SECTION V - Design principles

1. Approach

Every IE needs to be in a structure that conforms to this guideline (TMS). The TMS needs to be formatted in an EDIFACT, as specified within this guideline in section VI.

The formatted message needs to be transported across the national network, or across the Internet, according to the rules laid out in this document in section VI.

Because IEs are used to update data of Transit operations held by different applications, the data needs to be uniquely identifiable. Not all data is uniquely identifiable. Therefore, the following rules are applied to updates of Transit operation data:

- Key fields. The MRN is a key to the Transit operation and each goods item is uniquely identified by its goods item number within an MRN:
    - If information regarding an MRN needs to be changed, the MRN identifies the Transit operation.
    - If information regarding a particular goods item needs to be changed, the goods item number of that particular goods item is exchanged, together with the changed information.
- Non-key fields. That information that is not uniquely identifiable, e.g. with an MRN or goods item number, is completely exchanged and replaces the information that has already been exchanged. For instance, if a new goods item needs to replace an existing goods item, the existing goods item number with the new information replaces the previously exchanged information of that goods item.

2. Character Sets and data item conventions

2.1. Data item conventions

Every data item within a TMS will be either a numerical field or a text field. A number of rules and conventions have been defined for the possible data formats. These rules are the same for data exchanged in EDIFACT format, on the WEB and in the SADBEL format.

2.1.1. Numerical fields

Concerning numerical fields, it should be noted that these are either a cardinal value (positive integer value), or a decimal value.

The decimal separator is the decimal point ".". No other symbols are permitted as decimal separator.

Triad separators, such as a comma, shall not be used.

Signs, whether positive or negative shall not be used (all values are always intrinsically positive).

For decimal values, the decimal notation (with the decimal point) should only be used when there is a reason to indicate precision.

E.g., for a mass value:

- 89 kg, with a precision of 1 kg
- 89.2 kg, with a precision of 0.1 kg
- 89.20 kg, with a precision of 0.01 kg

For numerical values, leading zeroes shall not be used. Trailing zeroes should only be used to indicate precision.

If the decimal point is present, at least one digit shall be present before the decimal point.

If the decimal point is present, at least one digit shall be present after the decimal point.

Examples for a n..11,3 type.

- 12345678.123 (valid)
- 123456789.123 (invalid, too many digits before decimal point and too many digits in total)
- 12345678.1234 (invalid, too many digits after decimal point and too many digits in total)
- 0123 (invalid, leading zero not permitted)
- +123 (invalid, plus sign not allowed)
- -123 (invalid, minus sign not allowed)
- 1,234 (invalid, triad separator not allowed)
- .3 (invalid, no digit before decimal point)
- 12345. (invalid, no digit after decimal point)
- 0.3 (valid)
- 1.3E1 (invalid, only digits and decimal point allowed)
- 12345678901 (valid, n..11,3 can have maximally 11 digits of which maximally 3 after decimal point).

To be noted is that the rules above also apply to numerical values within code lists. Values in such a list should always be stored without leading zeroes (in order to avoid problems of comparing e.g. a value of 60 against a value of 060). If the leading zeroes are indeed omitted, a comparison should always work out fine, regardless of the fact if the comparison was done on a numerical or character basis.

To be noted is that there are no code lists with decimal values.

2.1.2. Text fields

Leading and trailing spaces shall not be used within text fields.

The EDIFACT separator characters (see section VI) can be used within such a field. The EDIFACT release character (?) can be used to include the separator characters in fields.

2.2. Character set usage

EDIFACT only supports byte based character sets.

2.2.1. Exchanges in EDIFACT format

Text fields can be either language-sensitive (with an associated LNG field), or not.

For language-sensitive text fields as well as non-language sensitive text fields and numeric fields, only the UNOC character set can be used: UNOC: Latin-1, ISO 8859-1 with the exception of for instance the Dutch IJ.

3. Exception Handling

3.1. Introduction

An exception is the generic term used to refer to any behaviour of one or more system components of the NCTS that is not in accordance with the specification given in this guideline.

There are three possible error notification mechanisms:

- **Functional errors**: a functional message is not filled according to its FMS and rules (e.g. missing functional data item or wrong value). A Functional NACK (IE906: C_FUN_NCK) is sent by the party detecting the error to the party that has sent the erroneous functional message. The following are also used to report functional errors in the External Domain:
  - Arrival notification rejection (IE08: E_ARR_REJ)
  - Declaration rejected (IE16: E_DEC_REJ)
  - Unloading remarks rejection (IE58: E_ULD_REJ)
  - Amendment rejection (IE05: E_AMD_REJ)
  - Release request rejection (IE62:E_REQ_REJ).
- **UN/EDIFACT errors**: a UN/EDIFACT interchange and its UN/EDIFACT message(s) is not filled according to its specification given in section VI. A UN/EDIFACT NACK (IE907: CONTRL) is sent by the party detecting the error to the party that has sent the erroneous interchange.
- **Communication errors**: some error occurs during the exchange of messages .

This section only covers exceptions regarding the exchange of UN/EDIFACT messages and their functional message structure

3.1.1. Additional mechanisms for protecting against loss of messages due to some error cause specified in the next section.

The mechanisms specified in this section are based on the following assumptions:

1. Fallback and recovery procedures are outside the scope of this guideline. In principle, every action related to IEs needs to be logged to allow recovery and identification of a failed component. Furthermore, OTS can serve as a fallback procedure in case of failure in message exchange.
2. There is a layered approach to error detection upon reception of information. It consists of the following three layers:
   - Communication errors are handled by communication software
   - UN/EDIFACT layer: syntax errors are detected in addition to the ones detected by the communication software.
   - Functional layer: functional errors are detected on top of those detected by the UN/EDIFACT and communication layers.
3. Security functions in the External Domain.
4. UN/EDIFACT is used as specified in section VI, e.g. one UN/EDIFACT interchange contains one UN/EDIFACT message.
5. The FMS is the basis for identifying functional errors. Although the FMS is assumed that data group sequencing of sender and recipient is identical. Sequencing provides an easy mechanism for an error pointer.

3.1.2. Examples of error causes

Errors can be caused for the following reasons:

Error cause	Description
Failure of components	A distinction is made between three types of failures: • failure of an application; • failure of the network; • failure of the link between an application and the network. Failures will cause errors in message sequencing; e.g. one of the applications involved did not receive a message and has not been able to produce the proper answer. This may cause applications to be out of sync.

Error cause	Description
Software bug	A function of a receiving or sending application does not reach its proper state, because a required function is missing, incomplete, or incorrect. Errors may be detected in another function; e.g. the recipient of a message may detect errors. For example, a software bug can cause the production of a message of an incorrect type (message sequencing) and/or the incorrect content of a message. The sending application is then likely to be out of synch with the receiving application.
Human mistake	A mistake is a wrong action due to incorrect human intervention; e.g. another MRN is selected than was intended. A human mistake can cause incorrect contents of a message and incorrect sequencing of a message.
Incorrect code list	Improper code lists are used by a sending or receiving application, e.g. a sender uses an outdated code list when preparing a message or a recipient does the same when verifying the data of a received message. An incorrect code list will produce incorrect contents of a message.

Table 15: Error causes

3.2. Scenarios for exception handling

3.2.1. General procedure

In general, all errors must be logged upon their detection. Depending on their circumstances, one of the following scenarios has to be initiated:

- Exchange of functional errors (E_AMD_REJ, E_ARR_REJ, E_DEC_REJ, E_ULD_REJ, E_REQ_REJ and C_FUN_NCK).
- Exchange of UN/EDIFACT errors (CONTRL-IE907).

3.2.1.1. Functional errors

Every IE exchanged can contain functional errors, e.g. an MRN or a required data item is missing or has a value that is not allowed, or a data item is not allowed to have a value due to a specific rule. The figure below shows an example of a functional error to the reception of an AAR. The possible values of these errors are specified further in this section.

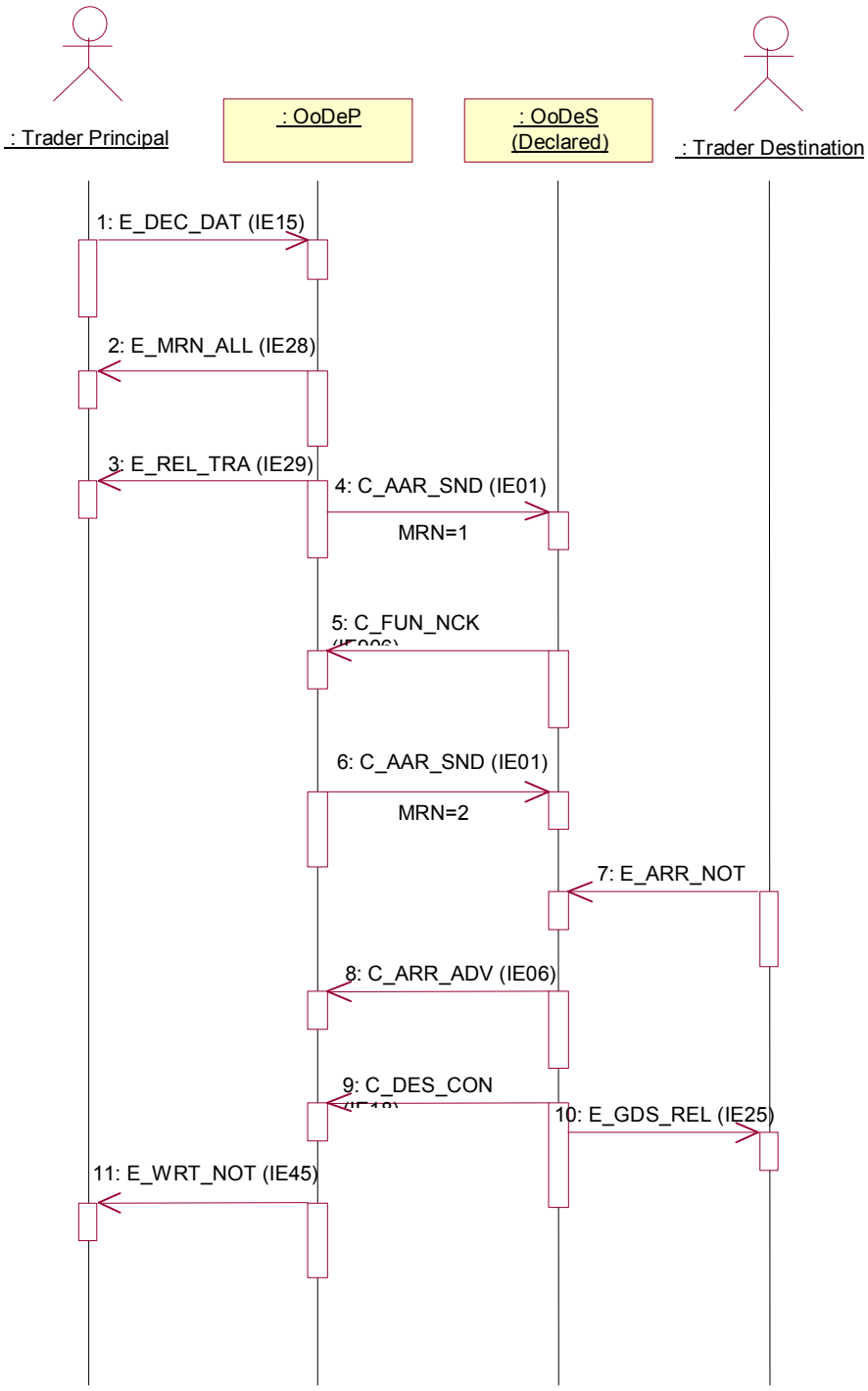

Figure 38 - Functional error across the Common Domain

This figure shows the detection of an AAR with an MRN that already exists (MRN=1). This MRN is sent again for another movement; it causes a rejection of that AAR by the Office of Destination with a C_FUN_NCK. The correct MRN (2) is sent afterwards.

With respect to UN/EDIFACT, if messages are resent after correction of an error, the interchange in which they are resent requires a new interchange reference in the UNB segment otherwise a duplicate will be detected by an EDIFACT translator and a CONTRL with error code '26' is exchanged. As the message has not yet been processed by the receiving application, that latter application will not detect a duplicate.

3.2.1.2. UN/EDIFACT errors

Every interchange exchanged can contain UN/EDIFACT errors, e.g. a missing segment or use of a syntax version that is not allowed. The figure below shows the exchange of a UN/EDIFACT CONTRL message after detection of an error in an interchange. The original interchange and message within the interchange are referred to in the CONTRL. The possible values of these errors are specified later in this section .

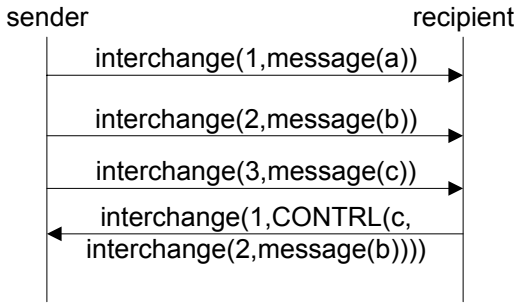

Figure 39 - UN/EDIFACT error

The figure shows the exchange of an error detected in the interchange with reference '2' and message with reference 'b'. As this figure shows, the recipient returns an interchange with reference '1' containing a CONTRL message that refers to the original interchange in which the error has been detected.

This use of reference numbers of interchanges and messages is arbitrary, as long as an interchange reference is unique between a sender/recipient pair and a message reference is unique within a Transit Movement identified with its unique MRN (Section VI).

A CONTRL message carries its own interchange and message reference, in the figure '1' and 'c' respectively. The reference to the interchange and message, in which an error has been detected, is exchanged in the UCI and UCM segments respectively.

Figure 39 only shows a sending and receiving role of an organisation, whereas an organisation can have both roles at the same time. Therefore, this figure is a simplification of the actual communication between two organisations.

3.3. Error codes

This section presents a table of error codes that can be used by the functional error messages (IE05, IE08, IE16, IE58, IE62 and IE906) and the UN/EDIFACT CONTRL (IE907). The table is a subset of the generic table provided by UN/EDIFACT and its contents is based on the use of UN/EDIFACT for NCTS.

The functional and UN/EDIFACT error codes are specified in the next table. The values in the column 'UN/EDIFACT error segments' specify segment tags that are used in the UN/EDIFACT CONTRL message for exchanging errors (see section VI, usage of UN/EDIFACT CONTRL).

It is assumed that errors are detected by reception of a message, which implies that functional errors are specified in more detail than errors at UN/EDIFACT layer. FMS specify more detail on the functional level with respect to:

- Status of a data item: a UN/EDIFACT data element can be optional, whereas the related data item of an FMS is required.

- Code values: code values are specified at functional level, with the exception of those codes that are specific to UN/EDIFACT (e.g. qualifier values).
- Dependency rules: values of data items can be dependent on each other as specified by additional conditions.

The following table shows amongst others the following two columns:

- identifying the **UN/EDIFACT segment** for exchanging a particular error. If this column does not have an entry, the error is not a UN/EDIFACT error.
- identifying the **FMS** errors. If this column does not have an entry, the error is not an error at FMS level.

Code	Name	Description	UN/EDIFACT error segment	FMS
2	Syntax version or level not supported	The syntax version or syntax level is not in accordance with the one specified in this guideline	UCI	
7	Interchange recipient not actual recipient	The interchange recipient (S003) is different from the actual recipient.	UCI	
12	Incorrect (code) value	Value of an element in a message is outside the predefined domain or not part of the applicable code list. The following situations are covered: • Data type **differences**: the received data type is different from the expected data type, e.g. numeric expected and alphanumeric received. • Data type **constraint difference**: the data type is not within the constraints given to it. This covers the following three situations: – **Length** differences: the received data is too long; – format **constraints**: the received value is not within the expected format constraints e.g. the received date is outside the possible dates; – Code **value constraints**: the received value is not within the expected code value range. This type of error is applicable to all elements with code values, so it is also applicable to those elements that identify a message type.	UCI,UCM,UCD	X
13	Missing	A mandatory/required element is missing in the received data, e.g. a mandatory segment is missing or a required element like the MRN for the Arrival Advice (C_ARR_ADV) is not present.	UCI, UCM, UCS, UCD	X
14	Value not supported in this position (code value constraint)	Notification that a recipient does not support use of the specific value of an identified element in the position where it is used. This type of error refers to the use of an improper code value for a specific element.	UCI, UCM, UCD	X

Code	Name	Description	UN/EDIFACT error segment	FMS
15	Not supported in this position	An element is not allowed to have a value due to one of the following two reasons: • The element is not allowed to be present according to the message specification (FMS or UN/EDIFACT). • The element is not allowed to be present according to some additional condition (FMS), e.g. if type of packages has the value 'UNPACKED' then number of package can not have a value (condition C60).	UCS, UCD	X
16	Too many constituents	A segment or composite data element contains too many elements. This type of error can only be detected in field separated messages like UN/EDIFACT	UCI, UCD	
19	Invalid decimal notation	The decimal notation is not according to the decimal formatting standards	UCI, UCM, UCD	X
21	Invalid character(s)	One or more character(s) used in the interchange is not a valid character as defined by the syntax level indicated in UNB. This type of error can only occur in elements with alphanumeric data types, e.g. a text with strange characters.	UCI, UCM, UCD	
22	Invalid service character(s)	The service character(s) used in the interchange is not a valid service character as according to this guideline.	UCI, UCM, UCD	
23	Unknown interchange sender	The Interchange sender (S002) is unknown.	UCI	
26	Duplicate detected	The same interchange is received again. Duplication is detected by reception of an interchange reference that has already been received. Duplicate messages can be received at UN/EDIFACT level, because a message is uniquely identified within an MRN. Therefore, more than one UN/EDIFACT message can have the same reference in the UNH segment.	UCI	X
28	Invalid control reference	The control reference in UNB/UNH does not match the one in UNZ/UNT.	UCI, UCM	
29	Invalid control count	The number of messages/ segments does not match the number given in UNZ/UNT.	UCI, UCM	
32	Lower level empty	The interchange did not contain any messages.	UCI	
33	Invalid occurrence outside message	An invalid segment or data element occurred in the interchange, between messages. Rejection is reported at the level above.	UCI	
35	Too many repetitions	Too many occurrences of a segment or data group.	UCS	X

Code	Name	Description	UN/EDIFACT error segment	FMS
37	Invalid type of character(s) (data type differences)	Notification that one or more numeric characters were used in an alphabetic element or that one or more alphabetic characters were used in a numeric element.	UCD	X
38	Missing digit in front of decimal sign	One or more digits do not precede the decimal sign.	UCD	X
39	Element too long (length constraint)	Notification that the length of the element received exceeded the maximum length specified.	UCD	X
40	Element too short (length constraint)	Notification that the length of an element received is shorter than the minimum length specified.	UCD	X
90	Unknown MRN	The MRN of the received FMS is not known, whereas it is expected to be known. This type of error can not be detected in an AAR (C_AAR_SND)		X
91	Duplicate MRN	The MRN of the received FMS is already known and is therefore not unique according to the specified rules.		X
92	Message out of sequence	The message can not be processed, because the receiver is not in a proper state.		X
93	Invalid MRN	The structure of the MRN does not conform to specifications given in FTSS 3.1.		X

Table 16 - Error codes