COMPONENT TECHNICAL SPECIFICATION

UN/CEFACT Core Components Technical Specification (CCTS) is defined as Part 8 of the ebXML (electronic business XML) Framework and is approved as ISO 15000-5 (CCTS).

The essence of UN/CEFACT CCTS is to design documents from standard, reusable building blocks, called Core Components (CCs). The aim is to provide interoperability among electronic business documents by requiring all Business Information Entities (BIEs) to be related back to the common core components. A considerable number of Core Components are available from the UN/CEFACT Core Component Library (CCL) for discovery and reuse, and more will be available as the work progresses.

The first step to provide interoperability based on Core Components is to represent values in the components consistently. Hence the starting point for the design of Core Components is the *Core Component Types* (CCTs) and *Data Types* (DTs).

3.1. Core Component Types and Data Types

Core Component Types constitute the leaf-level type space of UN/CEFACT Core Components. They specify the basic information types, such as amount, binary object, code, and date time, and they are built from primitive data types (e.g. binary, decimal, integer, and string). A CCT is composed of a *Content Component*, where the actual primitive content resides, and one or more *Supplementary Components*, which further describe the Core Component Types. In other words, Supplementary Components help to interpret a value in the Content Component.

For example, the “Code” CCT’s Content Component is of type string and has a set of Supplementary Components such as *Code List Agency Identifier*, which is the identifier of the Agency that maintains the code list, and *Code List Agency Name*, which is the name of the Agency that maintains the code list.

On the other hand, Data Types are based on one of the Core Component Types and further restrict them. In this respect, CCTs can be thought of as abstract types from which more specialized data types are produced. For example, in the current version of the UN/CEFACT data types, there is a Data Type, called the “CurrencyCode”. This Data Type is based on the “Code” CCT and restricts it as follows.

—*Content Component*. The value in the Content Component should be a three-letter code.

—*Code List Identifier*. The identifier of the code list is ISO 4217.

—*Code List Version Identifier*. The version of the code list is 2006-11-21.

The relationship among Core Component Types, Data Types, and other types of Core Components is shown in Figure 3 (CCTS). Up to now, UN/CEFACT has approved 10 Core Component Types and defined 35 permissible Data Types, and has undertaken their maintenance. Furthermore, the Data Types provided by UN/CEFACT can be used without restrictions (*Unqualified Data Types* (UDTs)) or further restricted (*Qualified Data Types* (QDTs)) to accommodate specific business needs. UN/CEFACT also provides the rules to restrict the Data Types to Qualified Data Types.

3.2. Naming Convention Used

A naming convention is necessary to consistently name the defined components to facilitate the comparison during the discovery and analysis process. Furthermore, ambiguities can be prevented such as developing multiple Core Components with different names that have the same meaning. The naming convention used in CCTS is derived

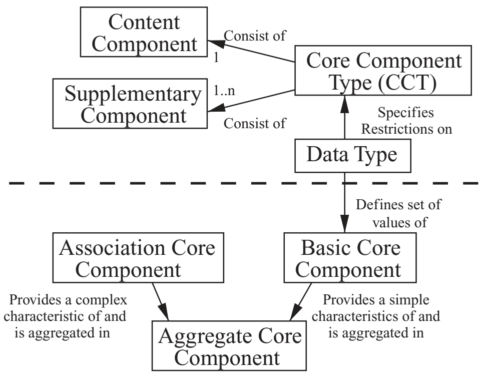

Content Component
Consist of
1
Supplementary Component
1..n
Consist of
Core Component Type (CCT)
Specifies Restrictions on
Data Type
Defines set of values of
Association Core Component
Basic Core Component
Provides a complex characteristic of and is aggregated in
Provides a simple characteristics of and is aggregated in
Aggregate Core Component

Fig. 3. Core component overview (CCTS).

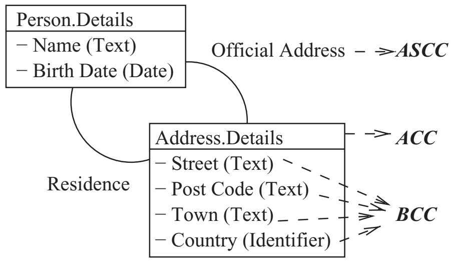


Fig. 4. Examples of BCCs, ACC, and ASCCs (CCTs).

from ISO 11179 Part 5 (ISO11179). This naming convention has three major parts: *Object Class*, *Property Term*, and *Representation Term*. For example, when the Core Component "Invoice. TaxAmount. Amount" is expressed according to the CCTS naming convention, "Invoice" is the Object Class, TaxAmount is the Property Term, and "Amount" is the Representation Term.

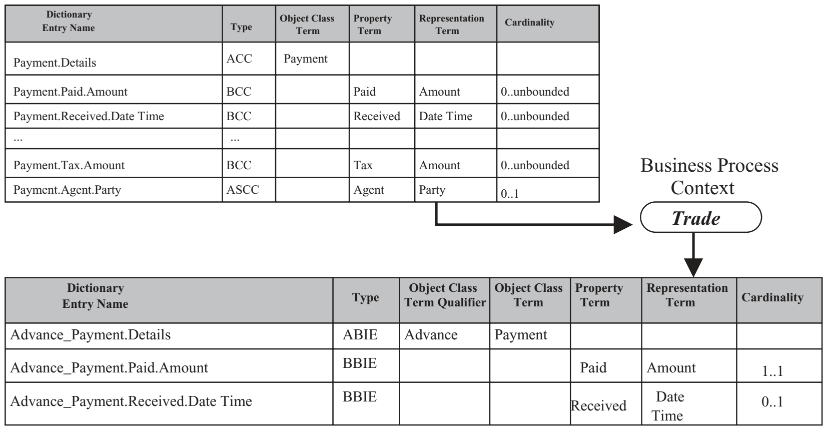
3.3. Types of Core Components

A Core Component is a reusable building block for creating electronic business documents. There are three types of Core Components.

—*Aggregate Core Component* (ACC). A distinct real world object with a specific business meaning such as "Address" or "Purchase Order" is termed an Aggregate Core Component. An Aggregate Core Component has at least one and possibly more Basic Core Components (BCCs). For example, as shown in Figure 4, "Address. Details" is an ACC containing several (BCCs).

—*Basic Core Component*. This describes a property of an ACC by using a Data Type. For example, as shown in Figure 4, "Address. Details. Street" is a Basic Core Component and is of the "Text" Data Type. In other words, the Data Types are used as Representation Terms of Basic Core Components.

—*Association Core Components* (ASCCs). Sometimes it is necessary to define an association between Aggregate Core Components. This is realized through Association Core Components. As shown in Figure 4, "Person. Details. Residence" is an Association Core Component referencing the "Address. Details" ACC.



Dictionary Entry Name	Type	Object Class Term	Property Term	Representation Term	Cardinality
Payment.Details	ACC	Payment			
Payment.Paid.Amount	BCC		Paid	Amount	0..unbounded
Payment.Received.Date Time	BCC		Received	Date Time	0..unbounded
...	...				
Payment.Tax.Amount	BCC		Tax	Amount	0..unbounded
Payment.Agent.Party	ASCC		Agent	Party	0..1

Dictionary Entry Name	Type	Object Class Term Qualifier	Object Class Term	Property Term	Representation Term	Cardinality
Advance_Payment.Details	ABIE	Advance	Payment			
Advance_Payment.Paid.Amount	BBIE			Paid	Amount	1..1
Advance_Payment.Received.Date Time	BBIE			Received	Date Time	0..1

Fig. 5. Customizing an Aggregate Core Component to the Business Process Context "Trade".

3.4. Business Information Entity

A Core Component is designed to be context-independent so that it can later be adapted to different contexts and reused. When a Core Component is restricted to be used in a specific business context, it becomes a *Business Information Entity* (BIE) and given its own unique name.

The possible business contexts that can be used are defined to be the *Business Process Context; Product Classification Context; Industry Classification Context; Geopolitical Context; Business Process Role Context; Supporting Role Context; System Capabilities Context*, and *Official Constraints Context*.

For example, when the Business Process Context is specialized to "Purchasing", and the Geopolitical Context is set to be "EU," the "Invoice. Tax. Amount" BCC becomes the "Invoice. VAT_ Tax. Amount" *Basic Business Information Entity* (BBIE).

Similarly, when an Association Core Component is used in a context, it becomes an *Association Business Information Entity* (ASBIE) and the Aggregate Core Component becomes *Aggregate Business Information Entity* (ABIE). For example, in Figure 5 an "Advance. Payment. Details" ABIE is created by customizing the "Payment. Details" ACC to the Business Process Context "Trade" as follows: an *Object Class Term Qualifier* is added as an additional property and the related BCCs are customized to create the BBIEs by restricting their cardinality.

Figure 6 (CCTS) gives the relationship between the types of Core Components and the corresponding Business Information Entities.

3.5. UN/CEFACT Core Component Library

The Core Component Library (UN/CCL) is the repository for UN/CEFACT CCTS artifacts. Currently there are quite a number of UN/CEFACT artifacts in the Core Component Library.

4. UNIVERSAL BUSINESS LANGUAGE 2.0

The Universal Business Language (UBL) initiative from OASIS adopts the UN/CEFACT Core Component Technical Specification (CCTS) approach and develops a set of standard XML business document definitions.

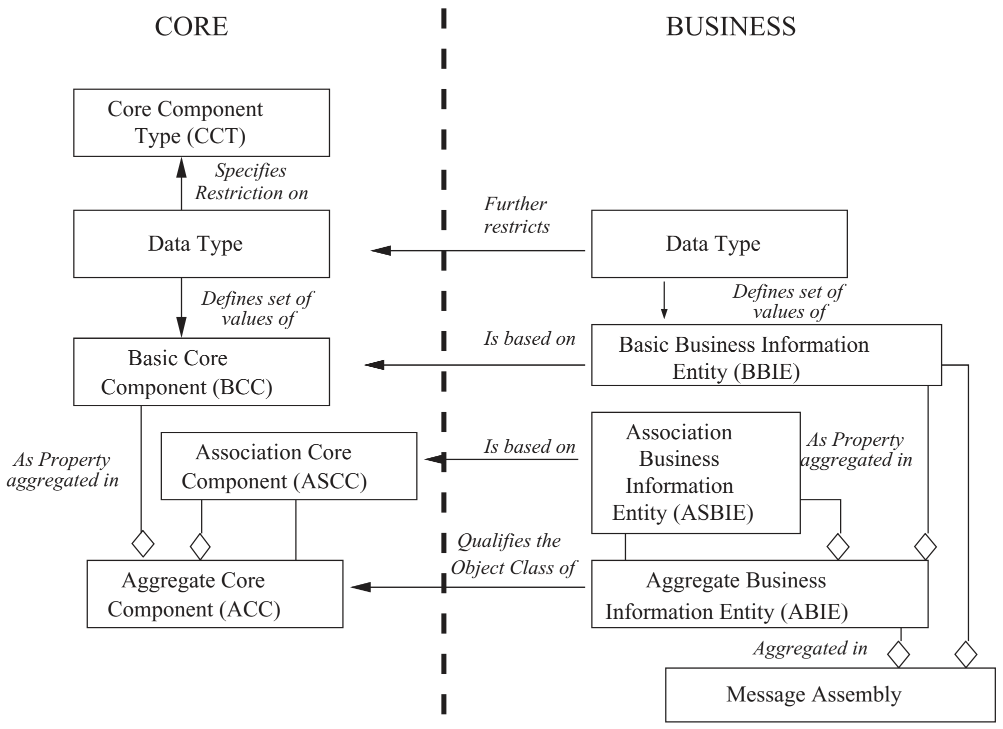

CORE
BUSINESS
Core Component Type (CCT)
Specifies Restriction on
Data Type
Further restricts
Data Type
Defines set of values of
Defines set of values of
Basic Core Component (BCC)
Is based on
Basic Business Information Entity (BBIE)
As Property aggregated in
Association Core Component (ASCC)
Is based on
Association Business Information Entity (ASBIE)
As Property aggregated in
Qualifies the Object Class of
Aggregate Core Component (ACC)
Aggregate Business Information Entity (ABIE)
Aggregated in
Message Assembly

Fig. 6. Relationship between Core Components and Business Information Entities (CCTS).

Currently, the approved version of UBL is 2.0 and there are 31 XML schemas for common business documents such as “Order,” “Despatch Advice,” and “Invoice.” In addition to the document definitions, UBL 2.0 provides a library of XML schemas (XSDs) (UBL Schemas) for reusable common data components like “Address,” “Item,” and “Payment” from which the documents are constructed. UBL 2.0 reuses Core Component Type and Data Type definitions from UN/CEFACT CCTS such as “AmountType,” “CodeType,” and “DateTimeType.” When UN/CEFACT CCTS Data Types are imported to UBL type space, they are termed *Unqualified Data Types* (UDTs). Additionally, UBL defines *Qualified Data Types* (QDTs), which are primarily for code lists such as *CurrencyCodeType* or *CountryIdentificationCodeType* defined for use within UBL.

At the time the UBL initiative had started, UN/CEFACT CCTS had not yet specified core components. Therefore UBL created its own BIEs based on CommerceOne’s xCBL (XML Common Business Library) 3.0 (xCBL) and the EDI for Administration Commerce and Trade (UN/EDIFACT) dictionary. Hence the UBL vocabulary consists primarily of Aggregate Business Information Entities (ABIEs).

Figure 7 shows the structure of the UBL documents. It should be noted that in addition to identifying conceptual BIEs, UBL uses the CCTS artifacts 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, ABIEs are used in two different ways: (1) the document ABIEs which represent UBL documents such as “Order” and “Invoice” and (2) more fine-grained reusable ABIEs such as “Address” and “Party.” As shown in Figure 6, an ABIE is composed

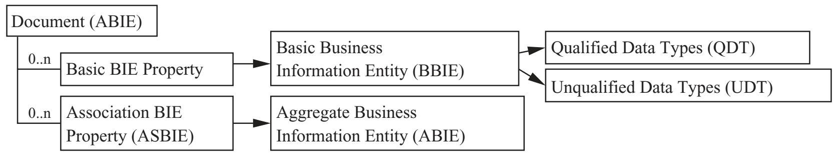

Document (ABIE)
0..n
Basic BIE Property
0..n
Association BIE Property (ASBIE)
Basic Business Information Entity (BBIE)
Aggregate Business Information Entity (ABIE)
Qualified Data Types (QDT)
Unqualified Data Types (UDT)

Fig. 7. The UBL components.

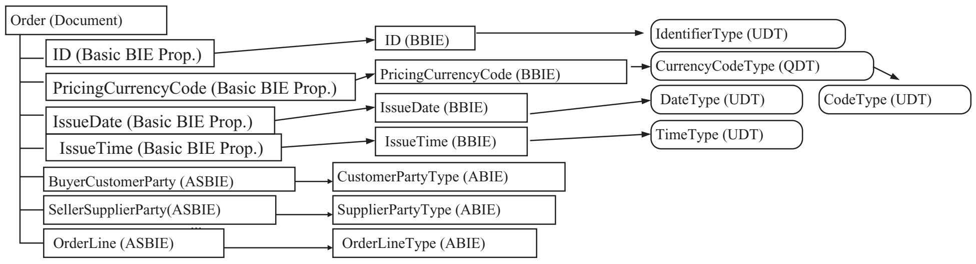


Fig. 8. An example UBL document schema structure.

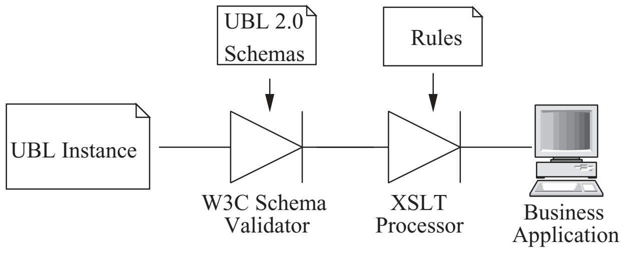


Fig. 9. Two-phase validation of UBL messages.

of BBIEs and ASBIEs as in UN/CEFACT CCTS. In UBL 2.0, according to the UBL 2.0 naming and design rules, this composition is realized through *BIE Properties*. A BBIE has a single content whose type is specified either with qualified Data Types or Unqualified Data Types. Figure 8 shows an example UBL 2.0 "Order" document.

4.1. UBL Customization and Extensibility

There are two types of customizations specified in UBL 2.0: Conformant customization and Compatible customization.

Before going into the details of customization, it is worth describing the validation of UBL documents. UBL 2.0 recommends a two-phase validation technique, as shown in Figure 9. In the first phase, an incoming UBL document is validated against UBL 2.0 XSD schemas (or customized versions of them). If the instance passes the first phase, in the second phase it is checked against the rules, which specify additional constraints on the values of the elements in the instance. Generally, the rules are specified through Extensible Stylesheet Language (XSL) or the Schematron languages. If the instance passes both of the phases successfully, it is delivered to the processing business application.

```
<Order
 xmlns="urn:oasis:names:specification:ubl:schema:xsd:Order-2"
 xmlns:cac="urn:oasis:names:specification:ubl:schema:xsd:CommonAggregateComponents-2"
 xmlns:cbc="urn:oasis:names:specification:ubl:schema:xsd:CommonBasicComponents-2"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

 <UBLExtensions xmlns="urn:oasis:names:specification:ubl:schema:xsd:
  CommonExtensionComponents-2">
  <UBLExtension>
   <ExtensionContent>
    <OrderExtensions>
     <productForm>Granule</productForm>
     <bonusPoint>100</bonusPoint>
    </OrderExtensions>
   </ExtensionContent>
  </UBLExtension>
 </UBLExtensions>
 ...
 <cac:BuyerCustomerParty> ...  </cac:BuyerCustomerParty>
 <cac:SellerSupplierParty> ...  </cac:SellerSupplierParty>
 <cac:AnticipatedMonetaryTotal> ...  </cac:AnticipatedMonetaryTotal>
 <cac:OrderLine> ...  </cac:OrderLine>
</Order>
```

Fig. 10. UBL extension example.

4.1.1. Conformant Customization of UBL 2.0. The key idea behind the conformant customization is that the XML instances in the customized implementation must also conform to the original standard UBL 2.0 schemas. There are four ways of performing conformant customizations:

(1) *Inserting additional elements through the use of an "UBLExtensions" element*. An optional UBLExtensions element appears as the first child of all UBL 2.0 documents and is used to include non-UBL data elements. For example, there could be elements containing data whose inclusion is mandated by law for certain business documents in certain regulatory environments. The UBLExtensions element is composed of multiple UBLExtension elements, each containing a single element ExtensionContent of type "xsd:any" to accommodate the widest possible range of extensions. This means that any well-formed XML element from any vocabulary can be inserted into an ExtensionContent element without modifying the schema.

    An example UBL extension is given in Figure 10 where the UBLExtensions element is inserted into the beginning of the order document. It contains a "productForm" element, which shows the requested form of the ordered product, and a "bonusPoint" element, which is the bonus amount gained by the buyer upon purchasing the ordered products.

(2) *Subsetting original UBL 2.0 schemas*. There are very many possible elements in a UBL document. For example, there are about 50,000 possible elements in a UBL Order Document. Most applications will not need all this data. Therefore, UBL 2.0 allows users to create subsets of its schemas. Subsets remove any optional information entities that are not necessary to the specific implementation. UBL 2.0 Small Business Subset (UBL-SBS) is an example of this subsetting mechanism.

(3) *Placing constraints on the value space of information entities and/or putting constraints among these values*. 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 requirement can be reflected in the UBL schemas by

type restriction; however, this requires schema modification. On the other hand, the latter type of requirement cannot be represented through XSD schemas. However, users can describe both of these constraints through Schematron or XSL rules and feed these rules into the second phase of validation as already described.

(4) *Customizing the code lists*. Code list customization is described in Section 4.1.3.

4.1.2. Compatible Customization of UBL 2.0. Sometimes conformant customization may not be sufficient for a specific implementation. Users may need to perform more complex modifications such as extending an ABIE, creating a new ABIE or creating a new document. To handle these cases, the compatible customization approach can be used. In compatible customization, the users modify an existing UBL 2.0 schema or create a new one by reusing the "largest suitable" aggregation from the UBL library. When performing compatible customization, the users need to follow the UBL Naming and Design Rules (UBL-NDR).

4.1.3. The Use of Code Lists. In UBL 1.0, the standard and the default code list values are specified directly in the UBL schemas as XSD enumeration constraints. This allows all UBL 1.0 instances to be validated in a single pass using generic XSD processors. However, the specification of the default values directly in the schemas also makes it difficult to modify the code lists to meet customization requirements.

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, these enumerations are specified in Unqualified Data Types from UN/CEFACT, and UBL 2.0 includes them as they are for the attribute values.

The other code lists used in UBL are not enumerated in the schema expressions. Instead of enumerating the codes in the XSD schemas, UBL uses a common base type called *CodeType*, which is an extension of "xsd:normalizedString" for all elements expressing values from the code lists. The UBL 2.0 package includes files for every code list. These files are separate from the provided XSD schemas and they are in a standard format. Trading partners can modify or replace any of these files to meet their business requirements. After this step, they can convert these files in proprietary format to Schematron or XSL rules. The OASIS Code List Representation Technical Committee (CLR TC) provides tools for this purpose. Later these rules can be fed into the second phase of validation as already described.

5. OPEN APPLICATIONS GROUP INTEGRATION SPECIFICATION (OAGIS) BUSINESS OBJECT DOCUMENTS VERSION 9.0

The Open Applications Group, Inc. (OAGi) is a not-for-profit open standards organization that defines electronic document standards called *Business Objects Documents* (BODs). Since its first release in 1995, several versions of Open Applications Group Integration Specification (OAGIS) BODs have been produced, the latest one being the OAGIS BOD version 9.0. This version has been redesigned to be based on the UN/CEFACT Core Components Technical Specification.

The BOD is based on a pair of concepts called the *Noun* and the *Verb*. The Verb identifies the action to be applied to the Noun. Noun is the object or document such as "PurchaseOrder," "RequestForQuote," and "Invoice" that is being acted upon. Examples of Verbs include "Cancel," "Get," "Process," and "Synchronize." The Verb and Noun combination provides the name of the BOD. For example, when the Verb is "Process" and the Noun is "PurchaseOrder," the name of the BOD is "ProcessPurchaseOrder." There are 77 Nouns and 12 Verbs defined in OAGIS 9.0.

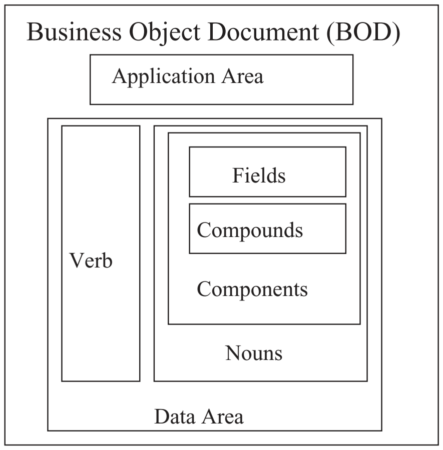


Fig. 11. The structure of the OAGIS Business Object Document.

The separation of Verb and Noun components increases the reusability of data. For example, the Noun “PurchaseOrder” contains all of the information that might be present in a “PurchaseOrder.” The instantiation of each of the possible Verb and Noun combinations then further restricts the document to a context. For example, in a “ProcessPurchaseOrder” transaction, business partners and line item data must be provided, whereas in a “CancelPurchaseOrder” only the order identifier is needed to carry out the transaction. Note that these constraints do not change the schema of a document. Rather, they provide the constraint rules to be applied in the validation of a BOD. Like UBL, OAGIS recommends a two-phase validation. When an OAGIS document is received, it is first validated against the corresponding XML Schema and afterwards against the corresponding Schematron/XSL rules. Only after the OAGIS instance document passes this two-phase validation is it delivered to the business application that processes the document content.

OAGIS provides some recommendations on the usage of Verbs. Verbs may come in pairs, meaning that the response to a Verb should be another specific Verb. For example, the response Verb of “Process” is “Acknowledge.”

As shown in Figure 11, a BOD is a message structure composed of an *ApplicationArea* and a *DataArea*. The ApplicationArea carries necessary information for transport software to send the message to the destination such as the sender, the signature of the sender, and the unique identifier of the BOD. The need for the ApplicationArea stems from the following: the application software that creates a BOD may be separate from the transport software that sends the BOD to the destination. Therefore the application software creating the BOD should provide the transport software with the necessary configuration information to send the BOD. In other words, the ApplicationArea contains the configuration information created by the application software and conveyed to the transport software.

The DataArea contains a single Verb and multiple Nouns. A Noun may be assembled from *Component*, *Compound*, and *Field* document artifacts. Components are large-grained building blocks and may in turn consist of other Components, Compounds, and Fields. Examples of Components include “PurchaseOrder Header,” “Party,” and “Address.” Compounds, which are used across all BODs, are a logical grouping of Fields (low-level elements). Examples include “Amount,” “Quantity,” “DateTime,” and “Temperature.” Fields are the lowest-level elements used in OAGIS Components and Compounds. Figure 12 shows an example BOD assembly with OAGIS artifacts.

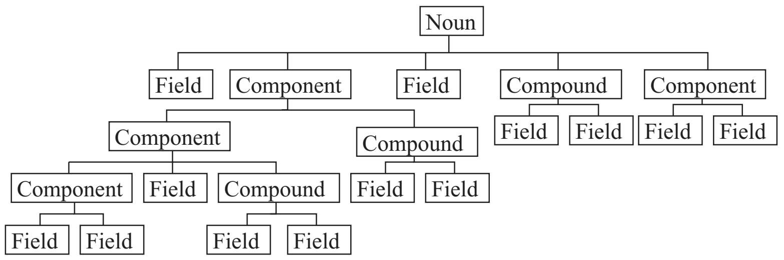


Fig. 12. OAGIS Business Object Document assembly example.

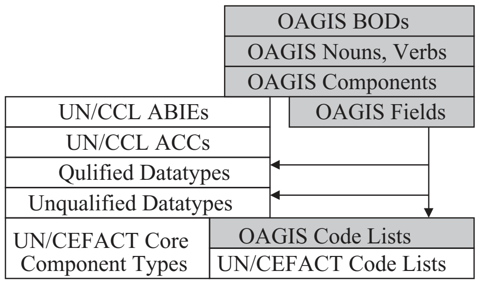


Fig. 13. OAGIS usage of UN/CEFACT CCT.

An OAGIS implementation of the Core Component Technical Specification (CCTS) is shown in Figure 13. In OAGIS 9.0, the Core Component Types and Unqualified Data Types are directly used in the OAGIS schemas. In other words, all OAGIS Field types are based on UN/CEFACT Core Component types. Furthermore, the code lists, such as ISO 54217 Currency Codes and ISO 5639 Language Codes, recommended by UN/CEFACT, are also used as described in Section 5.1.3.

As shown in Figure 13, OAGIS incorporates the UN/CEFACT ABIEs into OAGIS Components rather than using them directly. When using these ABIEs in their Components, OAGIS appends an "ABIEType" suffix to the name of the ABIE in order to identify that it is an ABIE from UN/CEFACT.

OAGIS Naming and Design Rules (NDR) are based on the version UN/CEFACT ATG2 Naming and Design Rules (NDR) (ATG 2-NDR).

5.1. OAGIS Extensibility

OAGIS provides two mechanisms to extend its specifications: *UserArea* extensions and *Overlay* extensions.

5.1.1. UserArea Extensions. The *UserArea* extensibility provides a means of adding implementation specific content to an existing OAGIS Component in an existing OAGIS BOD. When a few simple Fields are needed to complete the information for the exchange, UserArea extensions are used. There is a UserArea element of type "xsd:any" at the end of each OAGIS Component where the users can insert any valid XML instance without changing the original OAGIS schema.

For example, in Turkey, the addresses contain "Mahalle" information, which basically specify a district in a city. In OAGIS, the "Address" Component does not have such a Field to carry "Mahalle" information. This "Mahalle" information can be inserted in the UserArea part of the "Address" Component in a BOD instance when it is used in Turkey, as shown in Figure 14.

```
<Address>
....
        <UserArea xmlns:myTrImpl="http://www.myTrImpl.org">
                <myTrImpl:Mahalle>EsatOglu<myTrImpl:Mahalle>
        </UserArea>
</Address>
```

Fig. 14. UserArea example.

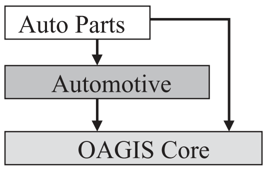


Fig. 15. OAGIS overlay layering example.

5.1.2. Overlay Extensions. When the users need more complex changes such as creation of a new BOD or creation of new a Component, an *Overlay* extension mechanism is used. The Overlay extensions result in the creation of new XML schemas for the BOD in their own separate namespaces. It should be noted that only Nouns and Components are Overlay extensible.

The overlay extension mechanism adopts a layering approach. New layers, called *overlays*, are defined in their own respective namespaces on top of core OAGIS Schemas. Specialized BODs and Components are defined by extending BODs from lower layers and/or by composing new BODs from a combination of existing components, extended components, and new components. In Figure 15, an example for overlays is shown where an "Automotive" overlay is created from core OAGIS schemas, whereas "Auto Parts," a subdomain of "Automotive," is built on "Automotive" and the OAGIS core.

With overlay extensions, users are allowed to create a new BOD, a Noun, a Component, a Compound, or a Field, or to extend any of the previously defined OAGIS artifacts. For example, a user may extend the "Invoice" Noun of OAGIS by adding the following: a new Component for representing total discounts, an existing Compound for grand total, and a new Field for a special purpose. Figure 16 shows how these extensions are realized. The user first creates a new Noun called "MyInvoiceType" by extending the "Invoice" provided by OAGIS. Afterwards, the user inserts the elements mentioned. Finally, the user defines the "MyInvoice" element of type "MyInvoiceType." Note that the "MyInvoice" element is in the same "xsd:substitutionGroup" as OAGIS "Invoice," which means that, anywhere the OAGIS "Invoice" element is included in a model, the "MyInvoice" element can be inserted as well. In order to preserve interoperability among different Overlay extensions, XSLT transformations are defined to convert an instance document conforming to an Overlay into another.

UserArea extensions are faster to apply than Overlay extensions. However, they do not provide the same level of control on the schemas as the Overlay extensions do. This is because the UserArea extensions are applied to the OAGIS BOD XML instance documents and not to the OAGIS BOD schema itself.

5.1.3. Code List Extensions. OAGIS uses and recommends the code lists from UN/CEFACT, and allows additional values to be present. This is accomplished as follows: OAGIS defines two "xsd:simpleType"s for each coded Field: (1) an enumeration type, which lists the codes to be used and (2) a "xsd:simpleType," which is a union of that enumeration type and the "xsd:normalizedString." In other

```
<xs:complexType name="MyInvoiceType">
 <xs:complexContent>
  <xs:extension base="oa:Invoice">
   <xs:sequence>
    <xs:element ref="ia:TotalDiscounts" minOccurs="0"/>
    <xs:element name="GrandTotal" type="oa:Amount" minOccurs="0"/>
    <xs:element name="MyInfo" type="xs:string" minOccurs="0"/>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>

<xs:element name="MyInvoice" type="my:MyInvoiceType"
  substitutionGroup="oa:Invoice"/>
```

Fig. 16. Overlay extension example.

```
In the CodeLists.xsd:

        <xsd:simpleType name="PaymentMethodCodeEnumerationType">
                <xsd:restriction base="xsd:normalizedString">
                        <xsd:enumeration value="Cash"/>
                        <xsd:enumeration value="Cheque"/>
                        <xsd:enumeration value="CreditCard"/>
                        <xsd:enumeration value="DebitCard"/>
                        <xsd:enumeration value="ElectronicFundsTransfer"/>
                        <xsd:enumeration value="ProcurementCard"/>
                        <xsd:enumeration value="BankDraft"/>
                        <xsd:enumeration value="PurchaseOrder"/>
                        <xsd:enumeration value="CreditTransfer"/>
                </xsd:restriction>
        </xsd:simpleType>

        <xsd:simpleType name="PaymentMethodCodeContentType">
                <xsd:union memberTypes="PaymentMethodCodeEnumerationType xsd:normalizedString"/>
        </xsd:simpleType>

In the Fields.xsd:

        <xsd:simpleType name="PaymentMethodCodeContentType">
                <xsd:restriction base="oacl:PaymentMethodCodeContentType"/>
        </xsd:simpleType>
```

Fig. 17. Code list example.

words, with the specification of “xsd:normalizedString” any code can be inserted into a BOD XML instance without affecting the validity against the BOD schema. For example, as presented in Figure 17, the “PaymentMethodCodeContentType” Field is associated with “oacl:PaymentMethodCodeContentType,” which is the union of “PaymentMethodCodeEnumerationType” and “xsd:normalizedString.” The use of “xsd:normalizedString” allows the users to send codes that are not listed in “PaymentMethodCodeEnumerationType.”

6. GLOBAL STANDARDS ONE

Global Standards One (GS1) is a family of standards focusing on different aspects of supply chain integration such as electronic products codes, product information synchronization, and the electronic document standards. GS1 was formed in early 2005 by the European Article Number (EAN) and the Uniform Commercial Code (UCC) organizations when they joined together. EAN and UCC were two organizations that heavily contributed to the adoption and proliferation of barcodes.

The part addressing the electronic document interoperability in this family of standards is GS1 eCom. In GS1 eCom, there are two distinct categories: the earlier

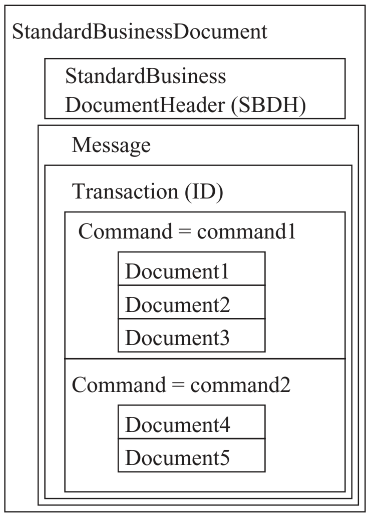


Fig. 18. The structure of GS1 XML structure.

eCom standards based on Electronic Document Interchange (EDI), called *EANcom* (EANCOM), and the newer generation GS1 XML (GS1 XML), which is defined using XML schema.

The other standards in GS1 family include the Global Data Synchronization Network (GDSN), and EPCglobal (EPCglobal). The Global Data Synchronization Network enables product data and location information synchronization so that trading partners have consistent item data in their respective systems.

EPCglobal drives the development of the Electronic Product Codes (EPCs) related to RFID standards. The specifications are based on the Radio Frequency Identification (RFID) research performed at the MIT AutoID Labs (MIT-AutoID).

6.1. GS1 XML

As shown in Figure 18, a GS1 XML document is represented with a *Standard-BusinessDocument*, which contains a *StandardBusinessDocumentHeader* (SBDH) and a *Message*. StandardBusinessDocumentHeader is based on the SBDH defined by UN/CEFACT (UN/SBDH) and provides information about the routing and processing of the XML instance document contained in the GS1 XML Message. The SBDH is used for the same purpose as OAGIS's ApplicationArea element; that is, it contains the configuration information for the transport software to send the message to its destination.

A GS1 XML document includes either a set of Commands or a set of Transactions which in turn contain Commands.

—*Command*. A *Command* instructs the recipient to perform a particular action, such as "Add," "Delete," and "Refresh," related to the documents within the command. The use of these commands decreases the number of documents needed. The same document can be used with different commands. Hence, no separate documents like "Add Order", "Change Order," or "Delete Order" are needed; the same "Order" document can be sent with a relevant command. In a similar way, several documents can reuse the same command.

—*Transaction*. A *Transaction* provides the functionality of executing multiple commands atomically as in relational databases. If one command in a transaction fails, the transaction fails, causing all other commands in the transaction to be discarded applying the principle of "all or nothing."

As an example, assume that a sender needs to send a message about two products and the first product is related to the second one. Instead of sending two distinct transmissions, the sender can transmit them together in one Transaction that contains one Command, which holds two Documents each of which is for a product. If the products are not related, the sender can send them without using the Transaction element. In other words, the user sends only one Command containing two Documents.

GS1 XML is compliant with UN/CEFACT CCTS methodology in that GS1 XML uses the same modeling, design, and technical principles. However, unlike UBL or OAGIS, which use UN/CEFACT artifacts (such as Core Component Types, Data Types, and Business Information Entities), GS1 XML does not use UN/CEFACT CCTS artifacts in their XML Schemas. Yet the GS1 Core Components are submitted as an input to UN/CEFACT CCTS development.

While developing their e-business standards, GS1 uses its Global Data Dictionary (GS1 GDD) to store, reuse, and share common components and business definitions, and their corresponding representations in XML. In other words, the GDD is the repository of the following:

—data components, used to create the GS1 XML standards, developed according to the UN/CEFACT Core Components Technical Specification (CCTS), and
—business terms and their representation in GS1 XML.

Through GDD, the search of previously defined components is facilitated.

In the GS1 XML documents, some of the components such as *Measurement*, *DocumentStatus*, and *MontetaryAmount* are common to more than one business document and more than one context. Therefore, these components are included in a common library as a part of the GDD. This approach allows reusing the same information constructs in all business messages.

6.1.1. Customization and Extensibility. In GS1 XML, the following context categories are defined for customization:

—The *Business Process Context*, in which collaboration takes place such as ordering or delivery.
—The *Industry Sector Context*, in which the business partners are involved, such as automotive.
—The *Geopolitical Context*, reflecting the geographical factors that influence the business semantics. This can be either country-specific, for example, only for France or Sweden, or limited to certain economic regions, for example, NAFTA or the European Union, and, finally, it can be applicable everywhere in the world, in which case the context is defined as "Global."

The context information is reflected in the documents through their namespaces. In other words, the GS1 information components are assigned to a namespace that reflects the context they are defined in. For example, the namespace for the documents that are used in the Global Data Synchronization Network (GDSN) is "gdsn=urn:ean.ucc:gdsn:2." As another example, the documents for alignment of trade items in Sweden use "sw=urn:ean.ucc:align:sweden:2" as their namespace. On the other

```
<extension>
<gdsn:attributeValuePairExtension
  xsi:schemaLocation="urn:ean.ucc:2
  ../Schemas/AttributeValuePairExtensionProxy.xsd">
    <value name="packagingWeightValue">15</value>
    <value name="packagingWeightUnitOfMeasure">kg</value>
</gdsn:attributeValuePairExtension>
</extension>
```

Fig. 19. Attribute/Value Pair mechanism to populate extension area.

hand, the schemas in the common library have “eanucc=urn:ean.ucc:2” as their namespace, because they do not belong to any specific context.

GS1 XML supports extensibility of its document schemas. Starting from release 2.0, there is an element called an “extension” at the end of each business document XML schema where additional context-specific information not defined by GS1 XML can be inserted. This element is of type “xsd:any,” which allows the users to insert any XML data to the exchanged instance documents without changing the standard GS1 XML schema.

Before starting to exchange GS1 XML instances with other parties, each organization that requires additional elements in their documents publishes their extensions to the “Extended Attributes” section of the Global Data Dictionary Web site. When a sender wishes to send a message to a receiver, the sender first checks whether the receiver has an extension by consulting the GDD Web site. If there is an extension, the sender sends the message using the Attribute/Value Pair mechanism. The Attribute/Value Pair mechanism is a way to populate the “extension” area of a document. As an example, assuming that the receiver requires two additional elements, “packagingWeightValue” and “packagingWeightUnitOfMeasure,” the sender populates the “extension” area as shown in Figure 19.

6.2. The Use of Code Lists

In GS1 XML, there are two types of code lists, external and internal. External code lists are defined and maintained by other standard bodies outside GS1 XML. Example external code lists include the following:

—Country codes, ISO 3166-1:1997;

—Country subdivision codes, ISO 3166-2:1998;

—Currency codes, ISO 4217:2001

The external code lists are defined as “xsd:string” and restricted to an appropriate number of characters. Figure 20 shows an example for a “countryISOCode” element defined as type “xsd:string” whose length is three characters. However, GS1 XML does not import the code list values to the GS1 XML schemas because of copyright and maintenance issues. In other words, they are not enumerated in the GS1 XML schemas.

The internal code lists are those developed and maintained within the GS1 system. They are defined as “xsd:enumeration” and imported into the business document schema that uses them. Figure 21 provides an example internal coding list for payment method types used in GS1 XML. It should be noted that all of the possible values are enumerated in the provided XML schemas.

```
<xsd:complexType name="ISO3166_1CodeType">
  <xsd:sequence>
    <xsd:element name="countryISOCode">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="3"/>
          <xsd:minLength value="1"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Fig. 20. Example country code element.

```
<xsd:simpleType name="PaymentMethodListType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="BANK_CHEQUE">
    </xsd:enumeration>
    <xsd:enumeration value="CASH">
    </xsd:enumeration>
    <xsd:enumeration value="CERTIFIED_CHEQUE">
    </xsd:enumeration>
    <xsd:enumeration value="CHEQUE">
    </xsd:enumeration>
    <xsd:enumeration value="CREDIT_CARD">
    </xsd:enumeration>
    <xsd:enumeration value="LETTER_OF_CREDIT">
    </xsd:enumeration>
    ...
  </xsd:restriction>
</xsd:simpleType>
```

Fig. 21. Example payment method list element.

7. ANALYSIS OF THE ELECTRONIC BUSINESS DOCUMENT STANDARDS

In this section, the surveyed electronic document standards are analyzed with respect to their document design principles, how they handle customization and extensibility, their coverage of the other layers of interoperability, and their industry relevance.

7.1. The Document Design Principles

The document design principles involve the document artifacts used in composing the documents, the code lists used to convey the meaning of the values in the elements, and the use of XML namespaces. Furthermore, since all the document standards surveyed are based on UN/CEFACT CCTS, how this methodology is used in the design of the document schemas is also discussed. Table III summarizes the document design principles.

7.1.1. Document Artifacts and the Use of UN/CEFACT CCTS Methodology. The document artifacts used in EDI are "Interchange," "Message," "Segment," and "Element" (Section 2). Note that EDI is not based on the UN/CEFACT CCTS methodology. UBL 2.0 uses the CCTS methodology to generate the document artifacts. UBL 2.0 currently considers only the Business Process Context and identifies the BIEs and bases the type of their artifacts on UN/CEFACT Unqualified Data Types and Core Component Types. UN/CEFACT develops its own BIEs, Core Components, and Data Types and stores them at the UN Core Component Library (UN/CCL). OAGIS 9.0 uses some of the UN/CEFACT ABIEs in their Components and bases the types of its Fields on UN/CEFACT Unqualified Data Types and Core Component Types. GS1 XML uses the UN/CEFACT CCTS methodology to generate its own artifacts by using its Global Data Dictionary.

Table III. Document Design Principles

	Document artifacts	Use of CCTS methodology	Use of code lists	Use of namespaces	Naming and design rules
EDI	Interchange, Message, Segment, Element	Not used	UN/EDIFACT recommends a number of code lists; local and external codes are also allowed	Not used	UN/EDIFACT syntax rules (ISO 9735) or X12.5 and X12.6 syntax rules
UN/CCL	Uses CCTS-Based Document Artifacts such as Core Component Types and BIEs	Fully based on CCTS methodology	Defines five code lists: Country Codes, Subdivision Codes, Currency Codes, Binary Object Mime Codes, and Unit Codes	Syntax-independent	ISO 11179-5
UBL 2.0	Uses CCTS artifacts	Fully based on CCTS methodology	Through a common base type called *CodeType* “xsd:-normalized-String”	Mostly for document categorization	UBL 2.0 Naming and Design Rules
OAGIS 9.0	BODs, Applica-tionAreas, Nouns, Verbs, Components, Compounds, Fields	Fields are UDT and CCT based; some components are UN/CEFACT ABIE based	Defines two “xsd:simple-Types” for each coded fields	To identify the overlay extension elements	UN/CEFACT ATG2 Naming and Design Rules
GS1 XML	SBDH, Transactions, Commands, Documents	Uses the CCTS methodology to generate its own document artifacts	External code lists; internal code lists defined through “xsd:-enumeration”	Namespaces indicate document context	GS1 XML’s UML to XSD conversion rules

7.1.2. The Use of Code Lists. Code lists are important to uniquely convey the semantics of elements in electronic documents such as the country codes, currency codes, and the payment units. All of the surveyed document standards provide default code lists and allow them to be modified and/or extended to support local codes.

As shown in Table III, EDI provides codes for structuring of the message artifacts (e.g., segment codes). Furthermore, UN/EDIFACT recommends *ISO Country Code, Currency Code, Numerical Representation of Dates, Times, Periods of Time*, and *UN/LOCODE* (ISO Codes). EDI also allows implementers to convey their own local or external codes through the use of two data elements, 1131 (UN/EDIFACT 1131) and 3055 (UN/EDIFACT 3055).

UN/CEFACT defines five code lists: *Country Codes*, *Subdivision Codes*, *Currency Codes*, *BinaryObject Mime Codes* and *Unit Codes*.

UBL 2.0 uses *Currency Codes*, *BinaryObject Mime Codes* and *Unit Codes* from UN/CEFACT and enumerates them in its schemas to validate attribute values. The other code lists used in UBL are not enumerated in the schema expressions. Instead of enumerating the codes in the XSD schemas, UBL uses a common base type called *CodeType*, which is an extension of “xsd:normalizedString,” for all elements expressing values from code lists. As described in Section 4.1.3, UBL allows the users to implement their own local/external codes.

For use of code lists, OAGIS defines two “xsd:simpleType”s for each coded Field: (1) an enumeration type, which lists the codes to be used, and (2) a “xsd:simpleType,” which is a union of that enumeration type and the “xsd:normalizedString” as explained in Section 5.1.3. With this mechanism, the implementers can use their own local/external code lists.

In GS1 XML, there are two types of code lists, external and internal. External code lists are defined and maintained by other standard bodies outside GS1 XML. The internal code lists are those developed and maintained within the GS1 System. They are defined as “xsd:enumeration” and imported into the business document schema that use them as described in Section 6.2.

7.1.3. The Use of Namespaces. Generally, the namespaces in XML are used for avoiding name conflicts. The document standards make additional use of the namespace mechanism as follows: UBL achieves categorization of documents through namespaces, OAGIS identifies the extensions through namespaces (Section 5.1), and GS1 XML gives context to the both original documents and extended documents through the namespaces as described in Section 6.1.

7.1.4. Naming and Design Rules. The naming and design rules specify how to name and structure the artifacts, how to put relations between the artifacts, and how to use data types for the artifacts. UN/CCL uses ISO 11179 naming rules, which identify the artifacts in the Object Class, Property Term, and Representation Term formats described in Section 3. UBL 2.0 uses UBL 2.0 Naming and Design Rules, which are based on CCTS terms such as ABIE, ASBIE, and BBIE. Furthermore, these rules specify how to represent the artifacts such as ABIEs, ASBIEs, and BBIEs in XML schemas. For example, for every ABIE, a “xsd:complexType” must be defined and the name of this complexType must be in upper camel case (UCC) format (UCC capitalizes the first character of each word and compounds the name such as “AccountType”). OAGIS 9.0 applies naming and design rules based on Applied Technology Group XML Syntax (ATG2) Naming and Design Rules (NDR) (ATG 2-NDR). Note that UN/CEFACT ATG2 NDR is based on UBL 2.0 NDR.

GS1 XML first designs its information model in UML, before creating the corresponding XML schemas. GS1 XML uses its own UML-to-XSD conversion rules to generate its XML schemas and to name them.

7.1.5. Analysis of Document Design Principles. The differences with respect to document design principles as analyzed in this section result in considerable differences in document instances from different standards. As an example, in Figure 22, the OAGIS 9.0 “AddressBaseType” component and GS1 XML “NameAndAddressType” document elements are compared. As clear from Figure 22, there are differences in the element names, element positions, and structures as well as in the use of code lists.

7.2. Customization and Extensibility

Any document interoperability standard faces two challenges. First, the standard needs to be extensible to allow the definition of information not contained in the standard’s artifacts because no standard can contain all of the data needed in every environment. Second, to be able to address a particular constraint in a specific context, it should be possible to customize the standard’s artifacts according to a context.

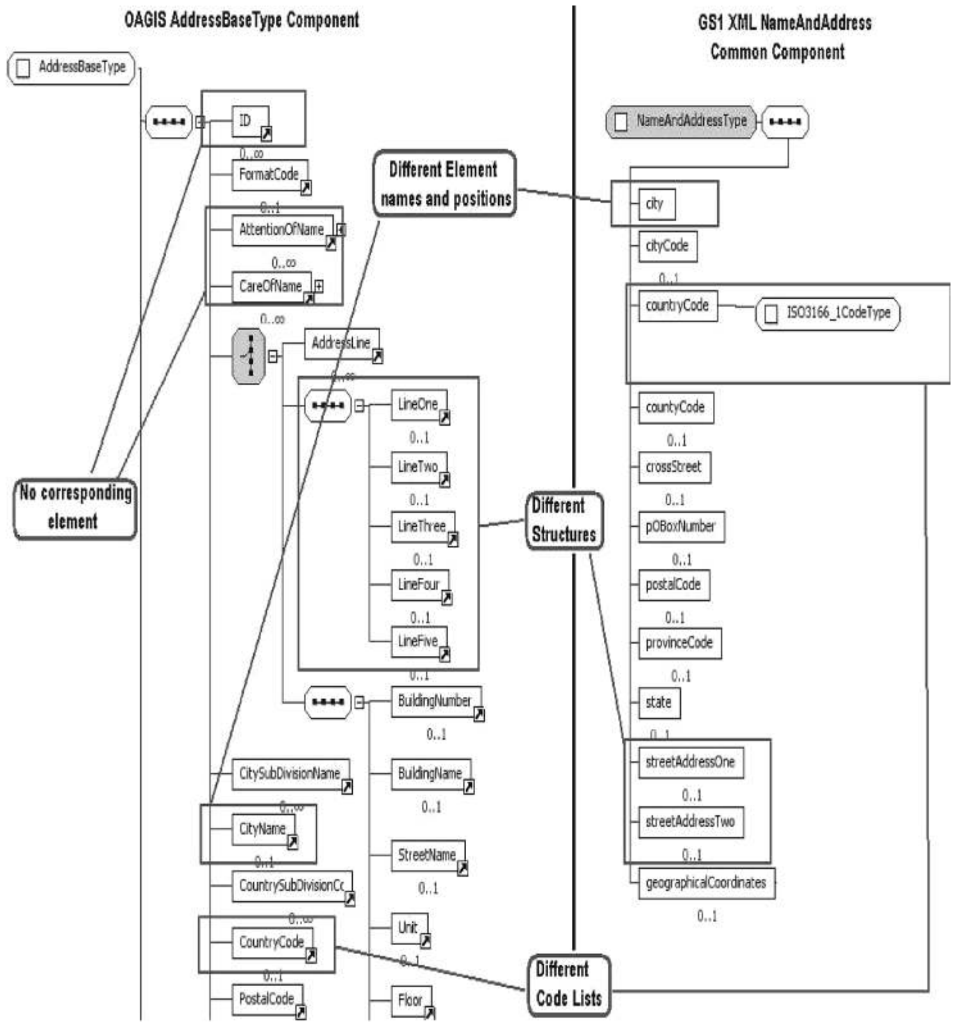


Fig. 22. An example comparing related parts of OAGIS BOD 9.0 and GS1 XML documents.

Table IV presents a summary of how the standards addressed in this article handle customization and extensibility. Note that conformant customizations are also extensible.

EDI addresses the customization through a subsetting mechanism to cover the requirements of a specific context. The EDI messages are subsetted first through industry Implementation Guides (IGs), which are then subsetted into trading partner IGs, and into departmental IGs.

Extensibility in EDI is difficult because the EDI systems are highly static and inflexible: introducing a new type or changing an existing type of business document is a complex process. Such changes require the modification of translation software and must be validated by the related EDI committees.

In UN/CEFACT CCTS, a Core Component is designed to be context-independent and is customized to one of the eight contexts defined by UN/CEFACT to become a Business

Table IV. Customization and Extensibility

	Customization	Extensibility
EDI	Subsetting EDI documents through context specific implementation guidelines.	Introduction of new types of business documents, which has to be validated through related EDI committees.
UN/CCL	Core Components are customized according to eight contexts to create BIEs.	New components can be published to the Core Component Library.
UBL 2.0	Conformant customization through "UBLExtensions" element, or subsetting or placing constraints on the value space.	Compatible customization by reusing the largest suitable aggregation from the UBL Library.
OAGIS 9.0	No formal methodology for defining user specific customizations	Through UserArea and overlay extensions.
GS1 XML	Through the following three contexts: Business Process, Industry Sector, Geopolitical	Through the "extension" element at the end of each document schema.

Information Entity. The possible business contexts that can be used are defined to be the *Business Process Context, Product Classification Context, Industry Classification Context, Geopolitical Context, Business Process Role Context, Supporting Role Context, Systems Capability Context*, and *Official Constraints Context*.

UN/CEFACT CCTS supports extensibility as follows: if users cannot find proper components in the *Core Component Library* to model their documents, they can create and publish new core components. In other words, UN/CEFACT CCTS thrives on extensibility by allowing users to define core components with possible future harmonizations and removal of redundancies.

UBL 2.0 allows customization through (1) the UBLExtensions element, (2) subsetting by removing optional information entities that are not needed, and (3) putting constraints to the elements as described in Section 4.1.1. On the other hand, the users can extend the UBL 2.0 schemas through the mechanisms described in Section 4.1.2.

In OAGIS BODs, there is no formal mechanism to handle user specific constraints. However, users are free to restrict an already existing BOD as they wish and share it with other partners.

OAGIS provides two mechanisms to extend its specifications, as detailed in Section 5.1.

—*UserArea Extensions*. UserArea Extensions provide an optional element within each OAGIS defined Component that may be used by an implementer to carry any necessary additional information. This area is of type "xsd:any," which means any valid XML instance can be inserted in this area without modifying the OAGIS standard XML schemas (XSDs).

—*Overlay Extensions*. Overlay Extensions allow users to extend an OAGIS BOD, Noun and Component to meet their own needs, even adding new BODs, Verbs, Nouns, and Components where necessary. It is also possible for users to provide additional constraints in their own XSL constraints, which may then be applied to OAGIS document instances. The overlay extension mechanism is used when the implementers have more complex customization requirements than a few additional elements.

Every document in GS1 XML is used in a business context, and, in GS1 XML, there are three context categories: *Business Process*, *Industry Sector* and *Geopolitical Contexts*, as described in Section 6.1.1.

GS1 XML supports extensibility of its document schemas. Starting from release 2.0, there is an element called an "extension" at the end of each business document XML schema where additional context-specific information not defined by GS1 XML can be inserted. This element is of type "xsd:any," which allows the users to insert any XML data to the exchanged instance documents without changing their standard XML schema.

Before starting to exchange GS1 XML instances with other parties, each organization that requires additional elements in their documents publishes its extensions to the "Extended Attributes" section of the Global Data Dictionary (GDD) Web site. When a sender wishes to send a message to a receiver, the sender first checks whether the receiver has an extension by consulting the GDD Web site.

7.2.1. Analysis of Customization and Extensibility. Customization and extensibility affect how the documents are processed. There are two cases to be considered:

—In the first case, if the parties use the same document schema with the same extensions and customizations, a two-phase validation at the receiving end is applied: in the first phase, the incoming document instance is validated against the common XSD schema. If the document instance passes the first phase, in the second phase it is checked against the rules, which specify additional domain specific constraints on the values of the elements in the instance. Generally, the rules are specified through XSL or Schematron languages. If the instance passes both of the phases successfully, it is delivered to the processing business application.

—In the second case, when two enterprises use different customizations or extensions of the same document schema, the schema changes need to be mapped to each other through manually provided XSL transformations. For instance, Figure 23 shows the XSL transformations necessary to map between two different example Overlay Extensions in OAGIS BODs. A classification of problems and solutions using XSL transformations to convert business documents is given in [Würstner 2002].

Once the transformations are applied, the document instance goes through the two-phase validation as described for the first case.

7.3. Coverage of Other Layers of Interoperability

Document interoperability is only one of the layers in the interoperability stack. The other layers of interoperability include the transport protocol, the message header and the business processes. A detailed survey of business-to-business (B2B) interactions in general is given in Medjahed et al. [2003], where a survey of the main techniques, systems, products, and standards for B2B interactions are presented together with a set of criteria for assessing them.

The standards covered in this survey do not enforce any specific transport protocol. However, some of them recommend certain transport protocols: GS1 XML recommends the use of the EDIINT AS1 (EDIINT-AS1) and AS2 (EDIINT-AS2) transport protocols, which define a minimum set of parameters and options to enable secure/reliable transport for the exchange of EDI or XML data. EDIINT-AS1 is based on SMTP and EDIINT-AS2 is based on HTTP. Between them, AS2 is the transport protocol of choice. However, the exchange of GS1 XML documents is not limited to these standards. OAGIS is currently moving in the Web service technology direction, although any technology can be used to transport BODs.

The document standards first analyze the relevant business processes or scenarios before deciding on the document components. For example, through the analysis of an invoicing business process, it may be revealed that a component is necessary to

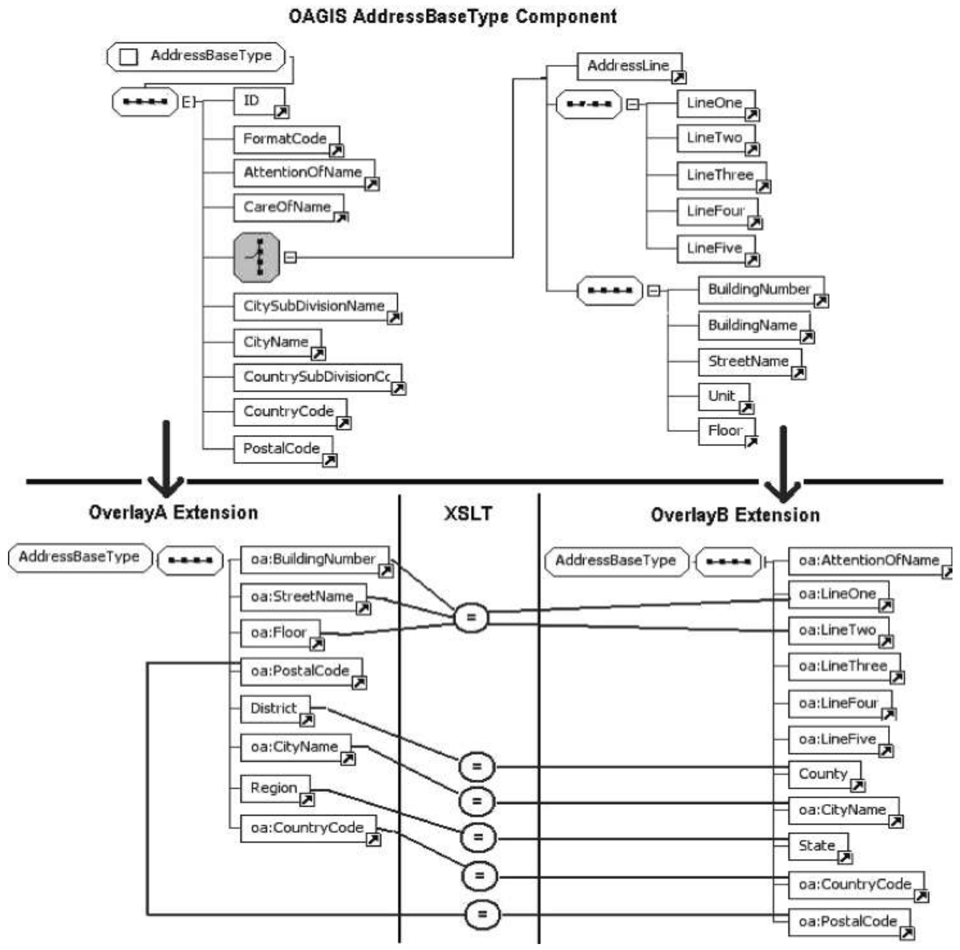


Fig. 23. Example XSL transformations necessary to map between two different overlay extensions in OAGIS BODs.

represent the "tax amount" in the invoice. Hence, "Tax Amount" is defined as a component that can be discovered and reused in any business document. However, no formal business process specification is provided by the standards surveyed in this article. Yet it's worth mentioning that there is work, called the Universal Business Process (UBP), for defining UBL 1.0 processes through ebBP 2.0 (ebBP); however, currently it is only informative.

All of the standards (except for UBL and UN/CCL) provide message header information to be conveyed to the transport protocol header. The EDIFACT message headers are the Interchange Control Header Segment, UNB (ICHS), and the X12 Interchange Control Header, ISA (ICH). The ApplicationArea in an OAGIS BOD is used to convey configuration information from application software to transport software. The GS1 XML *StandardBusinessDocumentHeader* (SBDH) carries transport-related information from application software to transport software just as in the case of OAGIS ApplicationArea.

7.3.1. Analysis of Layers of Interoperability Addressed. The surveyed standards do not specify a transport protocol but provide configuration information for the transport protocol message header.

Refraining from specifying other levels of interoperability has the advantage that it allows a wide variety of implementation techniques to be used and hence provides ease of implementation. However, the differences in the implementation techniques may cause interoperability problems.

7.4. Industry Relevance

EDI, being an early horizontal standard, is still used in several industry domains. For example, financial and monetary systems like the *Society for Worldwide Interbank Financial Telecommunication* (SWIFT) and *Electronic Funds Transfer* (EFT) use EDI. Furthermore, all airplane booking and ticketing operations are done over EDIFACT through the *International Air Transport Association* system (IATA).

Contrary to popular belief, electronic business interoperability is still achieved heavily through EDI-based messages, and EDI use is growing 3 to 5% every year (Vollmer). It seems large organizations will continue to use EDI for the foreseeable future mostly due to the existing infrastructure investments.

UN/CEFACT CCTS is gaining widespread adoption by standards organizations. As already mentioned, a number of standardization efforts have taken up CCTS methodology, including UBL, GS1 XML, OAGIS, CIDX, and PIDX in addition to UN/CEFACT's own Core Component Library (CCL).

The merits of CCTS for improving interoperability have also been noticed by industry and governments. For example, the German government has made a formal announcement identifying CCTS as the future data standard for domestic affairs (Crawford).

One of the first companies to support the UN/CEFACT CCTS methodology and core components in their products is SAP (SAP). SAP Global Data Types (GDTs) form the basis of business objects and enterprise services. All leaf elements of these SAP GDTs are based on Core Component Types and Data Types (Stuhec; Stuhec2).

UBL has been adopted by several communities around the world, especially in electronic government applications. The U.S. Department of the Navy (DON) designed its XML Naming and Design Rules around UBL 2.0 NDR.

The first government to use the UBL invoice was Denmark. The use of the UBL invoice is realized through the *Offentlig Information Online UBL* (OIOUBL) project and has been mandated by law for all public-sector businesses (OIOUBL) in Denmark. Also in Sweden, the National Financial Management Authority recommended the UBL invoice be customized to Sweden, namely, Svefaktura, for all government use (Svefaktura).

Following the success of Danish and Swedish examples, representatives from Denmark, Norway, Sweden, the U.K., Finland, and Iceland have created a Northern European Subset [NES] for UBL to ensure interoperability among these countries.

In the U.S., the Department of Transportation has developed a UBL-based pilot project for a demonstration of state-of-the-art electronic commerce in a real-world setting (US/DOT).

OAGIS BODs are being used in more than 40 countries and in more than 38 industries (OAGIS-Usage). The fact that OAGIS allows BODs to be extended by a vertical industry helps with its extensive use. The vertical standards based on OAGIS BODs include AiAG, Odette (ODETTE), STAR (STAR), and Aftermarket (AAIA) in the automotive industry. Other standards bodies focused on the human resources, chemical, and aerospace industries also use OAGIS BODs.

There are products based on OAGIS BODs such as Oracle E-Business Suite (ORACLE), where OAGIS BODs are implemented as Web services. As another example, the IBM WebSphere Commerce service interfaces are defined using the OAGIS message structure (ROWELL).

GS1 XML is being used in more than 20 countries and in more than 20 industries all over the world. GS1 is a business solution partner of many companies, including Oracle, Siemens, and Philips. The GS1 standards are also leveraged in SAP business solutions packages.

8. CONCLUSIONS

Today, an enterprise's competitiveness is to a large extent determined by its ability to seamlessly interoperate with others, and electronic document standards play an important role in this.

Although all the document standards surveyed in this article (with the exception of EDI) are based on the UN/CEFACT CCTS methodology, their analysis reveals that there are considerable differences in the resulting document schemas. This is mostly because the standards like OAGIS BODs and GS1 XML existed long before the UN/CEFACT CCTS methodology was proposed, and therefore these standards adapted their existing document schemas rather than starting afresh. However, all of these standards are still being developed, and their future versions may become more harmonized.

In fact, by observing that the divergent and competing approaches to electronic document standardization threatens intersectoral coherence in the field of electronic business, four major standard bodies, namely, the International Electrotechnical Commission (IEC), the International Organization for Standardization (ISO), the International Telecommunication Union (ITU), and the United Nations Economic Commission for Europe (UNECE) signed a Memorandum of Understanding to specify a framework of cooperation (MoU). In the year 2000, they established a Memorandum of Understanding Meeting Group for eBusiness standards harmonization. Up to now, OAGIS 9.0 and UBL 2.0 have achieved a degree of harmonization in that they are based on the same UN/CEFACT unqualified Data Types and Core Component Types. However, the harmonization needs to be extended to the upper-level artifacts such as the BBIEs and the ABIEs.

Currently, transforming an electronic business document from one standard format into another is generally achieved by means of Extensible Stylesheet Language (XSL) using schema matching techniques as described in Rahm and Bernstein [2001].

An alternative emerging approach to document interoperability is the semantic mediation of the electronic document schemas. Yarimagan1 et al. [2007a] have argued that providing syntactic interoperability among document schemas based on XSL transformations or Schematron alone is not enough. Syntactic interoperability needs to be supported by semantic interoperability, that is, it must be possible for automated processes to discover and reuse customizations provided by other users. For this purpose, the authors described how the semantic representations of the context domains are provided and how this semantics is utilized by automated processes for component discovery and schema customization in UBL.

In Yarimagan2 [2007b], a component ontology for UBL is developed by using the Web Ontology Language (OWL) to represent the semantics of individual components and their relationships within customized schemas. Then this ontology is processed through description logic reasoners for the discovery of similar components and the automation of the translation process among different UBL customizations.

In Anicic [2005], semantic Web technologies are used to transform documents between two vertical industry standards both based on OAGIS: one conforming to

Standards in Automotive Retail (STAR) schemas and the other conforming to *Automotive Industry Action Group* (AiAG) schemas. First, the STAR and AiAG XML schemas are converted to Web Ontology Language. Then these independently developed ontologies are merged. By using the merged ontology, the STAR document instances are converted to the corresponding AiAG documents and vice versa.

In Ye et al. [2007], a supply chain management ontology, called *Onto-SCM*, is developed, which represents a common semantic model of supply chain management. The authors then showed how Onto-SCM can be used for converting document schemas of different standards.

As a final word, although the electronic document standards developed so far have proven to be very useful for industry and government applications, further efforts are needed to increase their harmonization and semantic interoperability.

REFERENCES

Anicic, N. and Ivezic, N. 2005. Semantic Web technologies for enterprise application integration. *Comput. Sci. Inform. Syst. 1*, 119–144.

Brown, N. and Reynolds, M. 2000. Strategy for production and maintenance of standards for interoperability within and between service departments and other healthcare domains. Short strategic study. CEN/TC 251/N00-047. CEN/TC Health Informatics, Brussels, Belgium.

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

Medjahed, B., Benatallah, B., Bouguetta, A., Ngu, A. H. H., Elmagarmid, A. K. 2003. Business-to-business interactions: Issues and enabling technologies. *VLDB. J. 12*, 1, (May), 59–85.

Rahm, E. and Bernstein, P. A. 2001. A survey of approaches to automatic schema matching. *VLDB J. 10*, 4, 334–350.

Würstner, E., Hotzel, T., and Buxmann, P. 2002. Converting business documents: A classification of problems and solutions using XML/XSLT. In *Proceedings of the 4th IEEE International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems* (WECWIS).

Yarimagan, Y. and Dogac, A. 2007a. Semantics based customization of UBL document schemas. *J. Distrib. Parall. Databases. 22*, 2-3 (Dec.) 107–131.

Yarimagan, Y. and Dogac, A. 2007b. A semantic based solution for the interoperability of UBL schemas. `http://www.srdc.metu.edu.tr/webpage/publications/2007/YarimaganDogac-UB% LTranslationPaper.pdf`.

Ye, Y., Yang, D., Jiang, Z., and Tong, L. 2007. Ontology-based semantic models for supply chain management. *Int. J. Adv. Manufact. Tech.* (online) May.

Received December 2007; revised May 2008; accepted November 2008