Modelling functional agreements in EDIFACT environments

C. Huemer[*]*, G. Quirchmayr*[*]*, A M. Tjoa*[**]
**Institute of Applied Computer Science and Information Systems. University of Vienna, Austria, e-mail: {ch,gq}@ifs.univie.ac.at*
***Institute of Software Technology, Vienna University of Technology . Austria, e-mail: tjoa@ifs.tuwien.ac.at*

Abstract
Communication processes in the commercial sectors are more and more using the services offered by the telecommunication industry. This is also due for the electronic data interchange (EDI) of business data which represents one of the most recent teleservices. This trend leads to the standardization of the data exchange, where EDIFACT (Electronic Data Interchange For Administration, Commerce and Transport) seams to be the global standard for the forseeable future. However, the EDIFACT standard approach includes a number of shortcomings that keep many firms and organizations from participating in this area of electronic commerce. One of the worst shortcomings is that in order to be generally valid EDIFACT is too complex in its structure and consequently too hard to read and navigate. The ‘real’ data interchange between two business partners is usually based on a very small subset of the generally valid standard. Therefore, a detailed functional interchange agreement to define the format of the ‘real’ data is necessary before two partners start the interchange for the first time. This agreement is usually done through conventional communication methods, e.g. via telephone or fax, which results in a serious backdraw. Our paper presents a concept for interchanging the functional interchange agreements between the business partners via EDI itself.

Keywords
Electronic Data Interchange (EDI), EDIFACT, communication standards

1 INTRODUCTION

Since communicating with other organizations becomes more and more important, efficient communication processes are one of the most important factors of success for a company. Therefore, many firms already take advantage of the services offered by the telecommunica-

tion industry, like faxes or e-mail. Using faxes and e-mail reduces the transportation time for a business document, but is still slow and expensive, because someone has to interpret and rekey the exchanged information. In addition, rekeying also increases the error rate. A solution to speed, cost and error problems would be the electronic movement of business data between or within firms in a structured, computer processable data format that permits data to be transferred without rekeying from a computer supported business application in one location to a computer supported business application in another location. This movement is defined as Electronic Data Interchange (EDI) in (Hill, 1989). A precondition for EDI is that both business partners agree on the representation of information to be sent from one computer application to the other (Berge, 1989). In addition to the format, what items and how they are individually structured and put together, both partners must also have the same understanding of the semantics of the interchanged data.
In the past decade various different standards have been proposed, but they are either limited to a certain branch of industry (e.g. ODETTE for the automobile industry) or build just a national solution (e.g. ANSI X.12 in the USA). Since 1988 the United Nations (UN) are developing the EDIFACT (electronic data interchange for administration, commerce and transport) standard to meet the requirements of an internationally valid general business standard. This international standard (EDIFACT, 1989) includes the rules on the application level for the structuring of user data and of associated service data in the interchange of messages in an open environment. Beside the syntax the EDIFACT standard also covers the definition of data elements (= the data information as basic component for message types), segments (= a functionally related set of data elements) and message types (structured representation of the full information on an electronic business transaction).

Since new message types are developed and definitions of existing message types are changing in the course of time, EDIFACT can be considered a dynamic standard. The complete documentation of the EDIFACT guidelines of a period of time is included in an UN/EDIFACT directory which comprises the message type directory, the segment type directory, the composite data element type directory, the simple data element type directory and the code list directory. At the moment EDIFACT directories are published by the UN either on paper or in ASCII format. In order to express any business transaction of a real-world situation, the consequence is a rather long and complex structure of the EDIFACT standards. Thus, in order to be generally valid the various directories are quite inflated compared to the elements actually used in a single transaction between two business partners. Therefore, it is hard to read and browse the paper documents which describe the standard directories to implement a business transaction.

Owing to the complexity of the standard it is impossible to implement a standard-to-application converter which can handle all semantics that can be included in any incoming standard message. Thus, two business partners willing to exchange business transactions via EDIFACT must agree on the subset of elements of the standard message which they really want to exchange. Such a functional agreement is usually done through conventional communication methods, e.g. via telephone or fax, which results in a serious drawback. But this is at the moment the only way to specify an interchange format including limited as well as sufficient semantics to implement a standard-to-application converter. Our paper presents a concept for interchanging the functional interchange agreements between the business partners via EDI itself. The proposed method is based on the EDIFACT Directory Definition Message (DIRDEF) which will allow the transmission of an EDIFACT Directory set or parts thereof in

EDIFACT syntax. The DIRDEF message only has draft status, but already includes the semantics needed for our project.

The first main component of our method comprises a tool for building subsets of existing messages and for creating wild subset which manipulate the messages in a non standard conform way. This enables the adoption of a message design to the real business needs of the user's company. The self-created revisions can be translated into the DIRDEF format which allows electronic transmission to the partner company. Furthermore, the message definition in DIRDEF format is a valid input format to the second main component - the EDI translator, which is used to map an instantiated EDI message carrying business data to the input format of the business application. If the business partner also uses an EDI translator that is able to accept DIRDEF as input format, he will be able to handle messages in the format created by the initiating company. This means a consequent extension of the basic idea of EDI, because the agreement between two companies on the interchange structure will be based on EDIFACT.

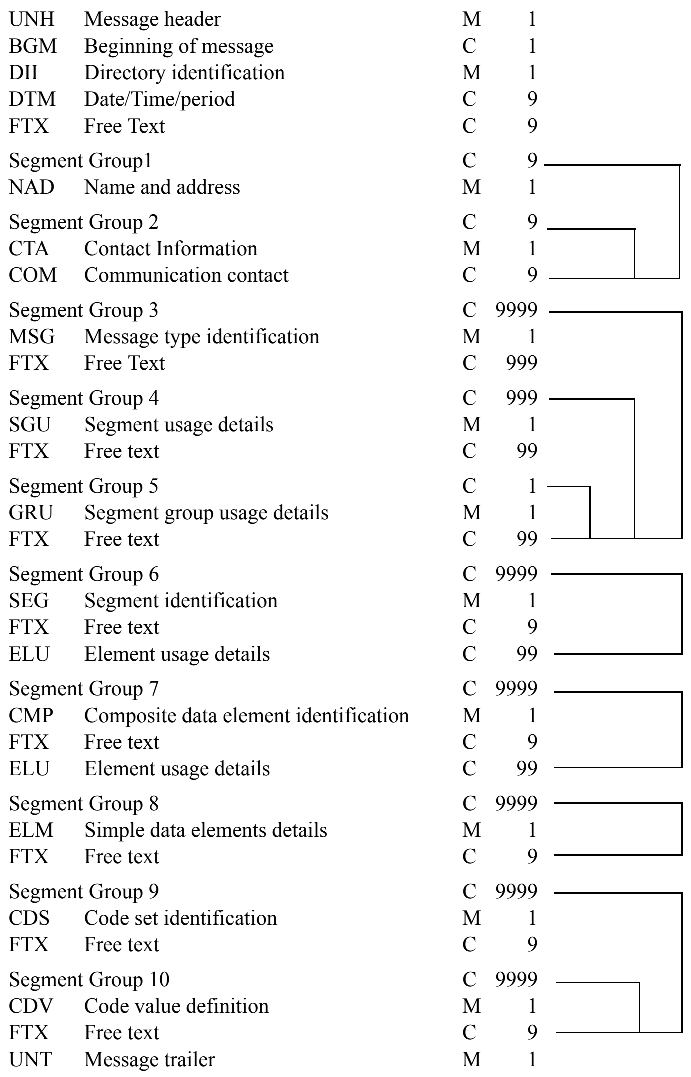
The subsequent sections of the paper are organized as follows. Section 2 describes the DIRDEF message which is needed for a complete understanding of our method. In section 3 we present our standard browsing tool which enables the establishment of subsets or wild subsets. The concept of exchanging functional agreements is shown in Section 4. In Section 5 we describe our approach by means of an order message as practical example. We conclude with a short summary and future perspectives.

2 THE DIRECTORY DEFINITION MESSAGE (DIRDEF)

The specification depicted in Figure 1 provides the definition of a UN/EDIFACT Directory Definition Message (DIRDEF) to be used in EDI between partners involved in administration, commerce and transport. The message allows the transmission of a UN/EDIFACT Directory. One occurrence of the message can contain only one version of a UN/EDIFACT Directory set or parts thereof (DIRDEF, 1993). A UN/EDIFACT Directory set comprises:

- Message type directory
- Segment type directory
- Composite data element type directory
- Simple data element type directory
- Code list directory.

The DIRDEF message is structured as in Figure 1. It starts with the header segment and the segment signifying the beginning of the message. The DII (directory identification) segment contains the information about the standard directory to be described within the message - such as the version, release, status and the controlling agency. The following segments up to Segment Group 2 are used to transmit more general information about the DIRDEF message. The definition of the message type directory is given in segment groups 3 to 5. For each included message a instantiation of a group 3 (which includes multiple groups 4 and 5) is necessary. The MSG (message type identification) segment identifies the message type to be specified, providing its identifier, name, version number etc. The following FTX (free text) segments are used to give further information on the message itself as unstructured text. The segment groups 4 and 5 are used to specify the structure of the message and its related functional definition. This means that for each segment used within the message a SGU (segment



UNH	Message header	M	1
BGM	Beginning of message	C	1
DII	Directory identification	M	1
DTM	Date/Time/period	C	9
FTX	Free Text	C	9
Segment Group1		C	9
NAD	Name and address	M	1
Segment Group 2		C	9
CTA	Contact Information	M	1
COM	Communication contact	C	9
Segment Group 3		C	9999
MSG	Message type identification	M	1
FTX	Free Text	C	999
Segment Group 4		C	999
SGU	Segment usage details	M	1
FTX	Free text	C	99
Segment Group 5		C	1
GRU	Segment group usage details	M	1
FTX	Free text	C	99
Segment Group 6		C	9999
SEG	Segment identification	M	1
FTX	Free text	C	9
ELU	Element usage details	C	99
Segment Group 7		C	9999
CMP	Composite data element identification	M	1
FTX	Free text	C	9
ELU	Element usage details	C	99
Segment Group 8		C	9999
ELM	Simple data elements details	M	1
FTX	Free text	C	9
Segment Group 9		C	9999
CDS	Code set identification	M	1
FTX	Free text	C	9
Segment Group 10		C	9999
CDV	Code value definition	M	1
FTX	Free text	C	9
UNT	Message trailer	M	1

Figure 1 Message structure of a DIRDEF message

usage details) determines the segment tag, the requirements designator, the number of occurrences, the level number and the sequence number (position in the message). Similarly, a GRU (segment group usage details) specifies the group identification, the requirement designator, the number of occurrences and the sequence number for each segment group used within the message. FTXs are included to provide further textual information on the segment or group usage. Segment group 6 defines the segment type directory which must be instantiated for each included segment. SEG (segment identification) identifies, among others, the segment tag and name, whereas the FTX provides further textual information on the segments of the directory. The following ELU (element usage details) segments specify the contents of data elements of the segment type by stating the data element tag, the requirement designator and the sequence number (position in the segment) of the included composite and simple data elements. The composite data element type directory is expressed in segment group 7, where each defined composite data element is one instance of group 7. CMP (composite data element identification) covers the composite data element tag and name; additional unstructured information is again stated in the following FTXs. The ELUs list the component data elements in the composite data element by indicating the simple data element tag, requirement designator and sequence number. Segment group 8 represents the simple data element type directory. ELM (simple data elements details) which specifies the data element tag, the character representation, the possible length, a flag for coded elements must be stated for each simple data element of the directory. The code list directory is provided in segment groups 9 and 10. Since for coded simple data element more than one different code set may exist, CDS (code set identification) specifies a code set by listing the code set identification, the corresponding simple data element tag. The used codes within the codes are identified in the loop of group 10, where each entry of CDV (code value definition) describes a code by value and name. The FTXs are used to provide further textual information on the code sets or the codes, respectively.

3 SPECIFICATION OF THE BROWSER AND EDITOR TOOL

The EDIFACT browser tool is designed to navigate through the EDIFACT standard directories and self-created subsets or versions in a very flexible and clearly arranged manner. It allows access at each level of the EDIFACT type directories (messages, segments, composite data elements and single data elements) and offers links between these levels. This means that the user need not read the whole paper-written standard to get an answer to a very specific question. The functional specification of the browser is depicted in Figure 2. The browser component for its own is a tool for the newcomer to EDIFACT who wants to get an overview of the possibilities of electronic data interchange in the commercial sector.

The browser tool provides the following information

on a **standard directory**:

- Included messages,
  segments,
  composite data elements and
  simple data elements
- Textual information on the directory

on a **message**:

- Functional description of the message
- Textual information on the message
- Composition of the message (Segment structure)
- Textual information on the usage of a segment in a specific position within the message

on a **segment**:

- Functional description of the segment
- Textual information on the segment
- Composition of the segment (Data element structure)
- List of messages which include the segment

on a **composite data element**:

- Functional description of the composite data element
- Textual information on the composite data element
- Composition of the composite data element (simple data element structure)
- List of segments which include the composite data element

on a **simple data element**:

- Functional description of the simple data element
- Textual information on the simple data element
- Code lists assigned to a coded simple data element
- List of segments and composite data elements which include the simple data element

on a **code list** and **codes**:

- Functional description of the code list
- Textual information on the code list
- Codes assigned to a code list
- Functional description of the codes
- Textual information on the codes

Figure 2 Functionality of the browser tool

In order to browse through an EDIFACT standard directory it first has to be included into the database of the browser tool. According to the basic idea of EDI it is desirable to receive the EDIFACT standard directory description on-line from the various EDIFACT reference databases (e.g. UN/EDIFACT reference database in Geneva, reference database of DIN in Germany) via a DIRDEF message. Then the browser tool is able to convert DIRDEF into an adequate format for database import. Since until now one cannot yet receive DIRDEF messages on-line from the reference databases, one has to work with discs. Furthermore, the organizations are not willing to provide all standard directories in DIRDEF format even on discs. Therefore the browser tool is also equipped with a converter, which is able to translate ASCII transcripts of the standard directory into the database input format.

Having performed the input procedure, the standard directory is not only ready to be browsed, but also serves as a starting point for creating subsets and wild subsets. The preferred way of building subsets is a top-down method. First the user selects those messages of the standard which should be included in the subset or in other words those he wants to interchange with the business partner. By activating a synchronisation method the subset is also reduced to the segments, composite and single data elements which are used in these messages. The segments and data elements not used in the messages are automatically deleted from the standard. The next step is to manipulate the message type directory by selecting only those segments of the segment structure in a message, which might actually be used in a data interchange and deleting the rest from the segment structure. The synchronisation method again deletes the unused data elements which have been deleted from all messages in this step. The two following steps of manipulating the segment, composite and simple data element type directory are also similar. Only those components (data elements of segments, simple data elements of composite ones) which might be candidates for the exchange of messages are selected and the rest is removed. An invocation of the synchronisation method deletes the unused components down the EDIFACT hierarchy. Since simple data elements have no components, the simple data element directory is just adapted automatically by the synchronisation method. But the code list and their codes for a coded simple data elements can be reduced to the one codelist and to its codes which are used in the business transaction, reflecting on the code list directory. Please note, that it is sufficient to apply the synchronisation method at the end of the whole procedure, but its invocation after each step increases the transparence and clearness of the process, because the user can concentrate on the actually remaining components.

The described procedure is the preferred way of adapting a standard directory to the business needs of a company leading to an EDIFACT conform subset. But usually the company wants to report on the implemented changes, so that the partner company also knows why and where the changes have been made. This is usually done in the textual information referring to a message, segment, segment usage within a segment, and so on. Therefore, the editing tool allows changes and add-ons to the unstructured text information which are provided for each component. Since the text adoptions do not reflect on the defined functionality of the interchange format, this is an allowed operation also for pure subsets.

More complex is the establishment of wild subsets, which change (the order) or add new functionality to the existing standard definition of EDIFACT. Furthermore, wild subsets are only useful, if both partners practice the whole process of modelling functional agreements or have at least a converter which accepts DIRDEF messages as input format for modelling the wild subset-to-application translation. To create wild subsets we propose to first apply the top-down method to define an EDIFACT-conform subset and then to implement the ‘wild’ manipulations bottom-up from simple data elements. We do not start from the real ‘bottom’ of codes, because codes must be assigned to a specific coded simple data element. Therefore, a coded simple data element must be established before its codes can be manipulated. This means that first a completely new coded simple data element has to be created or a former uncoded one has to be defined as coded before codes are added to it. In the case of manipulating the codes of a simple data element already defined as coded , the process starts at the code list type directory level. There, new codes might be added or the semantics of a code might be changed. Of course, it is also possible to define new uncoded simple data elements, which might - as the coded ones - be completely new or versions of an existing one with a different identifier. Up from the composite data element type directory, three kinds of manipulations are possible. First, the order of the component structure might be changed. Second, in the former step self-created components can be added to the component structure. Last, new types or versions of existing types can be created covering EDIFACT-conform as well as self-created components. These changes might be applied to all composite data elements, segments, and messages. Equivalent to pure subsets, the documentation of all the changes will all be made in corresponding text elements.

Another part of the browser and editor tool is a separate documentation tool. The documentation tool is not necessary for the modelling of functional agreements, but it permits the creation of paper documents of included pure and wild subsets. It produces a documentation of the created subsets in a similar way to the documentation of the EDIFACT standard made by the UN, including the functional structure such as segment tables and diagrams, segment composition and so on, as well as an informal description according to the specifications made in the text elements of the editing tool. If the partner company also uses our tool, the documentation tool is of special interest, because that way the business partner will also be aware of a paper documentation of the concrete exchange format defined in the subsets.

4 EXCHANGING FUNCTIONAL AGREEMENTS

In this section we describe our proposed scenario for exchanging functional agreements which are based on the business needs of the business partners via EDI and the subsequent exchange of messages based on these agreements. This section should be read in conjunction with figure 4 which depicts the overall process. The first step for the initiating business partner is to include the EDIFACT standard directories into the browser & editing tool. Hopefully, in the near future this can be done by receiving on-line a DIRDEF message from an EDIFACT reference database (1a). At the moment it is only possible to import them from the directory descriptions provided in ASCII formats on discs (1b). Now the initiating business partner develops an EDIFACT-conform subset (2a) or a wild subset (2b) of a standard version which will include only those formats of data which will actually be exchanged in the business transaction. Via the documentation tool the initiating business partner can print a documentation of the self-created standards, which can be used as implementation guideline (3a, 3b). The subset definition for the actual data exchange can be exported as DIRDEF message from the browser and editing tool. In case the converter accepts DIRDEF as input format the subset specification can be provided to the converter software (5). This means that the converter is aware of the subset definition and that mapping tables can be defined (6), which specify how data included in an EDIFACT message of the subset format is mapped to the application database input format or how data from the application database output format is mapped to an EDIFACT

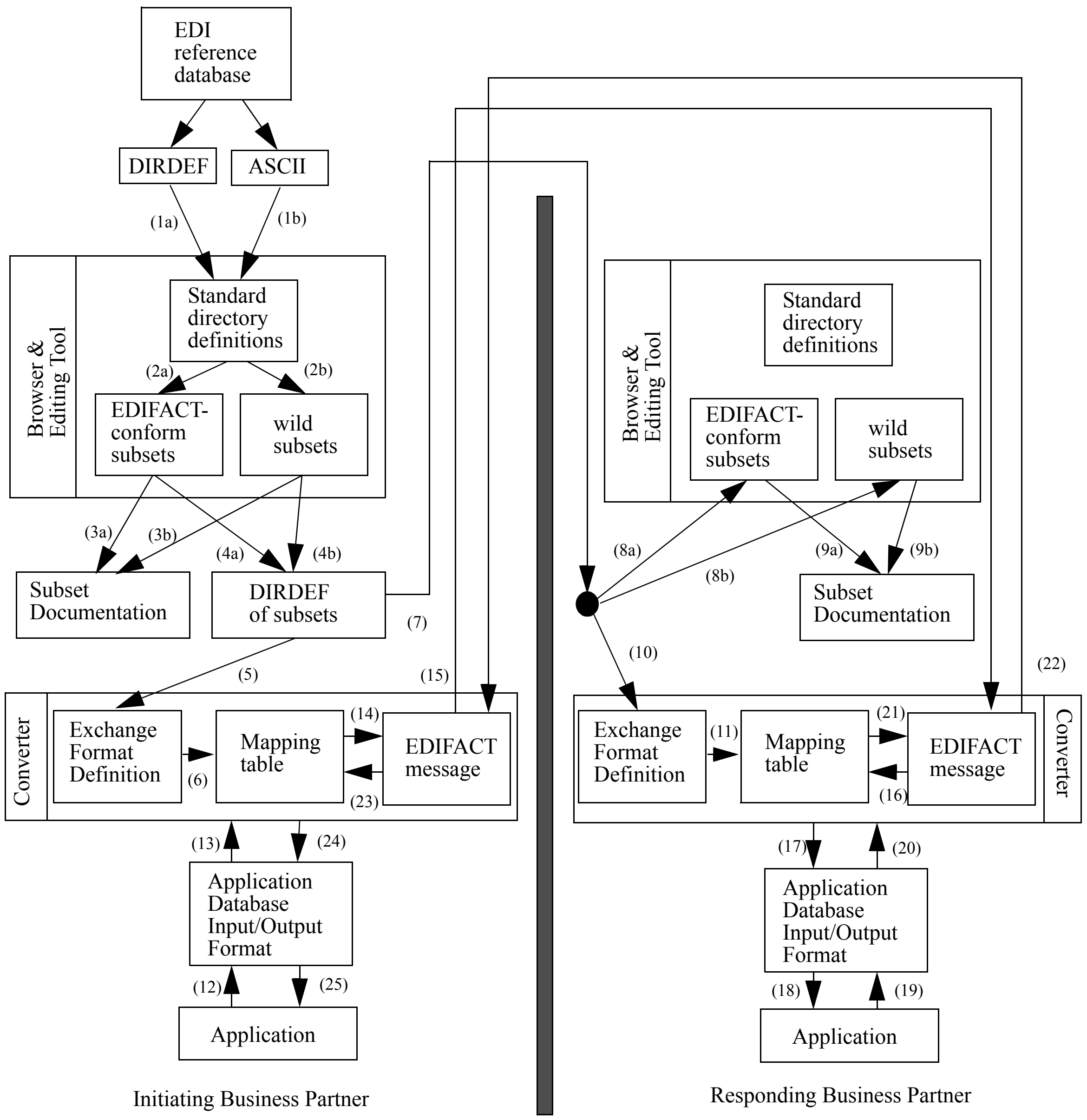


Figure 3 Scenario for exchanging functional agreements

message of the subset format, respectively. In order to ensure that the business partner has the same understanding of the exchange format defined in the subset, the initiating business partner transmits the DIRDEF message to the responding business partner(7). The responding business partner imports the subset definition via the DIRDEF message into his own browser and editing tool (8a, 8b). On the one hand he is now able to browse the definition created by the initiating business partner on-line and on the other hand he is also able to print the same subset documentation or implementation guideline as the initiating business partner (9). If possible, the responding business partner also provides the subset definition via DIRDEF to his converter and creates the mapping tables (11). Note that the mapping tables of the initiating business partner are usually different from those of the responding business partner, because they need not use the same application software. Now, the process of modelling functional agreements according to the business needs of the partners leading to a subset of data formats which will really be exchanged is finished and business transactions based on this subset might be processed.

A business scenario based on the defined subset will be similar to every EDIFACT scenario. The application of the initiating business partner creates the data to be transmitted and writes it into the application database output format (12). This output format is provided to the converter (13) which loads the corresponding mapping table and creates the appropriate EDIFACT message (14) according to the consignee and kind of business transaction. The EDIFACT message is transmitted to the responding business partner (15). The responding business partner usually receives the message from his mailbox and processes it via his converter. According to the sender and kind of message mentioned in the message header the converter loads the appropriate mapping table (16) and creates the corresponding application database input format (17). Finally, the received data is imported into the application of the responding business partner (18) and processed. Usually the application of the responding business partner answers to the received message by sending a response message. The steps of the response message (19 - 25) correspond to those of the original message (12 - 18) except for the direction of the process. The business partner who creates the subset - here called initiating business partner - is not necessaryly the partner who starts the electronic exchange of messages in the defined format.

5 CASE STUDY: DEFINING THE FORMAT OF AN ORDER MESSAGE

We now analyse our proposed process of modelling functional agreements by means of a practical example. For this purpose we have chosen a purchase order message since it is the most widely used standard message of EDIFACT. The example is based on that of (Steel, 1994) who uses it for the analysis of business transactions: A large retailer with several stores replenishes stock from a wholesaler. A different purchase order is raised for each store, the invoice goes to

UNH	Message header	M	1
BGM	Beginning of message	M	1
DTM	Date/time/period	M	2
Segment Group 1		M	2
NAD	Name and address	M	1
LOC	Place/location identification	C	1
Segment Group 2		C	999
LIN	Line item	M	1
QTY	Quantity	M	1
MOA	Monetary amount	C	1
UNT	Message trailer	M	1

Figure 4 Message structure of an adapted ORDERS message

the head office. Furthermore, a delivery date which refers to the whole order can be stated. Possible variants in the semantics of the delivery dates are earliest delivery date, latest delivery date and expected delivery date.

Consider the retailer to be the initiating business partner. Therefore, he imports the standard directories from the EDIFACT reference databases. Via the browser and editor tool he creates a subset of an existing standard directory. In our case study he establishes an EDIFACT-conform subset of the EDIFACT directory D 93A, which will only include the adapted order message. Then the retailer selects the elements of the message structure according to the above specification. Like all EDIFACT messages the adapted segment structure starts with the UNH (message header) and the BGM (beginning of message) segment. Then exactly two DTM (date/time/period) segments for specifying the date of the order and the delivery date must be included. The following segment group 1 is used to describe the seller and buyer involved in the business transaction by the NAD (name and adress) segment. The LOC (place/location identification) segment will be used to specify the store of the retailer for which the order is raised. These all are segments used in the header information, the detailed information on each ordered product is modelled in segment group 2. For each ordered product the line number in the order and the product identification must be specified in the LIN (line item) segment. The subsequent QTY (quantity) segment must be used to indicate the ordered quantity of the product. Finally, the price of each ordered unit might be stated within the MOA (monetary amount) segment and as usual the message ends with the UNT (message trailer) segment. The resulting segment structure of the adapted purchase order message is depicted in Figure 4.

DTM Date/time/period
- C507 Date/time/period
  - 2005 Date/time/period qualifier
  - 2380 Date/time/period
  - 2379 Date/time/period format qualifier

NAD Name and adress
- 3035 Party qualifier
- C082 Party identification details
  - 3039 Party identification
  - (1131 Code list qualifier)
  - (3055 Code list responsible agency)

LOC Place/location identification
- C517 Location identification
  - 3225 Place/location identification
  - (1131 Code list qualifier)
  - (3055 Code list responsible agency)
  - (3224 Place/location)

LIN Line item
- 1082 Line item number
- C212 Item number identification
  - 7140 Item number
  - 7143 Item number type
  - (1131 Code list qualifier)
  - (3055 Code list responsible agency)

QTY Quantity
- C186 Quantity details
  - 6063 Quantity qualifier
  - 6060 Quantity
  - 6411 Measure unit qualifier

MOA Monetary Amount
- C516 Monetary amount type qualifier
  - 5025 Monetary amount type qualifier
  - 5004 Monetary amount
  - (6345 Currency, coded)
  - (6343 Currency qualifier)
  - (4405 Status)

Figure 5 Segment structure of an adapted ORDERS message

In the next step the data element structure of the remaining segments is manipulated. As each EDIFACT message usually starts with UNH and BGM, which are used to identify the message itself, these two segments are not changed in any way. Also the DTM segment is composed of only one composite data element (C507 Date/Time/Period) which comprises the date/time/period qualifier (2005), the data/time/period itself (2380) and the date/time/period format qualifier (2379). Since both partners know the unique identification of each other, the NAD segment retains the first two data elements, namely the party qualifier (3035) and the party identification details (C082), whereas the rest of the structure needed to state the whole address is deleted. Similarly the LOC segment is reduced to known location identification (C517) which indicates the place of delivery. The composition of the LIN segment is reduced to solely the line item number (1082) and the item number identification (C212). The following QTY segment is equal to the standard one and only includes the quantity details which are built by the quantity qualifier (6063), the quantity (6060) and the measure unit qualifier (6411). The MOA segment too will include only one composite data element, the monetary amount (C516). The ending UNT segment is also identical to the standard one. The resulting data element structure of the segments is presented in Figure 5.

Having again called the synchronisation method, it is necessary to adjust the remaining composite data elements to the business needs. Those simple data elements which are listed in round brackets in Figure 5 are deleted from the simple data element structure of the corresponding composite data element. Since both involved parties know their identification, there is no need to specify the underlying code list and hence the code list qualifier (1131) and the code list responsible agency (3055) can be erased from the structure of party indentification details. Similarly, the seller knows exactly by virtue of the place/location identification (3225) to deliver to which store of the buyer, and consequently the two data elements mentioned before and the uncoded place/location (3224) can also be deleted from the location identification (C517). The same is true for item number identification (C212). Inasmuch as both companies use their national currency (e.g. US$) a specification of the same is not necessary within monetary amount type qualifier (C516) and therfore the currency (6345) and the currency qualifier(6343) can be ereased. Furthermore, the partners ommit to specify the status (4405) of the monetary amount.

2005	Date/time/period/qualifier
2	Delivery date/time, requested
4	Order date/time
63	Delivery date/time, latest
64	Delivery date/time. earliest
2379	Date/time/period format qualifier
101	YYMMDD
103	YYWWD

3035	Party qualifier
BY	Buyer
SE	Seller
7143	Item number type
EN	Int. Article Numbering Association (EAN)
VP	Vendor’s part number

6411	Measure unit qualifier
EA	Piece
KG	Kilogram
6063	Quantity qualifier
21	Ordered quantity
82	Quantity requirement for sampleinspection
5025	Monetary amount type qualifier
178	Exact Amount
179	Maximum Amount

Figure 6 Data element structure of an adapted ORDERS message

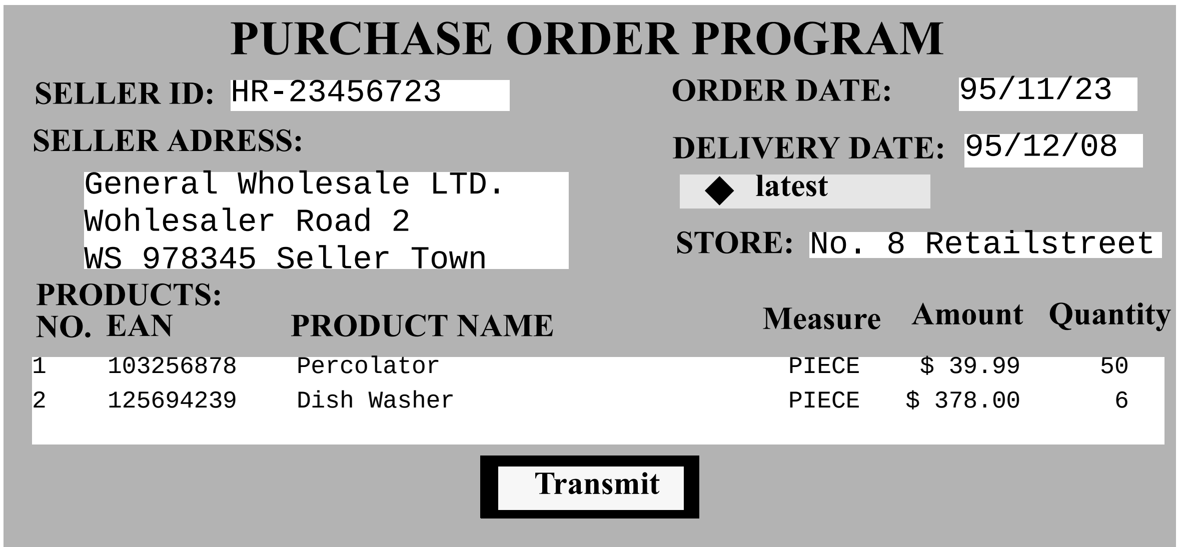
An invocation of the synchronisation method now results in a complete reduction to the number of required segments, composite and simple data elements of the purchase order message. The last required step is to fit the codes of the coded simple data elements to only those ones which might be used in the transmission. Since the indicated date must either be the date of the order (4) or one of the delivery dates with the semantics requestes (2), latest (63) or eraliest (64), the codes assigned to the date/time/period qualifier can be limited to these four. Furthermore, a date is always indicated in the order year, month, day (101) or year, week, day (103), which are the two possible codes for the date/time/period format qualifier (2379). The only codes for the party qualifier (3035) are the invoved business partners, the buyer (BY) and the seller (SE). To uniquly indicate an ordered item the international article number (EN) or the seller’s part number (VP) which are the legal codes for the item number type (7143) might be used. The ordered quantity is either measured in pieces (EA) or in kilograms (KG), which are in the codelist for the measure unit qualifier (6411). Furthermore, via the codes ordered quantity (21) and quantity requirement for sample inspection in the quantity qualifier (6063) the purpose of the order might be indicated. Finally, the buyer can indicate whether the amount indicated for each item is an exact amount (178) or a maximum amount (179) via the possible codes of the monetary amount type qualifier (5025).

Having finished these procedures and documented the changes in the corresponding textual parts, the needed subset is completely created. Now the retailer can print the documentation of the subset via the documentation tool and create the DIRDEF message of the subset. The resulting DIRDEF message without the documentation in the FTX (free text) segments is depicted in Figure 7.

```
UNH+1+DIRDEF:0:36:UN'
DII+D+93A++JRC'
MSG+ORDERS:D:93A:JRC+PURCHAS
E ORDER MESSAGE+2‘
SGU+UNH+M+1+1+0010‘
SGU+BGM+M+1+1+0020'
SGU+DTM+M+2+1+0030‘
SGU+NAD+M+1+1+0050'
GRU+1+M+2++0040'
SGU+LOC+C+1+2+0060‘
SGU+LIN+M+1+1+0080'
GRU+2+C+999++0070'
SGU+QTY+M+1+2+0090'
SGU+MOA+C+1+2+0100
SGU+UNT+M+1+1+0110'
SEG+BGM+BEGINNING OF
MESSAGE'
SEG+DTM+DATE/TIME/PERIOD'
ELU+C507+M+010'
SEG+LIN+LINE ITEM'
ELU+1082+M+010'
ELU+C212+M+020'
SEG+LOC+PLACE/LOCATION IDEN-
TIFICATION'
ELU+C517+M+010'
SEG+MOA+MONETARY AMOUNT'
ELU+C516+M+010'
SEG+NAD+NAME AND ADRESS'
ELU+3035+M+010'
ELU+C082+M+020'
SEG+UNH+MESSAGE HEADER'
SEG+UNT+MESSAGE TRAILER'
CMP+C082+PARTY IDENTIFICATION
DETAILS'
ELU+3039+M+010'
CMP+C186+QUANTITY DETAILS'
ELU+6063+M+010'
ELU+6060+M+020'
ELU+6411+M+030'
CMP+C212+ITEM NUMBER IDENTIFI-
CATION'
ELU+7140+M+010'
ELU+7143+M+020'
CMP+C507+DATE/TIME/PERIOD'
ELU+2005+M+010'
ELU+2380+M+020'
ELU+2379+M+030'
CMP+C516+MONETARY AMOUNT'
ELU+5025+M+010'
ELU+5004+M+020'
CMP+C517+LOCATION IDENTIFICA-
TION'
ELU+3223+M+010'
ELM+1082+3+V+6++Line item
number+2'
ELM+2005+2+V+3++Date/time/period
qualifier+1'
ELM+2379+2+V+3++Date/time/period
format qualifier+1'
ELM+2380+2+V+35++Date/time/
period+2'
ELM+3035+2+V+3++Party qualifier+1'
ELM+3039+2+V+17++Party identifica-
tion+2'
ELM+3225+2+V+25++Place/location
identification+2'
ELM+5004+3+V+18++Monetary
amount+2'
ELM+5025+2+V+3++Monetary amount
type qualifier+1'
ELM+6060+3+V+15++Quantity+2'
ELM+6063+2+V+3++Quantity quali-
fier+1'
ELM+6411+2+V+3++Measure unit quali-
fier+1'
ELM+7140+2+V+35++Item number+2'
ELM+7143+2+V+3++Item number
type+1'
VLI+2005'
CDV+2+Delivery date/time, requested'
CDV+4+Order date/time'
CDV+63+Delivery date/time, latest
CDV+64+Delivery date/time. earliest"
VLI+2379'
CDV+101+YYMMDD'
CDV+103+YYWWD'
VLI+3035'
CDV+BY+Buyer'
CDV+SE+Seller'
VLI+7143'
CDV+EN+Int. Article Numbering Associa-
tion (EAN)'
CDV+Vendor?'s part number'
VLI+6411'
CDV+EA+Piece'
CDV+KG+Kilogram'
VLI+6063'
CDV+21+Ordered quantity'
CDV+82+Quantity requirement for sample
inspection
VLI+5025'
CDV+178+Exact amount'
CDV+179+Maximum amount'
UNT+...
```

Figure 7 Resulting DIRDEF message describing an adapted ORDERS message

If this DIRDEF message is transmitted to the wholesaler, both business partners import it into their converter and specify the mapping tables to their applications according to the semantics of the business transaction. They can now start to exchange messages based on the subset format. Consider now the example presented in figure 8. The retailer uses his purchase order program to order 50 percolators for $ 39.99 per piece and 6 dish washers for $ 378 per piece from the wholesaler. He further expects the goods to arrive at latest on the 12th December 1995. By activating the 'Transmit'-Button the order is translated into the application database output format, which is mapped to an EDIFACT message by the converter according to the mapping table. The resulting EDIFACT purchase order message depicted in figure 9 is conform to the subset created in the functional interchange agreement. The wholesaler receives the order from is mailbox and imports it through his converter into its incoming order program. Yet, the wholesaler can apply his usual business procedures on the order. Normally, he would respond to the order by an EDIFACT order response message. But this scenario is beyond our tutorial example.


PURCHASE ORDER PROGRAM

SELLER ID: HR-23456723

ORDER DATE: 95/11/23

SELLER ADRESS:

General Wholesale LTD.
Wohlesaler Road 2
WS 978345 Seller Town

DELIVERY DATE: 95/12/08

◆ latest

STORE: No. 8 Retailstreet

PRODUCTS:

NO.	EAN	PRODUCT NAME	Measure	Amount	Quantity
1	103256878	Percolator	PIECE	$ 39.99	50
2	125694239	Dish Washer	PIECE	$ 378.00	6

Figure 8 Example of an order

```
DTM+4:951123:101'
DTM+63:951208:101'
NAD+BY+JC-84756302'
LOC+8'    Fig. 8
NAD+SE+HR-23456723'
LIN+1+103256878:EN'
QTY+ 21:50:EA'
MOA+178:39.99'
LIN+2+125694239:EN'
QTY+21:6:EA'
MOA+178:378.00'
```

Figure 9 Resulting ORDERS message of the order

6 SUMMARY

In this paper we presented a method for modelling functional agreements in EDIFACT environments. The core of our method is a tool which allows on the one hand a flexible browsing through and on the other hand it opportunity to build EDIFACT-conform and wild subsets of the EDIFACT standard versions. Using this tool it is possible to adapt a standard message design to a structure which corresponds to the real interchange format used by the business partners. Since the ‘real’ interchange is usually based on a very small subset of the generally valid standard, the complexity is reduced. Consequently, the generated subsets are easier to read and to understand. Furthermore, the tool also includes a component which offers a facility to create, in addition to the paper documentation, a description of the created subset in DIRDEF format. DIRDEF is a kind of EDIFACT meta-message, which is used to describe the format of EDIFACT messages in EDIFACT syntax. Since we anticipate that in the near future the EDIFACT converters will be equipped with an DIRDEF-import interface, a real data interchange based on the self-created subsets might be possible.

7 REFERENCES

Berge, J. (1989) The EDIFACT Standards. NCC Blackwell Limited; Oxford

United Nations, Economic and Social Council (1993) Trade Data Interchange Protocols Directory Definition Message. TAG 52.315, International Organization for Standardization, Geneva

United Nations, Economic and Social Council (1988) UN/EDIFACT Syntax Rules. ISO 9735, International Organization for Standardization, Geneva

Hill, N.C. and Ferguson, D.M. (1989) Electronic Data Interchange: A Definition and Perspective. EDI Forum: The Journal of Electronic Data Interchange, Vol. 1

Steel, K. (1994) Matching Functionality of Interoperating Applications. ISO/IEC JTC1/SC30 IT11/7/94-108