A Survey and Analysis of Electronic Business Document Standards

Yildiray Kabak and Asuman Dogac
Dept. of Computer Eng.
Middle East Technical University (METU)

The development of electronic document interoperability standards has been evolutionary based on the traditional EDI technology and affected both by the technological developments such as the Internet and XML and also by the dynamic interoperability needs of the current eBusiness applications.

No document standard is sufficient for all purposes because the requirements significantly differ amongst businesses, industries and geo-political regions. On the other hand, the ultimate aim of business document interoperability is to exchange business data among partners without any prior agreements related with the document syntax and semantics. Therefore, an important characteristic of a document standard is its ability to adapt to different contexts, its extensibility and customization. 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 [**Computer Applications**]: Administrative Data Processing—*Business*; J.1 [**Computer Applications**]: Administrative Data Processing—*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

Authors' addresses: Y. Kabak and A. Dogac, Dept. of Computer Eng., Software R&D Center, Middle East Technical University (METU), 06531, Ankara, Turkey, e-mail: {yildiray, asuman}@srdc.metu.edu.tr.

This work is supported by the European Commission through the IST-213031 iSURF project and in part by the Scientific and Technical Research Council of Turkey (TÜBİTAK), Project No: EEEAG 105E068

1. INTRODUCTION

Interoperability of business applications can be investigated at three broad layers: communication layer, 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 invent of the Internet. The first standard developed was the Electronic Data Interchange (EDI) framework [EDI] where document exchange was realised through dialup connections using proprietary networks.

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) [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. In the mean time, there are 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* [HL7], the *Petroleum Industry Data Exchange (PIDX) committee* [PIDX] the *Chemical Industry Data Exchange (CIDX) organization* [CIDX], Open *Travel Alliance* [OTA], and *RosettaNet Consortium* [RosettaNet] to name but a few.

The earlier standards have 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* 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 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.

—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.

CCTS is gaining widespread adoption by both the horizontal and the vertical standard groups. Universal Business Language (UBL) [UBL] is one of 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 processes 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 take up. 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 paper 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. 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 paper by describing harmonization efforts and the emerging semantic approach to the document standards interoperability. Since a large number of acronyms is introduced throughout the paper, a list of all acronyms and their meaning is provided in Table III.

2. ELECTRONIC DATA INTERCHANGE (EDI)

EDI is developed through two main branches: ANSI X12 and UN/EDIFACT. In the USA, the American National Standards Institute (ANSI) developed ANSI X12 [X12] and internationally EDI is standardised as UN/EDIFACT (United Nations/Electronic Data Interchange For Administration, Commerce, and Transport) [UN/EDIFACT]. 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 Organisation as ISO 9735 [UN/EDIFACT].

Fig. 1. The Basic EDI Architecture

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 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 with mid 1990s, there were several updates to the EDI architecture. First, the Internet protocol for email, 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 Hyper-Text Transfer Protocol (HTTP), was popularised, this became another mechanism for EDI document transfer.

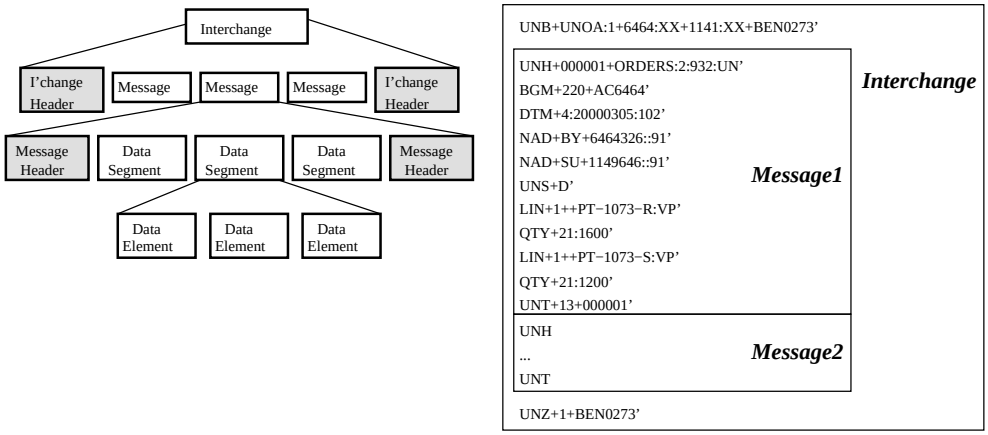

Fig. 2. (a) The Basic EDI Message Structure, (b) An Example EDI Message

3. UN/CEFACT CORE COMPONENT TECHNICAL SPECIFICATION (CCTS)

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*. 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* and *Data Types*.

3.1 Core Component Types and Data Types

Core Component Types (CCT) 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, CCT's 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 are shown in Figure 3 [CCTS]. Up to now, UN/CEFACT has approved 10 *Core Component Types* and defined 35 permissible *Data Types*, and undertook their maintenance. Furthermore, the *Data Types* provided by UN/CEFACT can be used without restrictions (*Unqualified Data Types (UDT)*) or further restricted (*Qualified Data Types (QDT)*) 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 semantic meaning. The Naming Convention used in CCTS is derived 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. Tax. Amount" is expressed according to the CCTS naming convention, "Invoice" is the *Object Class*, "Tax" 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*:

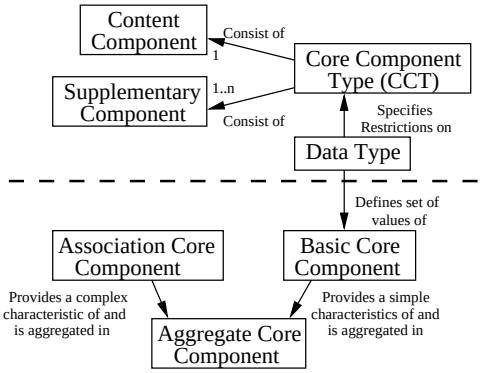

Fig. 3. Core Component Overview [CCTS]

—*Aggregate Core Component (ACC)*: A distinct real world object with a specific business meaning such as "Address" or "Purchase Order" is termed as an *Aggregate Core Component* (ACC). 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 *Aggregate Core Component (ACC)* containing several *Basic Core Components (BCCs)*.

—A *Basic Core Component* 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 (BCC)* and is of "Text" Data Type. In other words, the *Data Types* are used as *Representation Terms* of *Basic Core Components*.

—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* (ASCC) referencing the "Address. Details" ACC.

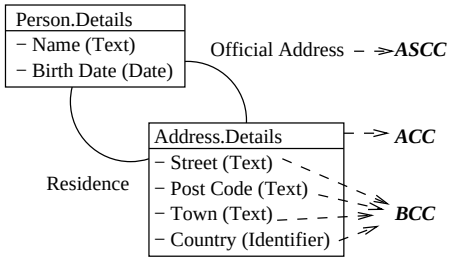

Fig. 4. Examples of *Basic Core Component* (BCC), *Aggregate Core Component* (ACC) and *Association Core Components* (ASCC) [CCTS]

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

Business Process Context

Trade

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 (BIE)

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: *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).*

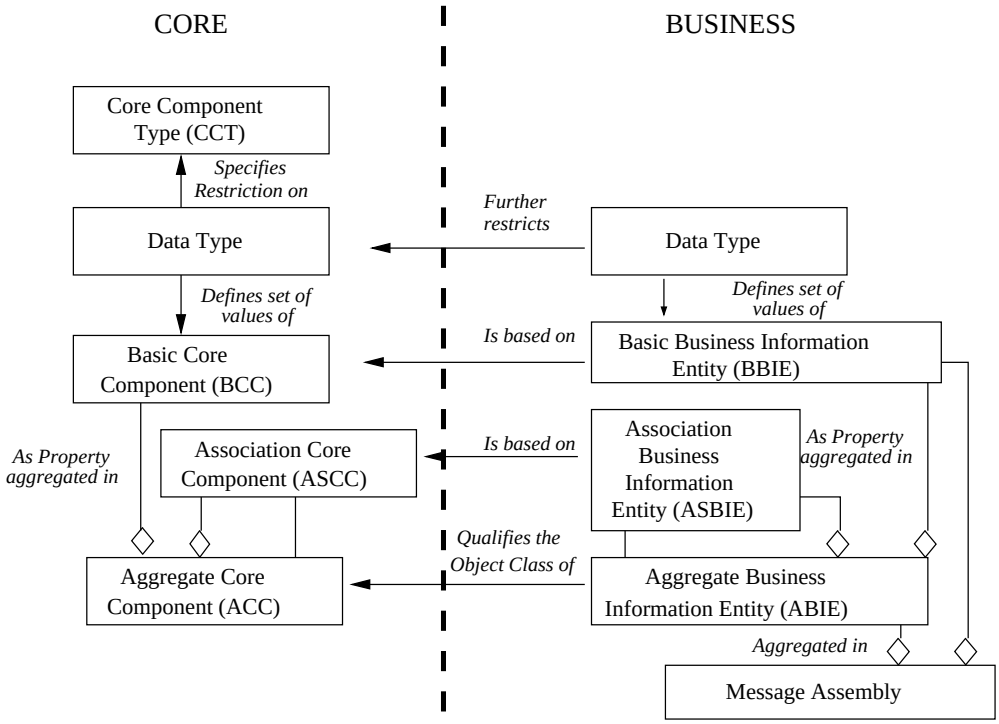

Fig. 6. Relationship between Core Components and Business Information Entities [CCTS]

Similarly, when an *Association Core Component* is used in a context, it becomes *Association Business Information Entity (ASBIE)* and *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 (UBL)

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.

Currently, the approved version of UBL is 2.0 [UBL] and there are thirty one 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) [UBLSchemas] 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

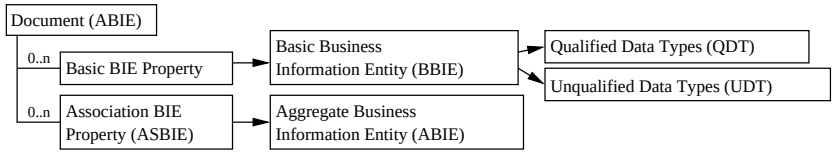

Fig. 7. The UBL Components

UN/CEFACT CCTS such as "AmountType", "CodeType" and "DateTimeType". When UN/CEFACT CCTS *Data Types* are imported to UBL type space, they are termed as the *Unqualified Data Types (UDT)*. Additionally, UBL defines *Qualified Data Types (QDT)* which are primarily for code lists such as *CurrencyCodeType* or *CountryIdentificationCodeType* defined for use within UBL.

At the time UBL initiative has started, UN/CEFACT CCTS has 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 UN/EDIFACT (EDI for Administration Commerce and Trade) dictionary [UN/EDIFACT]. 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 *Business Information Entities* (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, there are two types of ABIEs: (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 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*. 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. Note that there is no corresponding concept for *Basic BIE Property* in the UN/CEFACT CCTS. With this artifact, UBL diverges from UN/CEFACT CCTS Methodology where BBIEs are specialized from *Basic Core Components*. However 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). Figure 8 shows an example UBL 2.0 "Order" document.

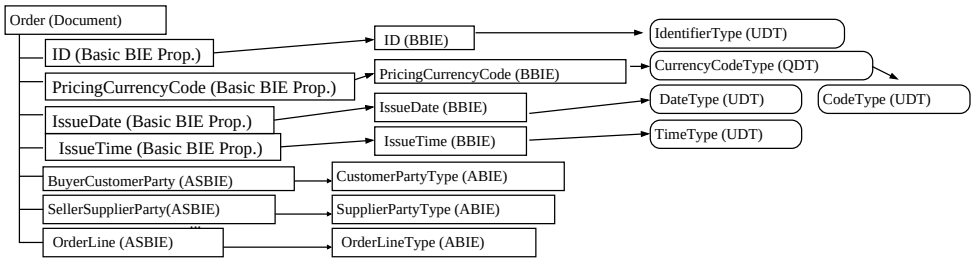

Fig. 8. An Example UBL Document Schema

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 details of customization, it is worth mentioning about the validation of UBL documents. UBL 2.0 adopts 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 XSL [XSL] or Schematron languages [Schematron]. If the instance passes both of the phases successfully, it is delivered to the processing business application.

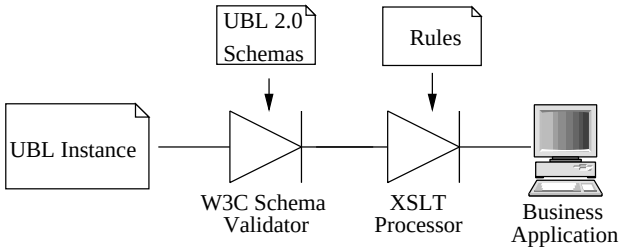

Fig. 9. Two-phase validation of UBL Messages

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 UBL 2.0 schemas.

There are four ways of performing conformant customizations:

(1) *Inserting additional elements through the use of "UBLExtensions" element*: An optional *UBLExtensions* element appears as the first child of all UBL 2.0 documents which 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. *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 *ExtensionContent* element without modifying the schema.

(2) *Subsetting original UBL 2.0 schemas*: There are very many elements in a UBL document. For example, there are about 50,000 elements in a UBL Order Document. Some applications may not need all this data. Therefore, UBL 2.0 allows the users to create subsets of its documents. 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 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, the users can describe these constraints through Schematron [Schematron] or XSL rules [XSL] 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. The 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, compatible customization approach can be used. In compatible customization, the users modify an existing UBL 2.0 schema or create a new one by re-using 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.

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 custom 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. UBL 2.0 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 (BOD) VERSION 9.0

The Open Applications Group, Inc. (OAGi) [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 [OAGIS]. This version is redesigned to be based on the UN/CEFACT Core Components Technical Specification.

The *Business Object Document* (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.

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 enough 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, it is 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".

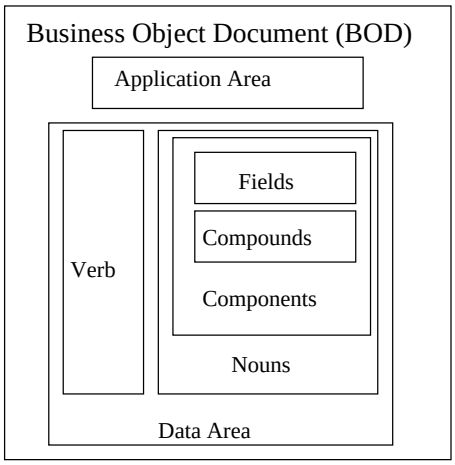

Fig. 10. The Structure of OAGIS Business Object Document (BOD)

As shown in Figure 10, BOD is a message structure composed of an *ApplicationArea* and a *DataArea*. The *ApplicationArea* carries necessary information for a 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 11 shows an example BOD assembly with OAGIS artifacts.

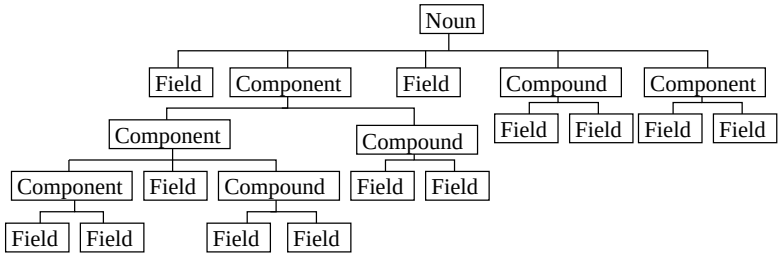

Fig. 11. OAGIS Business Object Document (BOD) assembly example

OAGIS implementation of the Core Component Technical Specification (CCTS) is shown in Figure 12. 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 12 OAGIS, rather than using the UN/CEFACT ABIEs directly, it incorporates them into OAGIS *Components*. When using these ABIEs in their *Components*, OAGIS appends "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 UN/CEFACT ATG2 Naming and Design Rules [ATG2-NDR].

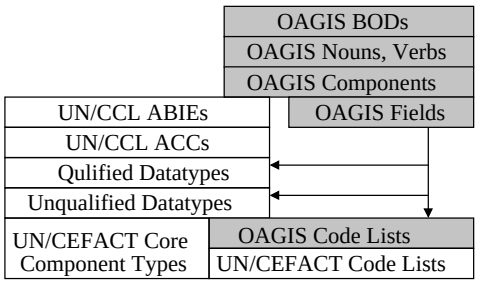

Fig. 12. OAGIS usage of UN/CEFACT CCT

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, "Address" *Component* does not have such a *Field* to carry "Mahalle" information. This "Mahalle" information can be inserted in the *UserArea* part of "Address" *Component* in a BOD instance when it is used in Turkey, as shown in Figure 13.

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

Fig. 13. UserArea 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*, *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 14, an example for overlays is shown where "Automotive" overlay is created from core OAGIS schemas, whereas "Auto Parts" that is a subdomain of "Automotive", is built on "Automotive" and OAGIS core.

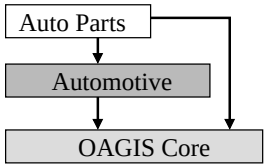

Fig. 14. OAGIS Overlay Layering Example

```
<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. 15. Overlay Extension Example

With *Overlay* extensions, the users are allowed to create a new BOD, a *Noun*, a *Component*, a *Compound* or a *Field*, or 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 15 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 "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.

```
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. 16. Code List Example

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" 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 words, with the specification of "xsd:normalizedString" any code can be inserted to a BOD XML instance without affecting the validity against the BOD Schema. For example, as presented in Figure 16, 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 (GS1)

Global Standards One (GS1) [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 is formed in the 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 eCom standards that are 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 (GDSN) 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 Code (EPC) related with 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 17, a GS1 XML document is represented with a *StandardBusinessDocument*, 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

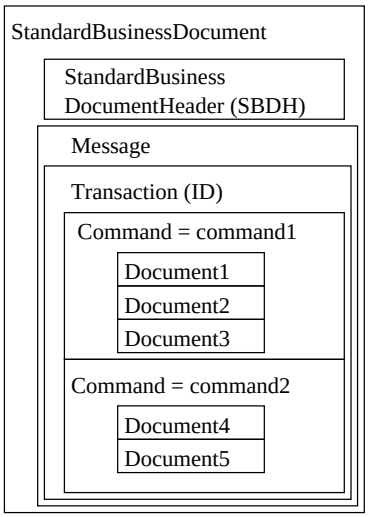

Fig. 17. The Structure of GS1 XML Structure

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 then, 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 modelling, 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 submit-

ted 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:

—Data components, used to create the GS1 XML standards, developed according to the UN/CEFACT Core Components Technical Specification (CCTS).

—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:

—*Business Process* context in which collaboration takes place such as ordering or delivery.

—*Industry Sector* context in which the business partners are involved such as automotive.

—*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 European Union, and finally, it can be applicable everywhere in the world in this case the context is defined as "Global".

The context information is reflected to the documents through their namespaces. In other words, the GS1 information components are assigned to a namespace that reflects the context it is defined in. For example, the namespace for the documents that are used in 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 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 "extension" at the end of each business document XML schema where additional context-specific information that are 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 Attribute/Value Pair mechanism. 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 18.

```
<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. 18. Attribute/Value Pair Mechanism to populate extension area

```
<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. 19. Example Country Code Element

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. The 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 19 shows an example for "countryISOCode" element which is defined of 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 the 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 it. Figure 20 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: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. 20. 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 I 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 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 *Business Information Entities* (BIEs) and bases the type of their artifacts to UN/CEFACT *Unqualified Datatypes* and *Core Component Types*. The UN/CEFACT develops its own BIEs, *Core Components* and *Datatypes* 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* to UN/CEFACT *Unqualified Datatypes* (UDT) and *Core Component Types* (CCTs). GS1 XML uses the UN/CEFACT CCTS methodology to generate its own artifacts by using its Global Data Dictionary.

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.

Table I. Document Design Principles

	Document Artifacts	**Use of CCTS Methodology**	**Use of Codelists**	**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*, *BinaryObject Mime Codes* and *Unit Codes*	It is syntax-independent	ISO 11179-5
UBL 2.0	Uses CCTS Artifacts	Fully based on CCTS Methodology	Through a common base type called *CodeType* "xsd:-normalizedString"	Mostly for document categorization	UBL 2.0 Naming and Design Rules
OAGIS 9.0	BODs, Application Areas, Nouns, Verbs, Components, Compounds, Fields	Fields are UDT and CCT based. Some Components are UN/CEFACT ABIE based	Defines two "xsd:simpleType" for each coded *Field*	To identify the *Overlay* extension elements	UN/CEFACT ATG2 Naming and Design Rules
GS1 XML	SBDH, Transactions, Commands, Documents	Use the CCTS methodology to generate its own document artifacts	External Code Lists; Internal Code Lists defined through "xsd:-enumeration"	The namespaces indicate the document context	GS1 XML's UML to XSD conversion rules

As shown in Table I, 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. 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" 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 uses it 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 *Object Class*, *Property Term* and *Representation Term* format as described in Section 3. UBL 2.0 uses UBL 2.0 Naming and Design Rules, which are based on the CCTS terms such as ABIE, ASBIE and BBIE. OAGIS 9.0, on the other hand, applies naming and design rules based on Applied Technology Group XML Syntax (ATG2) Naming and Design Rules (NDR) [ATG2-NDR]. UN/CEFACT ATG2 NDR basically specifies 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").

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 their 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 results in considerable differences in document instances from different standards. As an example, in Figure 21, the OAGIS 9.0 "AddressBaseType" Component and GS1 XML "NameAndAddressType" document elements are compared. As it is clear from this figure, there are differences in the element names, the element positions and structures as well as in the use of code lists.

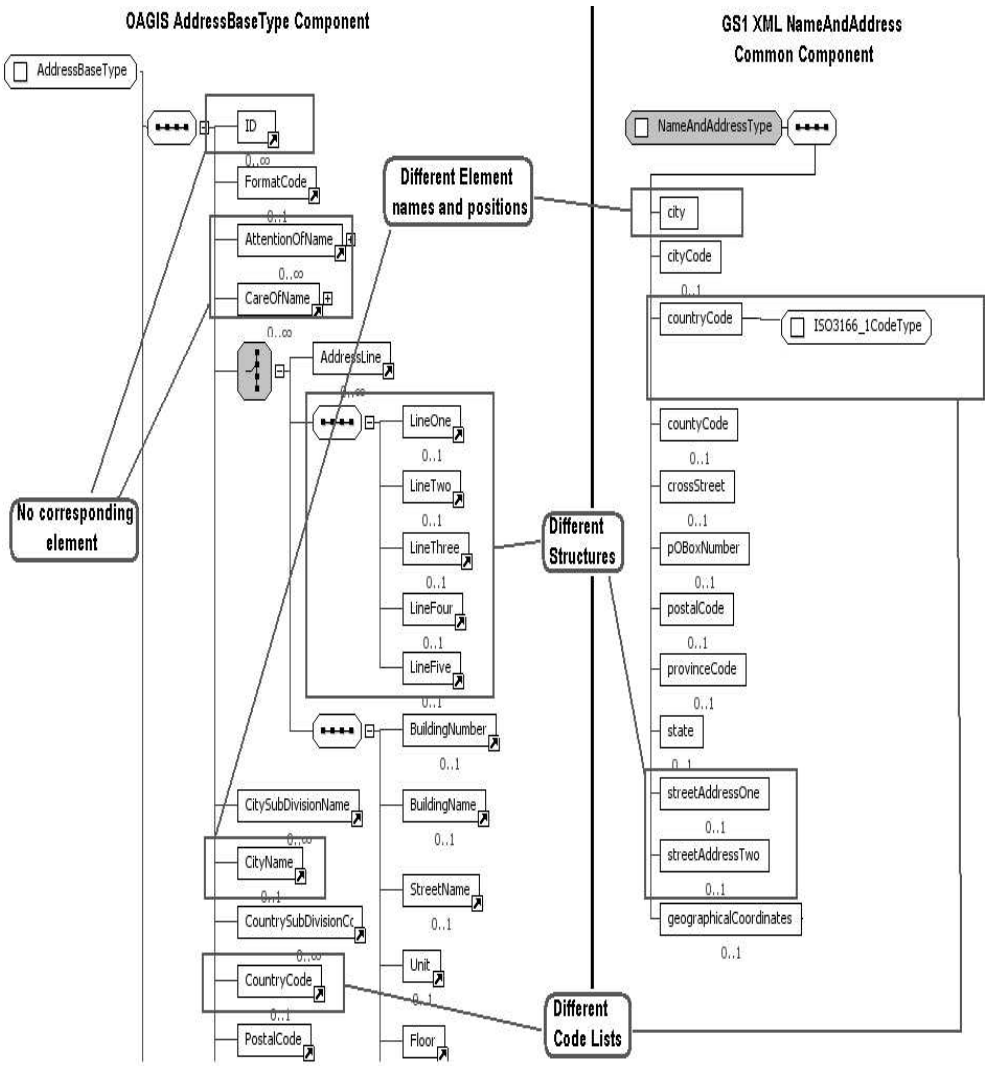

Fig. 21. An Example Comparing Related Parts of OAGIS BOD 9.0 and GS1 XML Documents

7.2 Customization and Extensibility

Any document interoperability standard faces two challenges. First, the standard needs to be extensible to allow definition of information that is not contained in the standard's artifacts because no standard can contain all of the data needed in every environment. Secondly, 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.

Customization and extensibility can be handled in two ways: by allowing changes in the standard document schema or without changing the document schema. Doc-

ument schema changes weaken interoperability.

Table II present a summary of how the standards addressed in this article handle customization and extensibility.

Table II. Customization and Extensibility

	Customization	**Extensibility**
EDI	Subsetting EDI documents through context specific Implementation Guidelines. No schema changes.	Introduction of new types of business documents which has to be validated through related EDI Committees. Schema changes
UN/CCL	Core Components are customized according to eight contexts to create BIEs. No schema changes.	New components can be published to the Core Component Library. Schema changes.
UBL 2.0	Conformant customization through "UBLExtensions" element, or subsetting or placing constraints on the value space. No schema changes.	Compatible customization by reusing the largest suitable aggregation from the UBL Library. Schema changes.
OAGIS 9.0	No formal methodology for defining user specific customizations	Through *User Area* and *Overlay* extensions. Schema changes in *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. No schema changes.

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 (IG), which are then subsetted into trading partner IGs, and into departmental IGs.

The extensibility in EDI is difficult because the EDI system is highly static and inflexible: the introduction of 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 in 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 Information Entity (BIE)*. The possible business contexts that can be used are defined to be: *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*.

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) *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, the users are free to restrict an already existing BOD as they wish and sharing 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 to 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 "extension" at the end of each business document XML schema where additional context-specific information that are 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 their extensions to the "Extended Attributes" section of 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.* The 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 [XSL] or Schematron languages [Schematron]. If the instance passes both of the phases successfully, it is delivered to the processing business application.

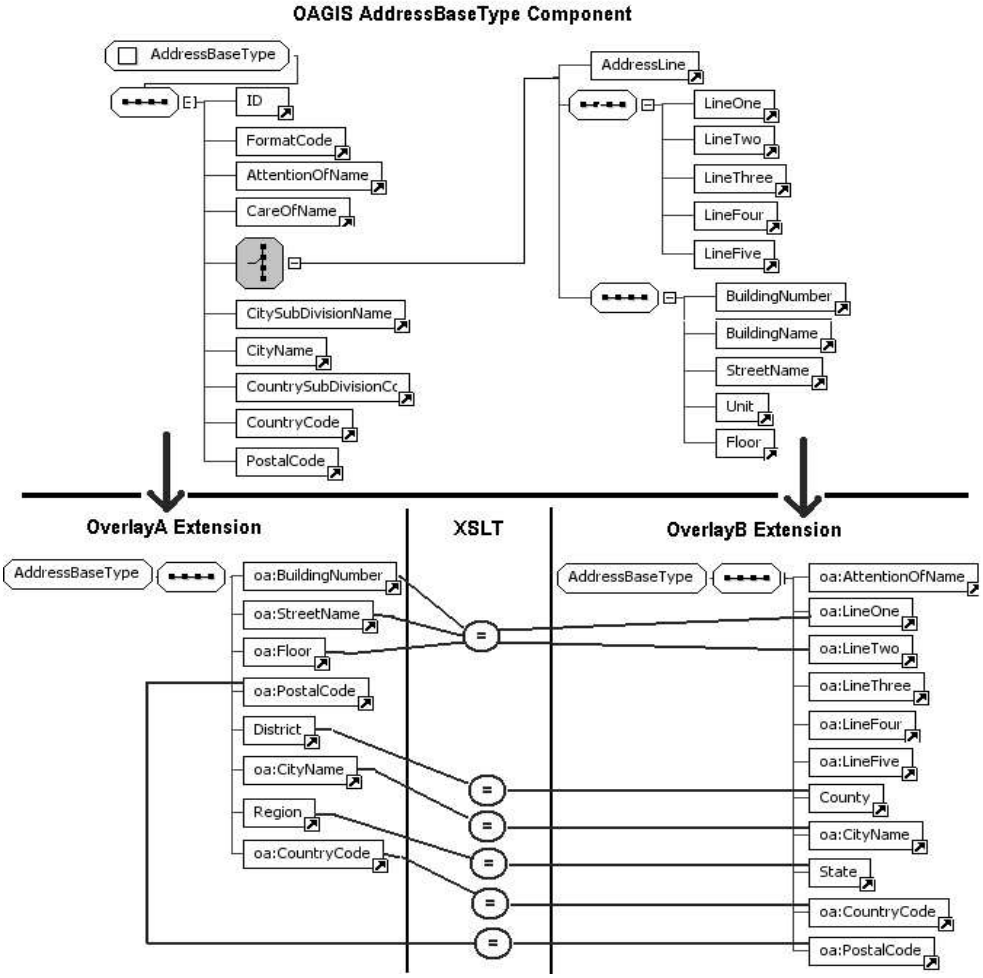

Fig. 22. Example XSL Transformations necessary to map between two different *Overlay* extensions in OAGIS BODs

—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 22 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 [Wüstner].

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 interactions in general is given in [Medjahed, et. al.].

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 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 upon SMTP and EDIINT-AS2 is based on HTTP. Among 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 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 worths mentioning that there is work, called 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 UN/CCL which is syntax independent) 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]. In UBL, the message header information to be conveyed to the transport protocol header is dispersed through out the document instances. The *Application Area* in an OAGIS BOD is used to convey configuration information from application software to transport software. GS1 XML *StandardBusinessDocumentHeader* (SBDH) carries transport related information from application software to the transport software just as in the case of OAGIS *Application Area*.

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 in implementation. However, the differences in the implementation techniques may cause interoperability problems.

7.4 Industry Relevance

EDI, being an early horizontal standard, is being used in several industry domains. For example, the financial and monetary systems like *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 the popular belief, electronic business interoperability is still achieved heavily through EDI based messages and EDI use is growing 3 to 5 percent 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 the 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 the industry and the governments. For example, the U.S. Department of the Navy (DON) designed their XML Naming and Design Rules around CCTS. 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 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 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)" Project and has been mandated by law for all public-sector businesses [OIOUBL] in Denmark. Also in Sweden, the National Financial Management Authority recommended UBL Invoice customized to Sweden, namely, Svefaktura for all government use [Svefaktura].

Following the success of Danish and Swedish examples, representatives from Denmark, Norway, Sweden, UK, Finland and Iceland have set up a working group to develop a Northern European Subset (NES) [NES] for UBL to ensure interoperability among these countries.

In the USA, the Department of Transportation is developing 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 forty countries in more than thirty eight 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 [AiAG], Odette [ODETTE], STAR [STAR], and Aftermarket [AAIA] in the automotive industry. Other standard bodies focused on 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, IBM WebSphere Commerce service interfaces are defined using the OAGIS message structure [Rowell].

GS1 XML is being used in more than twenty countries in more than twenty 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 [SAP].

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 UN/CEFACT CCTS Methodology, their analysis reveal 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 UN/CEFACT CCTS Methodology is proposed and therefore these standards adapted their existing document schemas rather than starting from fresh. However, all of these standards are still developing 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 level of harmonization in that they are based on the same UN/CEFACT *Unqualified Datatypes* and *Core Component Types*. However, the harmonization needs to be extended to the upper level artifacts such as the BBIEs and the ABIEs.

An alternative emerging approach to document interoperability is the semantic mediation of the electronic document schemas.

[Yarimagan1] argues that providing syntactic interoperability among document schemas based on XSL transformations or Schematrons 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 describe 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], 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], Semantic Web technologies are used to transform documents between two vertical industry standards both based on OAGIS: one conforming to the *Standards in Automotive Retail* [STAR] and the other conforming to the *Automotive Industry Action Group* [AiAG]. First, the STAR and AiAG XML Schemas are converted to Web Ontology Language [OWL]. Then these independently devel-

oped ontologies are merged. By using the merged ontology, the STAR document instances are converted to the corresponding AiAG documents and vice versa.

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

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

Table III. 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
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 Types
CCTS	Core Components Technical Specification
CEFACT	Centre for Trade Facilitation and Electronic Business
CIDX	Chemical Industry Data Exchange
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 Types
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 Guides
ISO	International Organization for Standardization
IT	Information Technology
ITU	International Telecommunication Union

continued on next page

continued from previous page	
MIME	Multipurpose Internet Mail Extensions
NDR	Naming and Design Rules
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 Types
RFID	Radio Frequency Identification
SBDH	Standard Business Document Header
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 Types
UMM	UN/CEFACT Modelling 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

REFERENCES

AAIA. Automotive Aftermarket Industry Association. http://www.aftermarket.org/Home.asp.

AiAG. Automotive Industry Action Group. http://www.aiag.org/.

Anicic. N. Anicic, N. Ivezic, Semantic Web Technologies for Enterprise Application Integration. ComSIS, Computer Science and Information Systems, International Journal, Volume 02 , Issue 01, June 2005.

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/.

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.

Medjahed, et. al. B. Medjahed, B. Benatallah, A. Bouguettaya, A.H.H. Ngu, A.K. Elmagarmid, Business-to-Business Interactions: Issues and Enabling Technologies. The International Journal on Very Large Data Bases, Vol. 12, No. 1, May, 2003.

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/.

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%.

Wüstner. E. Wustner, T. Hotzel, P. Buxmann, Converting Business Documents: A Classification of Problems and Solutions using XML/XSLT. Proc. of the 4th IEEE Intl Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems (WECWIS), 2002.

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/.

Yarimagan1. Y. Yarimagan, A. Dogac - Semantics Based Customization of UBL Document Schemas. Journal of Distributed and Parallel Databases, Springer-Verlag, Volume 22, Numbers 2-3 / December 2007, pp. 107-131.

Yarimagan2. Y. Yarimagan, A. Dogac - 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. Ye, D. Yang, Z. Jiang, L. Tong - Ontology-based semantic models for supply chain management. The International Journal of Advanced Manufacturing Technology, Published online, Springer London, May 2007.

Received ; revised