Memo – Additional Command Schemas

EAN.UCC business documents are sent encapsulated within the 'DocumentCommand' construct in the XML. The 'DocumentCommand' is a command to the receiving entity directing it to operate on the enclosed document. The DocumentCommand is always used with business documents. However, for maintenance of existing documents we need additional commands. For this purpose, EAN.UCC provides 2 other constructs in the XML schemas:

- DocumentIdentificationCommand
- LinkCommand

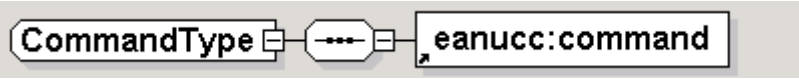
These commands have been constructed with the basic premise that for maintenance of existing documents, the entire document need not be repeatedly transmitted. Rather, these commands allow for the reference of existing documents using their unique identification. This package includes the 2 additional commands mentioned above. The 'DocumentCommand' is included with all Business Message Standard packages of standard EAN.UCC business documents. Information on EAN.UCC commands can be found in sections below.

1. COMMAND

Commands are used by a trading partner to instruct the receiving application about an action that should be performed on a given document. All commands are transitive and are discarded after executing the task.

Some of the actions to be performed are defined by the command itself, while other ones by an attribute of the command.

The 'command' is an abstract element, extended by the following elements:
'documentCommand'
'documentIdentificationCommand'
'linkCommand`



The Command Type structure

The substituting elements are defined in DocumentCommand.xsd, DocumentIdentificationCommand.xsd and LinkCommand.xsd.

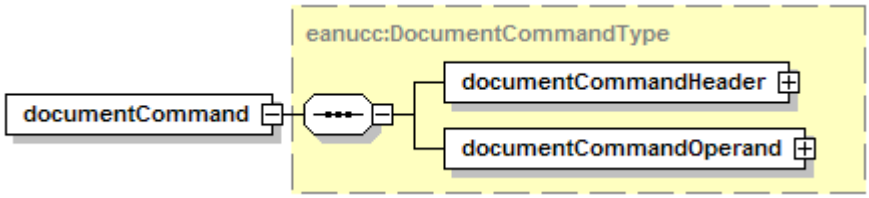
1.1 DOCUMENT COMMAND

The Document Command is used to instruct the recipient of that command to perform a particular action related to the documents within the command. The

document in question has to be present, i.e. it must be sent together with the command. The actions, which can be performed on it include:

- 'Add'
- 'Refresh'
- 'Correct'
- 'Delete'

The Document Command is an extension of the abstract Command. It contains two main components: the 'documentCommandHeader' and 'documentCommandOperand':

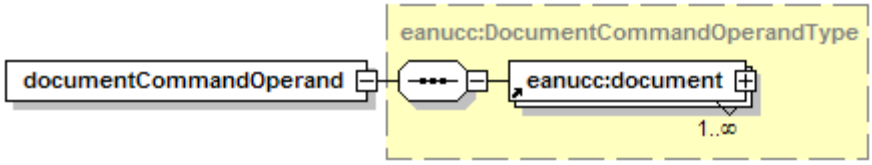


The Document Command structure

The 'documentCommandHeader' specifies the action that should be performed on the given document. Those tasks, defined as the required attribute of the header element, include:

ADD	– the receiving application is instructed to store the document or documents
CHANGE_BY_REFRESH	– the receiving application is instructed to update the existing document or documents, by total replacement
CORRECT	– the receiving application is instructed to update the existing document or documents, by total replacement, skipping certain business specific validation rules. The syntactical and content validation rules still apply. This command is used in cases where the specific validation rules would otherwise prevent the application from changing data. It can be used only if the correction does not impact the integrity of the corrected data. Otherwise, correction should be performed by sending two commands: DELETE (with the old document) and ADD (with the new document) in one transaction.
DELETE	– the receiving application is instructed to delete the document or documents

The 'documentCommandOperand' contains an abstract 'document' element that can be extended by the actual business document elements, defined in the business message schemas. Those are the documents upon which one of the actions defined in the header element should be performed.



The Document Command Operand structure

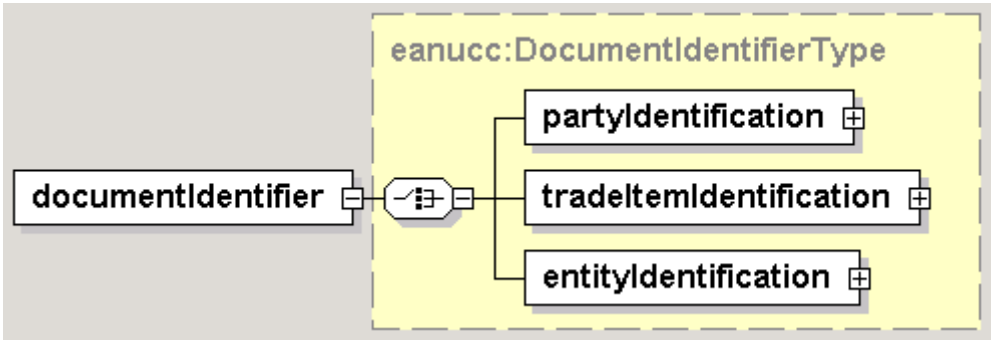
The list of the possible documents contains all the business messages defined within the given major version.

1.2 DOCUMENT IDENTIFICATION COMMAND

This is an extension of the abstract Command, defining operations that require just an identification of the relevant documents. The only operation currently supported by this method is DELETE, defined as a value of an attribute of the 'documentIdentificationCommand' element.

The Document Identification Command structure

The document on which the action should be performed is identified in the 'documentIdentifierList' element.



The Document Identifier structure

It can be specified in three ways:

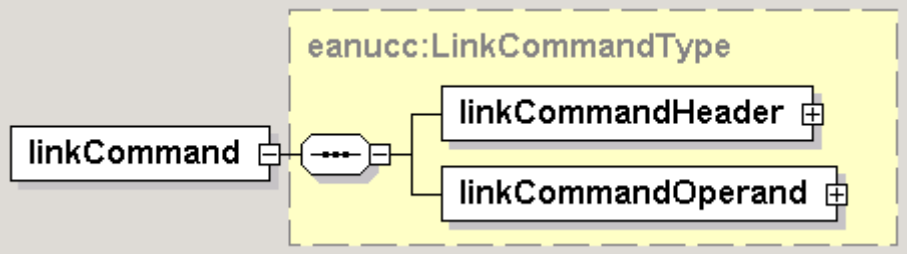
1. By 'partyIdentification' – using GLN and optionally, an additional identifier of the concerned party.
2. By 'tradeItemIdentification' – using GTIN and optionally, an additional identifier of the trade item concerned by the document in question.

3. By ‘entityIdentification’ - a combination of the unique document identifier assigned by its creator and the unique identification of that document creator (GLN and optionally, an additional identifier).

1.3 LINK COMMAND

This type of command allows to establish parent-child or peer relationships between several documents.

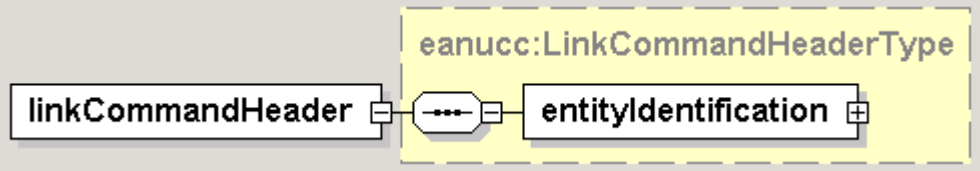
The Link Command consists of two major components: Link Header and Link Operand.



The Link Command structure

1.3.1 Link Header

The ‘linkCommandHeader’ specifies whether the function of the command is to link or unlink the documents concerned. They are defined as the enumeration values: LINK and UNLINK of the element’s attribute ‘LinkCommandListType’.

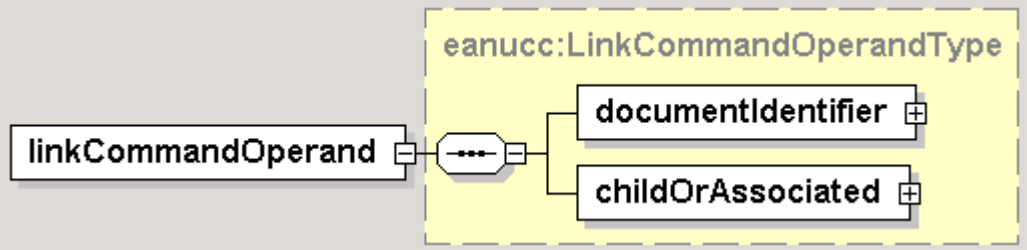


The Link Command Header structure

The header contains also the unique command identifier assigned by its creator, combined with the creator’s identification (GLN and optionally, an additional one).

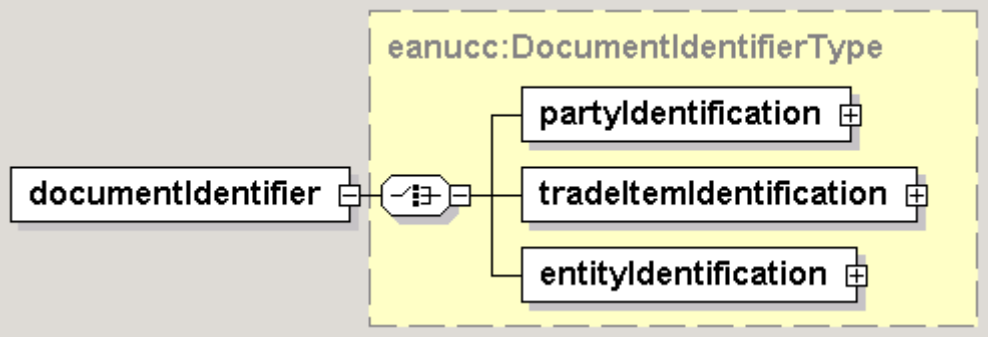
1.3.2 Link Operand

The ‘linkCommandOperand’ specifies the parent and children, identified by a document identifier.



The Link Command Operand structure

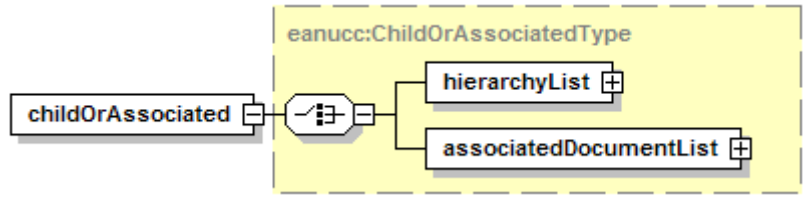
The document can be identified by an Entity Identification, a Party - identified by GLN and optionally, one of the additional identifiers, or Trade Item - identified by GTIN and optionally, one of the additional identifiers.



The Document Identifier structure

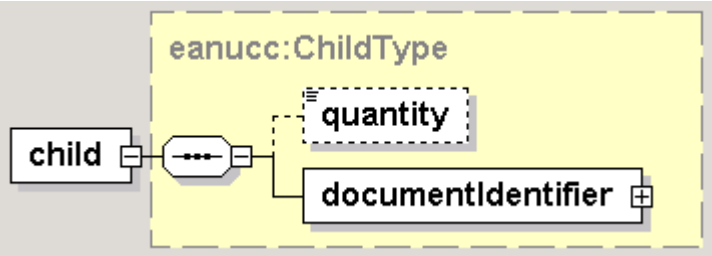
The first document identifier occurring in the sequence of Link Operand sub elements, defines the parent object.

The children objects can be identified using either a Hierarchy List or Associated Document List method.



The child or associated element structure

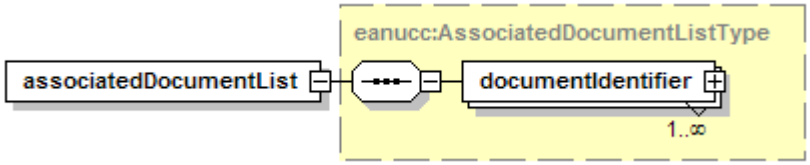
The 'hierarchyList' element allows to make a list of child documents, with just one document per child. Using this method it is also possible to specify the quantity of each child.



The child element structure

Therefore, the Hierarchy List can be used to specify the number of lower items in a packaging unit and the content of mixed containers.

The 'associatedDocumentList' allows to link some documents to a given parent.



The associated document list element structure

Within both parent and child elements, there is a choice of the components that are meant to be linked or unlinked. It can be ‘partyIdentification’, defining the trading partner that is to be linked to given information. The second option is the ‘tradeItemIdentification’, defining the product to be linked. The last possibility is to specify the document, using the ‘entityIdentification’.