



**TripCom**  
*Triple Space Communication*  
**FP6 – 027324**

Deliverable

**D7.4**  
**Evaluation and Refinement of Ontologies**

douglas foxvog

January 13, 2009

## EXECUTIVE SUMMARY

This Work Package has created a set of ontologies which can be used for encoding information in triplespace, allowing users who are unaware of each other to communicate using the same computer-interpretable language. The ontologies were created based on the message standards for EDIFACT subsets to provide the breadth of coverage which those messages were designed to handle.

This deliverable explains the process of evaluating and refining the ontologies after their initial creation.

## DOCUMENT INFORMATION

<b>IST Project Number</b>	FP6 – 027324	<b>Acronym</b>	TripCom
<b>Full Title</b>	Triple Space Communication		
<b>Project URL</b>	<a href="http://www.tripcom.org/">http://www.tripcom.org/</a>		
<b>Document URL</b>			
<b>EU Project Officer</b>	Werner Janusch		

<b>Deliverable</b>	<b>Number</b>	7.4	<b>Title</b>	Evaluation and Refinement of Ontologies
<b>Work Package</b>	<b>Number</b>	7	<b>Title</b>	Ontological Infrastructure for Business Processes and Data








<b>Date of Delivery</b>	<b>Contractual</b>	M30	<b>Actual</b>	31-Dec-08
<b>Status</b>	version 1.1		final <input checked="" type="checkbox"/>	
<b>Nature</b>	prototype <input type="checkbox"/> report <input checked="" type="checkbox"/> dissemination <input type="checkbox"/>			
<b>Dissemination Level</b>	public <input checked="" type="checkbox"/> consortium <input type="checkbox"/>			

<b>Authors (Partner)</b>	douglas foxvog (NUIG)			
<b>Resp. Author</b>	douglas foxvog (NUIG)		<b>E-mail</b>	doug.foxvog@deri.org
	<b>Partner</b>	NUIG	<b>Phone</b>	+353 (91) 495 150

<b>Abstract (for dissemination)</b>	This Work Package has created a set of ontologies based on EDIFACT standards which can be used for encoding information in triplespace allowing users who are unaware of each other to communicate using the same computer interpretable language. This deliverable explains the process of evaluating and refining the ontologies after their initial creation.
<b>Keywords</b>	EDI, EDIFACT, ontology, TripCom, triplespace, ontology evaluation

Version Log			
Issue Date	Rev No.	Author	Change
2008-11-19	1	doug foxvog	TOC defined
2008-12-16	2	doug foxvog	Deliverable completed
2008-12-29	3	doug foxvog	Deliverable corrected incorporating QA comments
2009-1-13	4	doug foxvog	Deliverable rewritten to meet QAC suggested redesign

## PROJECT CONSORTIUM INFORMATION

Acronym	Partner	Contact
Semantic Technology Institute Innsbruck <a href="http://www.sti-innsbruck.at">http://www.sti-innsbruck.at</a>	STI  STI · INNSBRUCK	Prof. Dr. Dieter Fensel Semantic Technology Institute (STI) Innsbruck, Austria E-mail: dieter.fensel@sti-innsbruck.at
National University of Ireland, Galway <a href="http://www.deri.ie">http://www.deri.ie</a>	NUIG  National University of Ireland, Galway Ollscoil na hÉireann, Galway	Dr. Laurentiu Vasiliu Digital Enterprise Research Institute (DERI) Galway, Ireland Email: laurentiu.vasiliu@deri.org
University of Stuttgart <a href="http://www.iaas.uni-stuttgart.de/">http://www.iaas.uni-stuttgart.de/</a>	USTUTT  Universität Stuttgart	Prof.Dr. Frank Leymann Inst. für Architektur von Anwendungssystemen (IAAS) Stuttgart, Germany E-mail: frank.leymann@informatik.uni-stuttgart.de
Vienna University of Technology <a href="http://www.complang.tuwien.ac.at/">http://www.complang.tuwien.ac.at/</a>	TUW  TECHNISCHE UNIVERSITÄT WIEN VIENNA UNIVERSITY OF TECHNOLOGY	Prof.Dr. eva Kühn Institut für Computersprachen Vienna, Austria E-mail: eva@complang.tuwien.ac.at
Free University Berlin <a href="http://www.ag-nbi.de/">http://www.ag-nbi.de/</a>	FUB  Freie Universität Berlin	Prof. Dr.-Ing. Robert Tolksdorf AG Netzbaasierte Informationssysteme Berlin, Germany E-mail : tolk@inf.fu-berlin.de
Ontotext Lab, Sirma Group Corp. <a href="http://www.ontotext.com/">http://www.ontotext.com/</a>	ONTO  Ontotext Knowledge and Language Engineering Lab of Sirma	Atanas Kiryakov, Vassil Momtchev, Ontotext Lab, Sirma Group Corp. Sofia, Bulgaria E-mail: vassil.momtchev@ontotext.com
Profium OY <a href="http://www.profium.com/">http://www.profium.com/</a>	Profium  profium	Dr. Janne Saarela Profium OY Espoo, Finland E-mail: janne.saarela@profium.com
CEFRIEL SCRL. <a href="http://www.cefriel.it/">http://www.cefriel.it/</a>	CEFRIEL  CEFRIEL FORGING INNOVATION KNOWLEDGE	Davide Cerri CEFRIEL SCRL. Milano, Italy E-mail: cerri@cefriel.it
Telefonica I+D <a href="http://www.tid.es/">http://www.tid.es/</a>	TID  Telefonica TELEFÓNICA INVESTIGACIÓN Y DESARROLLO	Noelia Pérez Crespo Telefonica I+D Madrid, España E-mail: npc@tid.es

---

## TABLE OF CONTENTS

1	INTRODUCTION	2
2	EDIFACT SUB-STANDARDS	3
3	ONTOLOGIZATION	4
3.1	Need for an ontologization standard . . . . .	4
3.2	Details of the standard . . . . .	4
3.3	Standard usage . . . . .	7
4	ONTOLOGY MERGING	8
5	ONTOLOGY EVALUATION AND REFINEMENT	9
5.1	Ontology Evaluation . . . . .	9
5.1.1	Ontology Evaluation Methodologies . . . . .	9
5.1.2	Evaluation Phase . . . . .	10
5.1.3	Aspects to be Evaluated . . . . .	10
5.1.4	Selected Evaluation Methodology . . . . .	10
5.1.5	Evaluation Results . . . . .	11
5.2	Refinement . . . . .	11
5.2.1	Refinement Methodology . . . . .	12
5.2.2	Refinement Results . . . . .	12
6	CONCLUSIONS	15

## LIST OF ABBREVIATIONS

<b>CEFACT</b>	Centre for the Facilitation of the Administration, Commerce and Transport
<b>EDI</b>	Electronic Data Interchange
<b>EDIFACT</b>	Electronic Data Interchange for Administration, Commerce and Transport
<b>EDIFICAS</b>	EDI in the Finance, Information, Cost, Accounting/Auditing and Social areas
<b>EDIFICE</b>	EDI Forum for companies with Interest in Computing and Electronics
<b>EMEDI</b>	European Medical Electronic Data Interchange
<b>ETIS</b>	European Telecommunications Information Service
<b>MIG</b>	Message Implementation Guideline
<b>UN/EDIFACT</b>	United Nations / Electronic Data Interchange for Administration, Commerce and Transport
<b>WP</b>	Work Package
<b>XML</b>	eXtensible Markup Language

# 1 INTRODUCTION

The aim of the Ontological Infrastructure work package (WP 7) has been to offer a means for a semantically rich definition of business processes for the express purpose of overcoming heterogeneity problems. The concrete goal of Work Package 7 has been to create formal ontologies to encode a significant portion of the current EDIFACT Electronic Data Interchange (EDI) standard in a form that enables the storage of business messages in triplespace as a basis for business-to-business process integration. The purpose of creating these ontologies is to provide a rich enough vocabulary so that potential users of Triple Space which have been previously unaware of each other will be able to communicate with each other without having to negotiate messaging methods and vocabularies.

A standard definition of an *ontology* is a shared, formal conceptualization of a domain[8]. For the purposes of this work package, formal conceptualization means an encoding in a logical computer-processable form and ontologies are restricted to formal definitions of terminology. A *knowledge base* stores information about a particular situation, through encoding that information using terminology defined in one or more ontologies.

Work Package 7 developed ontologies to represent the meanings implicit in EDIFACT messages in Year 2, as described in Deliverable D 7.2[4]. These ontologies represented the range of meanings for the various components of EDIFACT messages, specifically the message types, data segments within those messages, loops of such data segments, data elements within the data segments, and the various codes which were provided as possible fillers for the data elements. This work was grounded on an external ontology (OpenCyc<sup>1</sup>) that already covered half the meanings represented by EDIFACT terms in order to standardize the effort of multiple ontologists and to reduce the work load.

Each set of codes was essentially a separate domain to encode and could be represented by a separate ontology. These ontologies were inter-related, both by defining more general concepts which generalize concepts from different code sets and by determining mappings between concepts in related code sets. Mappings between terms in the developed ontologies and those developed for the two use cases as well as further mappings with the external ontology were constructed in M25-30 as described in Deliverable D7.3[2].

In conjunction with and following the inter-ontology mapping task, a general analysis and clean up of the developed ontologies was performed. The ontologization philosophy was more standardized (Chapter 3), defining relations instead of classes of objects defined by their relationship with something else (e.g., child), defining statuses instead of classes of objects defined by their status (e.g., broken), and ensuring that intrinsic classes of objects were defined by classes, instead of by individuals.

This document briefly discusses the EDIFACT sub-standards we used (Chapter 2), reviews the ontologization process (Chapter 3), discusses the merging (Chapter 4), evaluation and refinement of the created ontologies (Chapter 5). Chapter 6 summarizes the results presented in the previous chapters.

---

<sup>1</sup><http://opencyc.org>

## 2 EDIFACT SUB-STANDARDS

As described in D 7.1 [3], EDIFACT standardizes messages for a wide variety of industries and business purposes. The message definitions thus have to be very broadly defined, allowing for variations used by different industries. Individual industries have defined subsets of these standards (*sub-standards*) called Message Implementation Guidelines (MIGs), selecting only message types used in that industry and limiting the types and options in the messages they select to use. The majority of the messages types are not applicable to general business purposes and have no bearing on the TripCom use cases.

For these reasons and because the full EDIFACT standard with all its variations is far too large to ontologize within the limited number of staff months assigned to the work package, a few of the industry sub-standards which are most closely related to topics of the use cases were selected for developing the ontologies.

The industry standards (and their fields) chosen were

- EANCOM[12] (marketing),
- EDIFICAS [5] (accounting),
- EDIFICE [7] (electronics industry),
- EMEDI [10] (medical), and
- ETIS [6] (telecommunications).

Except for EMEDI (which defined medical messages that were unused in any other subset) each of these industry bodies published their EDIFACT sub-standards online, which allowed access directly to the subset. The EMEDI messages were taken directly from the general EDIFACT standard published by the UN body, CEFACT, since they were medical messages unused by any other industry.

A total of 69 message types were present in at least one of these standards and therefore selected for encoding, 15 of them being present in multiple standards. These messages used a total of 91 data segment types (in the selected subsets), 116 composite data element types, and 325 types of data elements which did not refer to the messaging system itself.

### 3 ONTOLOGIZATION

The main purpose of ontologizing EDIFACT terms is to enable the type of information normally encoded in EDI messages to be expressed in triplespace using the WSML terms created in this work package. A secondary purpose was to establish a mapping between the EDIFACT terms and those created in the ontology. In order to do this, WP 7 needed to map *relations* and *types* to those terms, not just *individuals*.

A *relation* (or “predicate”) is defined by a term that interrelates two or more things. A *type* (or “class” or “concept”) is defined by a term that can have one or more instances of that type. An *individual* (or sometimes “instance”) is defined by a term that is not a relation and that can not have instances of its own; it is normally defined as being an instance of a class.

#### 3.1 Need for an ontologization standard

For creating the ontologies, a methodology was defined in order to standardize the ontologies created by different individuals, to allow ontologies of similar areas to more easily be merged, to make concept overlaps more apparent, and to make the resultant ontologies easier to understand and use by prospective users.

#### 3.2 Details of the standard

To reduce computational complexity, we avoid defining classes as instances of other classes. We also avoided adding rules (or “axioms”) to semantically further define the terms both because the axioms were not provided in the EDIFACT descriptions and because of the extra effort it would take in an already time-constrained task.

In order to preserve the functionality of the EDIFACT messages, every statement implicit in such messages would have to be able to be expressed using the created ontologies. This is why every relationship expressed by a term in the EDIFACT source material required a relation to be defined in the set of ontologies.

It was sometimes appropriate to reify individuals and map EDIFACT codes to them. For example, codes for currencies, countries, organizations, specific laws and regulations, and units of measures were all mapped to individuals.

But in most cases, even though an actual EDIFACT message refers to individuals, the EDIFACT codes are used in the message to indicate either what type of individual it is, or to specify what relationship holds between that individual and something else.

Most of the work creating ontologies was based on defining terms that represented the meanings of codes provided for data elements. The ontologist would first determine a generalization for the set of terms to represent the range of thing which the data element might represent. Generally this was a class (type) of thing and the individual terms were either subclasses or instances of that type. Sometimes the generalization was a relationship, with the individual terms representing more specialized relationships (e.g., the relationship was `identificationString` and the individual values were relationships such as `socialSecurityNumber`, `serialNumber`, and `accountNumber`).

When the generalization was determined, the ontologist was to search for a term with the same meaning both in the external ontology and in the set of ontologies already created. If the term were not found it would be created, and its nearest existing

generalization would be searched for in the available ontologies. The generalized term would be related to the found term. If there may be multiple direct generalizations, each would be related to the term. If the coded terms fell into natural specializations of the general term, those specializations would be ontologized as well. This applies to both relations and classes. After the definition of the generalized term(s) each of the codes provided for that data element would have its meaning ontologized as a subclass, instance, or subrelation of the generalized term(s) and any other appropriate term from the ontology set.

On occasion, some of the codes would represent classes, while others would represent relations. Other times, there would be two or more natural clusters of codes with separate generalizations which seem unrelated. These events occur especially when one sub-standard adds a set of codes to those provided by the EDIFACT standard or when two different sub-standards specify their own codes. In such cases, several generalized terms would be determined for the data element, and the specializations of each would be treated as normal.

For most *relationships* that need to be expressed, there is a specific class of thing to which that relation applies (indicated in WSMML by the keyword “**ofType**”); but in some cases, the existence of the relationship defines the class as well. For example, if a **hasParent** relation holds between two people, one can conclude that the second person is a member of the class **Parent** (if such a class might be useful in the ontology). Such an implication can be indicated in WSMML by using the keyword “**impliesType**” instead of “**ofType**” in the definition of the relation. Note that there is no utility in having a class **Child** with the meaning of someone who has a parent, since every person has parents (at least historically). A class with a different meaning of that name could be useful, however, meaning those people under a certain age – WP 7 uses the class “**HumanChild**” from the external ontology for this meaning.

In an example from the EDIFACT system, the “**Contact Function**” data element is used to identify the type of contact between parties referred to in a message. The value which instantiates this data element could either mean the role which the contact takes in the present situation, a type of organization, or an occupation of a person. A role would be ontologized as a predicate, while an organization type or occupation would be a class (the occupation being treated as the class of all people having such a position). One or another or possibly several of these types of meanings would be useful to have in a business ontology.

The description of a **Contact Function** code taken from an EDIFACT reference might say:

AD = Accounting contact

The contact responsible for accounting matters.

Not only does the relationship need to be defined for encoding a message which uses this code, but several classes of entity are suggested for inclusion in the ontology. WP 7 ended up ontologizing this as follows, using some related concepts which already exist in the external ontology we use; first defining the classes:

```
concept Accountant_Generic
  subConceptOf cyc#IntelligentAgent
  nonFunctionalProperties
    ‘‘A person or organization that performs accountancy’’
  endNonFunctionalProperties
```

```

concept cyc#Accountant
  subConceptOf {Accountant_Generic, cyc#Person }
  nonFunctionalProperties
    ‘‘A person who performs accountancy’’
  endNonFunctionalProperties
concept cyc#AccountingFirm
  subConceptOf {Accountant_Generic, cyc#ServiceOrganization}
  nonFunctionalProperties
    ‘‘A firm that performs accountancy’’
  endNonFunctionalProperties
concept cyc#AccountingDepartment
  subConceptOf {Accountant_Generic cyc#Department}
  nonFunctionalProperties
    ‘‘A department that performs accountancy for a larger organization.’’
  endNonFunctionalProperties

```

and then a relation:

```

relation accountingContact
  ( ofType edi#BusinessMessage,
    impliesType Accountant_Generic)
  nonFunctionalProperties
  dc#description hasValue
    ‘‘accountingContact (MESS, ACCT) means that the accounting
    contact for dealing with the specified message is the
    accountant or accounting organization, ACCT.’’
  endNonFunctionalProperties

```

Note that the meaning of the EDIFACT code would **not** be an *instance* of `cyc#Accountant`, `cyc#AccountingFirm`, or `cyc#AccountingDepartment`. It would be used, instead, to state that the referent in a given EDIFACT message is such an instance. A statement such as:

```
instance AccountingContact memberOf Accountant_Generic
```

would mean that there is an individual accountant referenced by the WSML term, “AccountingContact”. If such an individual were specified as the meaning of an EDIFACT code, that would mean that every EDIFACT message which used this code referred to the exact same accountant.

The standard WSML naming conventions were used: names of relations start with a lower-case letter while names of individuals and concepts start with an upper-case letter.

The concepts, relations, attributes, and individuals that WP 7 created were designed, not only to be usable by the Use Case work packages for encoding their ontologies, but to provide ontologies to ease businesses into using triplespace for business communication.

A textual description of the meaning of each created term, derived from the description of the code provided by the data source, was included in the ontology to explain the meanings and help others to be able to use the ontology with the intended meanings.

A knowledge base was made for each coded data element mapping each provided code with the term created for that code. These were not part of the general ontologies, but served to allow future tools to be created to establish mappings to and from EDIFACT messages, and provided an evaluation point to ensure that an item was in the ontology to represent each EDIFACT code.

### 3.3 Standard usage

This method was used for ontology standardization in the ontology creation, merging (see Chapter 4), evaluation, and refinement phases (see Chapter 5).

## 4 ONTOLOGY MERGING

As the ontologies were being created for the terms in the code sets for different ED-IFACT data elements, it was often noted that there was significant overlap between two different code sets or that the topic of the code set being worked on was closely related to that of one which had already been ontologized. In such cases, the subsequent code set(s) were ontologized into the same ontology as the previous one(s) on the same topic. However, in other cases, for example when different people worked on related data elements, the similarity of two different data elements was not initially noted and separate ontologies were created for overlapping sets of meaning.

The ontology merging phase was discussed in Deliverable 7.3 [2], so will not be discussed here. However, it necessitated the simultaneous start of the ontology refinement phase (see Chapter 5). In cases in which different ontology modules used different coding techniques, they had to be standardized in order to perform the merge.

## 5 ONTOLOGY EVALUATION AND REFINEMENT

As the ontologies were being created for the terms in the code sets for different ED-IFACT data elements, it was often noted that there was significant overlap between two different code sets or that the topic of the code set being worked on was closely related to that of one which had already been ontologized. In such cases, the subsequent code set(s) were ontologized into the same ontology as the previous one(s) on the same topic. However, in other cases, for example when different people worked on related data elements, the similarity of two different data elements was not initially noted and separate ontologies were created for overlapping sets of meaning.

### 5.1 Ontology Evaluation

After the initial set of ontologies were created, the evaluation phase started, to ensure that the created ontologies followed the ontologization methodology described in Chapter 3 and that ontologies for overlapping or very similar topics were merged.

The evaluation was not handled as a separate stage, but was integrated into the refinement and merging (see Chapter 4) stages. The standardization of the technique for encoding the ontologies was necessary to enable different ontology modules created for similar subjects to be integrated and merged.

The purposes of the evaluation was to enable the standardization of knowledge encoding, the correction of the precision of term meanings, and the detection of missing information about individual terms (e.g., a second parent class) or any errors.

#### 5.1.1 Ontology Evaluation Methodologies

According to [1], ontology evaluation approaches normally fall into one of several categories:

- ones based on comparison with a “golden standard”,
- ones based on evaluating the results of using it in an application,
- ones based on comparison to a source of data, and
- manual evaluations assessing how well the ontology meets a set of predefined criteria.

and the ontologies are analyzed at different “levels”.

- Vocabulary
- Hierarchical
- Relations (non-hierarchical)
- Syntactic
- Structural

Ontology evaluation normally occurs during the design phase (by designers), the implementation phase (by implementers), the phase of integrating multiple ontologies (by the integrator), the final pre-release phase after the implementation is basically complete (by or for those responsible for the completed ontology), and after its release (by users). The KnowledgeWeb project classified design phase ontology evaluation as the “*pre-modelling stage (type 1)*”, implementation phase ontology evaluation as the “*modelling stage (type 2)*”, and the post-release ontology evaluation as “*type 3*” [9]. They do not distinguish an integration or final testing phase from the implementation phase.

### 5.1.2 Evaluation Phase

Work Package 7 uses the EDIFACT design, so we rely on on the massive effort of the designers of EDIFACT (and its subsets) for the design evaluation. We started the implementation evaluation in parallel with the implementation work itself. The integration phase started during the implementation phase and continued after the initial ontologies were completed. The final pre-release phase started after ontologies were basically stabilized. We are not yet to the phase of post-release evaluation.

### 5.1.3 Aspects to be Evaluated

The evaluation procedure analyzed the created ontologies with respect to preserving the functionality of the EDIFACT messages, the adherence of the ontologies to the ontologization standard, and its self-consistency. It did not evaluate the conceptual model of the ontologies, as the conceptual model laid out by the designers of the EDIFACT subsets was being used.

- The Vocabulary was evaluated to ensure that concepts, relations, and instances were not duplicated and that the names followed the WSML standards.
- The Hierarchy was evaluated to ensure that every new class was defined as a subclass of an appropriate class, every individual was defined as an instance of at least one appropriate class, and every relation had been examined to determine if it was a subrelation of an already defined relation.
- The Relations were evaluated to ensure that appropriate range and domain classes were defined.
- The Syntax was evaluated for validity by the WSMT tool.
- Structural evaluation was performed manually as described in the next subsection.

### 5.1.4 Selected Evaluation Methodology

Although a “golden standard” approach was not used, the work package decided early to base its ontology construction on ontology reuse of a large general purpose ontology – both to ensure that different ontologists would link similar concepts to the same area in existing hierarchies and to obviate spending effort re-creating work that had already been done and released to the public.

Since the set of ontologies developed in this work package was derived from a set of data sources, a basic approach was a manual evaluation based on the source of data – the documentation describing the message components of the selected EDIFACT subsets – and the criteria specified for developing the ontologies. Software tools were used for verifying the syntactic level and analyzing the hierarchical structures.

Because a knowledge base was created mapping each data element code to a term in the ontology used for that data element, an automatic comparison of that knowledge base with the source files was able to establish that every data element code was accounted for in the set of ontologies. The inclusion of the text string from the data source (or a variant thereof) associated with the ontology term, served to allow for a rapid verification that the code-term mapping seemed reasonable.

During ontology construction, the produced ontologies were loaded into the WSMT tool [11] to ensure syntactic validity. This process also flagged certain semantic faults such as loops in hierarchical definitions. As multiple ontologies were loaded into the tool, graphical displays could be used to examine subclass hierarchies making it easy to identify if two ontologies created similar subclasses of an existing class – either from the external ontology or from the developing set of ontologies. The positioning of new classes in a growing directed acyclic graph of existing classes could be examined for evaluation whether the new class was correctly positioned or indeed equivalent to an existing class.

Relations defined in the ontologies appeared in the WSMT graphical tool attached to the appropriate node in the subclass hierarchy. This allowed the viewer to easily consider whether the argument constraint should be broadened (or possibly narrowed). When the existence of a relation between two individuals necessarily restricted an argument to a narrower class (e.g. the relation `hasParent` implies that its second argument is an instance of the class `Parent`, a WSML `impliesType` declaration can be added to the predicate at this time.

Manual verification ensured that the generated ontologies followed the ontologization standard of Chapter 3.

For the purposes of dissemination, the ontologies were converted into OpenCyc's language, CycL, to enable the sharing of the new concepts created with the external ontology. The process was handled step by step through editor scripts, with semantic validity of the results being verified by the OpenCyc API. Typographical and conceptual errors were brought to light in this process, either through the lack of conversion by the editor script or by error messages from the OpenCyc API.

### 5.1.5 Evaluation Results

The results of ontology evaluation were tightly integrated with ontology refinement. Lists were not maintained of problems detected; corrections and refinements were made upon detection, allowing the evaluation process to proceed linearly.

## 5.2 Refinement

Standardization of the ontologies was necessary since, in some instances, the method described in Chapter 3 was not fully followed and some sets of classes had been defined as instances of metaclasses or some relationships had been encoded as classes.

### 5.2.1 Refinement Methodology

When violations of the ontologization standard were detected during evaluation, the faulty ontology was modified to follow those standards.

In cases in which classes were originally encoded as instances of metaclasses, a re-encoding was done. The (meta) classes of which such terms were defined as instances were removed from the ontology; instead a super-class was defined (or used if already extant) for each such group of classes, with the types defined as subclasses. Information about the existence of the metaclass was maintained as it would be encoded in the distribution to OpenCyc.

Sets of relations were examined to determine if a type should be concluded for the filler of either of the arguments. If such an implied type did not already exist in the ontology, it was added.

Similarly, sets of defined classes were examined to determine if they were properly modeled as classes, or whether relations were more appropriate. For example, the **Relationship Description** data element has a number of codes which can be interpreted as classes as well as relationships between people. This data element is used for “parent”, “child”, “owner”, “neighbour”, “guardian”, and the like. Although the meaning is clearly to identify one party based on its relationship with another (and thus should be modeled by binary predicates), some of the words also denote types of people – meanings which could be useful in some contexts – so appropriate classes were also included in the ontology.

### 5.2.2 Refinement Results

Following the ontology merging, evaluation, and refinement steps, around 80 topical ontologies remained. These have been published on the TripCom website at <http://www.tripcom.org/ontologies/EDI/generic.php>. The topics of these ontologies are:

Ontology	Topic
Acknowledgment	Acknowledgment and obligation type; document status
Address	Predicates relating to (parts of) an address
Allowance Charge	Charges, allowances, payments
Analog Binary Format	Format of data/transmission, or storage: binary/analog
Back Ordering	Conditions for back-ordering
Cargo	Type of goods: documents, hazardous, frozen, ...
Characteristic Type	Basic type of something that parts of message refer to
Code List Responsible Agency	Agency which defines code lists
Commitment	Variety of commitment
Communication Channel	Address and channel of communication
Computer Environment	Computer hardware, software, and processes
Contact Function	Department or position (in a company) of a contact
Contract Condition	Type of agreement or condition on a contract
Currency	Currency, countries, and the relationships between
Currency Usage	How a currency is treated in a business message
Dangerous Goods Regulation	Set of regulations governing dangerous goods
Date Format	Format used to express a date or time
Date-Time Relations	Relations for dates of events and scheduled events
Delivery Instruction	Type of instructions for delivering goods
Delivery Transport Function	Transport stage where responsibility for goods is switched
Description Format	Basic format in which a description is provided
Despatch Pattern	Frequency/timing of dispatch of (usually) goods
Dimension	Relations for what aspects are dimensions provided for
Duty Regime	Basis on which duties are charged
Equipment Supplier	Party responsible for shipping, sealing shipment, ...
Free Text Function	Why free text provided & whether standardized text
Gender	Gender
Geographic Range	Domestic, European, or int'l and if regulated as such
Government Agency	Government agencies (mostly those dealing with trade)
Handling Instruction	Features regarding or impinging upon handling of goods
Hierarchical Relationship	Relative position of item in a hierarchy
Hierarchy Object Qualifier	Hierarchy defined for product, party, accessory, location
Hospital Admission	Admission type by urgency, electivity, duration
Information Process	Processing type by how structured the data is
Instruction	Message type: request, acknowledgment, instruction, ...
Instruction Type	Generic type of business message
Inventory Balance	Type of inventory keeping
Inventory Movement	Transfers in and out of inventory
Item Characteristic	Aspect for which info is provided (e.g., pattern, colour)
Item Description	Availability, text feature, refundability, etc.

Ontology	Topic
ID Type	ID relations by ID type
Language	Human languages
Location Function	Relations dealing with location
Means of Transport	Vehicle type and other transport
Measured Attribute	Relations for measured amounts
Measurement Purpose	Quality measured or reason for measurement
Measurement Significance	Relations comparing two measurements
Medical	Things dealt with in medical messages & not other messages
Message Function	Function, status, or action with respect to a message
Message Information Type	Generic information type in a message
Monetary Amount for	Relations between an amount of money & something else
Name Format	Format for a person's provided name locating name parts
Occupation	Employment occupations
Package Marking	Types of markings on packages
Packaging	Types of packaging devices, incl. pallets and containers
Packaging Instructions	Instructions for packaging
Packaging Level	Nested packaging description
Party Function	Relations one party plays with respect to something
Payment Guarantee Means	Type of guarantee made for payment
Payment Term and Means	Payment type restricted by some condition
Percentage Basis	Relations applying percentages
Price Qualifier	Relations applying prices and charges to various things
Product Group	Types of predefined product groups
Product Identifier	How one product in a purchase order relates to others
Purpose	Purpose of event, plan, task, ...
Quantity	Relations of some aspect of a transaction to a quantity
Range	Relations specifying a range by type (temperature, ...)
Relative Time	Relations between two dates
Religion	Religions
Service Requirement	Type of loading, shipping, and delivery
Shipping Unit Quantity	Relations referring to count of aspect of shipping
Special Conditions	Types of conditions placed on objects (to be) shipped
Special Service	Special shipping event types, charges, and allowances
Standards	Standards which products need to comply with
Status	Possible statuses
Status Reason	Status related concepts (mostly notices)
Subject Qualifier	ID relations and code sets
Tax	Tax types
Temperature for Process	Relations specifying transport & storage temperatures
Time Period	Durations and calendrical time periods
Unit of Measure	Units of measure

Table 5.1: Ontology Topics

## 6 CONCLUSIONS

The ontologies developed within this work package capture the most common relations, types, and instances referred to in business messages. They have built upon a broad publicly available ontology (OpenCyc) whose terms are defined for the Semantic Web.

The developed ontologies were evaluated with respect to the ontologizing criteria described in Chapter 3. The coverage of the ontologies was left that of the EDIFACT messages from which they were derived; neither this coverage, nor the conceptual model implicit in the messages were evaluated.

The evaluation covered the created terms to ensure that they were defined as classes, relations, or individuals according to the ontologization criteria, that they covered the meanings in the data sources, and that separate terms with the same meanings were not created. Relations were checked to ensure that reasonable types restrictions were defined for their arguments. Classes were checked to ensure that they had reasonable parent classes defined. Individuals were checked to ensure that they were defined as instances of reasonable classes. Created terms were checked against the external ontology to determine whether their meanings were already ontologized in that ontology.

The developed ontologies have been refined and merged as appropriate and are now ready for distribution.

The significant overlap of these ontologies with those independently created for the two use cases demonstrates the usefulness of these ontologies for diverse communications tasks.

---

## REFERENCES

- [1] J. Brank, M. Grobelnik, and D. Mladenic. A Survey of Ontology Evaluation Techniques. In *Conference on DataMining and Data Warehouses (SiKDD 2005)*, 2005.
- [2] Guillermo López Reyes David de Francisco Marcos and doug foxvog. Ontological infrastructure for business processes and data; TripCom D7.3. Deliverable, TripCom, October 2008.
- [3] doug foxvog. Ontological infrastructure for business processes and data. <http://www.tripcom.org/docs/kickoff/WP7.pdf>, April 2006.
- [4] doug foxvog. Ontology of EDIFACT syntax and semantics; TripCom D7.2. Deliverable, NUIG, March 2008.
- [5] Cost Accounting/Auditing EDI in the Finance, Information and Social areas. Guide de l'archivage électronique sécurisé. <http://www.edificas.org>, 2005.
- [6] The Global IT Association for Telecommunications. ETIS-EDIFACT User Guide. <http://www.etis.org>, 2005.
- [7] The European B2B Forum for the Electronics Industry. UN/EDIFACT MIGs. <http://www.edifice.org>, 2005.
- [8] Tom Gruber. A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition*, 5:199–220, 1993.
- [9] Jens Hartmann, Peter Spyns, Alain Giboin, Diana Maynard, Roberta Cuel, Mari Carmen Surez-Figueroa, and York Sure. Methods for ontology evaluation; KnowledgeWeb D1.2.3. Deliverable, KnowledgeWeb, 2005.
- [10] European Medical Electronic Data Interchange. EMEDI overview. <http://www.accart.nom.fr>, 2003.
- [11] Mick Kerrigan. D9. 1v0. 2 web service modeling toolkit (WSMT). Technical report, 2005.
- [12] The Global Language of Business. EANCOM Guideline. <http://www.gs1.org>, 2005.