XML Technology, Lab course : Exercise 03

A „MOM“ example: Modelling purchase orders

Designing your own DTD

Coding of a matching XML document incl. validation

Organisation

- Working directory:
  `~/lv/xmltech/03/`
- Filenames:
  `03-orders.xml`
  `03-orders.dtd`
  `03-orders.txt`
- Turn in:
  `03-orders.xml, 03-orders.dtd, 03-orders.txt`
- Tools:

```
emacs          # or X-Emacs
nsgmls         # implicitely used, via Emacs
firefox        # for additional tests
```

Organisation

- Sharing the workload:
  - This exercise affordes a high workload. Therefore you are allowed to work in pairs this time.
  - Save time by working in parallel! Ex.:
    One person takes notes in file 03-orders.txt, while the other builds 03-orders.dtd
    While file-tuning your DTD, already start working on the XML sample document in 03-orders.xml
  - Enter both names & student id's in your result files when working in teams.
  - *Nonetheless, each person should submit his/her own result files!*
- Deadline:
  - This time, two weeks from now (along with result of ex 04)

A: Data Modeling

- **Design a data structure for purchase orders**
  - Use elements, attributes and a proper element nesting!
  - Your document structure should match purchase orders of the type given below as a sample as closely as possible.
    Well, „as closely as possible" leaves a lot to be desired with our limited capabilities of DTDs. Please record all such limits and compromises you encounter in your protcol file 03-orders.txt!
  - You only need pencil and paper for this modeling. It is just a preparation for the next step: Coding a DTD.
  - Hint: Note that purchase orders are separated in **header** data, **item** data, and **summation** data.
  - Pay close attention to the detailed specifications below!

Sample: A simple purchase order (PO)

	Data of sample PO:
• **Header data**	
– PO number	1234567
– PO data	11/09/2005
– Requested delivery date	11/23/2005
– Buyer	Company name = XY Trade Inc
	GLN = 2100000000005
– Supplier	Company name = ABC Ltd.
	GLN = 2900000000001
• **Item data**	
– Item number	Pos. 1
– Article number / id	ISBN = 0-13-065198-2
– Quantity	45 pieces
– Item number	Pos. 2
– Article number / id	GTIN = 2900000200005
– Quantity	50 Liter
• **Summation data**	
– Number of items	2

A: Data Modeling

- **Specifications** for modeling (header data):
  - The document element name should be „**order**“
  - PO number (mandatory!):
    Max. 14 positions; 0-3 leading letters, remaining pos. are digits
  - Dates (PO date is mandatory):
    Support three different date formats:
    TT.MM.YYYY (DIN), MM/TT/YY (US), YYYYMMTT (ISO)
    Model three date functions:
    „PO_date“, „requested“ and „latest“ delivery date, resp.
    Hint: See lecture section on NOTATION
  - Trading partner:
    „buyer“ and „supplier“ are mandatory, „ship-to“ is optional
    Ident numbering systems: Suppor „GLN“ and „D-U-N-S id“ !
    (GLN: 13 digits incl. Check digit, DUNS: 9 digits incl. check digit)
    „name“ of trading partners is optional, ident number is mandatory

A: Data Modeling

- **Vorgaben** zur Modellierung (Position):
  - Bestellpositionen (Mussfeld!):
    Positionen sollten aufsteigend nummeriert werden
    Lückenlose Nummerierung ist nicht gefordert, aber max. 999 Pos.!
  - Artikelnummer (Mussfeld!) - 2 Identsysteme unterscheiden:
    GTIN: 8-,12-, 13-, oder 14-stellig, nur Ziffern, letzte ist Prüfziffer
    ISBN: Immer 10-stellig, plus 3 Bindestriche bzw. Leerzeichen
    Erstes Zeichen ist immer eine Ziffer, letztes ist Ziffer oder ‚X'.
    Bindestriche/Blanks immer mit Ziffer(n) trennen, nie mischen.
  - Mengenangabe (Mussfeld!):
    2 Fälle unterscheiden: Bestellte Menge, Menge ohne Berechnung
    Einheiten: Stück, Liter und Kilogramm vorsehen
    Default-Einheit sei das Stück
  - Artikelbeschreibung (optional):
    Datenelement für Artikelbeschreibung (max. 35 Z.) vorsehen!

A: Data Modeling

- Suggested steps
  - Identify all relevant data elements
    Including statements about …zur Kennzeichnung von Bedeutungen
  - Decide which data elements to model as XML elements and which as XML attributes
    Keep in mind that restricting specifications should ideally be implemented in such a way that XML validation is able to catch violations of such restrictions
  - Issue reasonable (and valid) names for your elements and attributes; mind the 10 XML design goals
  - Build a suitable hierarchy of elements that matches the given document structure

B: DTD design

- Now encode your data model as a DTD
  - Create file **03-orders.dtd**
- Then check which of the given restrictions are not modeled by your DTD
  - Think again if some restrictions might indeed be met by the means available to DTDs if you review your DTD (within limits – don‘t overdo things!)
  - If possible, improve your DTD accordingly
- Hint:
  - Many restrictions cannot be properly modeled by the DTD means!

C: XML document

- Now create an XML sample document according to this:

  Create file **03-orders.xml** with following features:
  - The content should be an instance of document type „order“ as specified by your DTD.
  - It should contain the data of the sample PO listed above!
  - In addition, add an article description (text of your choice) to one of the line items
  - Include the DTD of 03-orders.dtd in your document.
- Now validate your document against your DTD!
  - Only „valid“ data my be turned in as solution.

D: The limits of DTDs

- Which specifications/restrictions
  a) are not at all subject to validation tests?
  b) can only be partly checked by validation?
  c) could be modeled and checked precisely?

- Now that you noticed some limitations of DTD based modeling – what would you require from an improved substitution of DTDs?

- Write your answers into file **03-orders.txt**
  – Example: one line of text per data element

Final remarks

- This exercise matches real-world requirements !
  – Data modeling and DTD design are typical procedures
  – The given PO data closely resemble real-world EDI data

- This exercise affords a high degree of activity on your own initiative.
  – The task is designed to let you discover the capabilities and limitations of DTDs as well as one‘s room for own decisions.
  – Don‘t expect complete partial solutions from the lecture parts. However, you can find all necessary building blocks for your solution!
  – Be prepared to explore several solution approaches. It is unlikely that you find the best approach immediately.
  – When in doubt, decide on your own account. Just keep track of your reasoning behind the decisions.

- Outlook
  – We‘ll return to the PO scenario when learning about XML Schema!