Why All of Humanity Does Not Speak Esperanto.

XML Prague

Václav Trojan, Jiří Kamenický, Jiří Měska

email: info@syntea.cz
http://xdef.syntea.cz

Annotation:

The paper is concentrated on tools available for formal definitions of XML languages (XML control data). We focus primarily on the requirement of usability of XML tools during the whole lifecycle engineering, which includes consultation with partners, analysis, implementation, support and requests for changes. The principal requirements from the XML formal definitions point of view are **comprehensibility** for all participants of the development lifecycle and at the same time **obligation**. Obligation means usability of XML formal definition directly for computer processing. In our paper we introduce draft Xdefinition. Xdefinition is the technology which Syntea software group Ltd. is still developing.

Motivation: XML's Task in a Constantly Changing World

What are the challenges that the development of extensive information systems (IS) must face today? There certainly are a lot of these challenges, but we would like to mention one which in our opinion has a deep impact on requirements for XML technology. This challenge is IS development and management in an **ever changing world**. From this perspective, it is not just that IS is not ever **complete**, but it is also not ever finally **defined**. With the increased tempo, not only the technological environment, but also the real problem of the actual solution changes. Information support is reaching stage, where it is closely linked to the constantly changing social situation, which can be something like the link to legal standards or adaptation to technological standards.

Pressure on **flexibility** and **robustness** of applications comprising IS is also increasing with the increasing pressure from an ever changing world. The requirement for flexibility or adaptability of applications is typically displayed by the shift from hardcoded logic to data control. These, then, become a parameter of the system, but only under the presumption that they are comprehensible on global level, not only the system level of programmers. In the contribution's conclusion we touch on the topic of IS robustness, in connection with the number of independent places, which must be changed simultaneously to achieve changes in the control data of the application.

But let's return to the topic of XML. From the technological perspective XML is becoming, thanks to its **simplicity and at the same time the structure** of the generally used data format. XML found its position:

- as a tool for structured data exchange,
- as a tool for parametrization and application management,
- as a tool for interface description (web services)
- as a tool for business process description (BPEL)
- and many others.

The spectrum of usebility leads to the situation that XML documents participate in all IS development phases. From this follow the basic criteria (requirements) which we set for ourselves for the description of XML documents.

- descriptions must be **understandable** for use in the following areas of IS creation
    - discussions with partners
    - analysis
    - code creation
    - documentaristics
- descriptions must be **obligatory** in the sense that they are machine processable for the needs
    - of data validation
    - of process data processing.

Besides these basic philosophical criteria, it is necessary to stipulate what factually XML documents must contain:

- Description of the structure of XML documents
- Description of the types of elements and attributes
- Definition of integrity limitations
- Relation to process data processing.

The answers of Syntea a.s. to these requirements is an Xdefinition concept and development of resources for supporting work with Xdefinitions. Details are available on http://xdef.syntea.cz

Example: Language for the Description of Traffic Accidents

XML is in itself merely a data format. As a result of this there are specific XML languages (mark up languages, description of the XML document), whether for the needs of subject resolution (such as traffic accident description) or for technological needs (XML Schema, BPEL etc.). In both cases it is necessary to define at least:

1) syntax of the language
2) semantics of the language and limiting conditions.

Let's start with an example, where our task is to create IS for traffic accident (hereinafter just TA) processing. The first step in working on a system like this is to define the TA conceptual model.

The following document is a fragment of a TA report from 5.5.2007 5:00, when two vehicles crashed, a **Škoda Oktávie** and a **Trabant**. The first vehicle was driven by Franta Škvor and his wife, Filipína Škvorová, was sitting in the passenger seat. The second vehicle was driven by Slavomil Tichý. The traffic accident was recorded under case number KRP-P-93/KDI-LM-2007.

```
<?xml version="1.0" encoding="windows-1250"?>
<Accident DateTime ="8.5.2007 05:00"
          RefNum="KRP-P-93/KDI-LM-2007">
 <Descriprion>
The vehicle BB-862 SK moved on main street with right of way near ...
 </Descriprion>
  <Vehicle Brand="Škoda Oktavie" RegNum="SK-262 AK">
    <Person Role="driver">
      <Individual Name="Franta" Surname="Škvor" PIN="5510116722"/>
   </Person>
   <Person Role="passenger">
     <Individual Name="Filipína" Surname="Škvorová" PIN="5860136722"/>
   </Person>
  </Vehicle>
  <Vehicle Brand="Trabant" RegNum="BB-862 SK">
    <Person Role="driver">
      <Individual Name="Slavomil" Surname="Tichý" PIN="8001010463"/>
    </Person>
  </Vehicle>
</Accident>
```

Two alternative descriptions of the traffic accident using the XML Schema and Xdefinitions demonstrate two different solutions of the following dilemma:

- if it is necessary to use a new mark up language for the description of the client mark up language (XML document)
- or to try to find a way to write the description in the actual client marking language

XML Schema

```
<xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="Accident">
  <xs:complexType mixed="true">
   <xs:sequence maxOccurs="unbounded">
     <xs:element ref="Description"/>
     <xs:element ref="Vehicle"/>
   </xs:sequence>
   <xs:attribute name="DateTime" type="xs:string"
       use="optional"/>
   <xs:attribute name="RefNum" type="xs:string"
       use="optional"/>
  </xs:complexType>
 </xs:element>
 <xs:element
    name = "Description"type="xsd:string" />
<xs:element name="Vehicle">
  <xs:complexType>
   <xs:sequence maxOccurs="unbounded">
    <xs:element ref="Person"/>
   </xs:sequence>
   <xs:attribute name="Brand" use="required">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:maxLength value="80"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:attribute>
   <xs:attribute name="RegNum" use="required">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:maxLength value="12"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:attribute>
  </xs:complexType>
 </xs:element>
 <xs:element name="Person">
  <xs:complexType>
   <xs:sequence>
    <xs:element ref="Individual"/>
   </xs:sequence>
   <xs:attribute name="Role"
       use="required">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:enumeration value="passenger"/>
      <xs:enumeration value="driver"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:attribute>
  </xs:complexType>
 </xs:element>
 <xs:element name="Individual">
  <xs:complexType>
   <xs:attribute name="PIN" type="xs:long"
       use="required"/>
   <xs:attribute name="Surname" use="required">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:minLength value="1"/>
      <xs:maxLength value="36"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:attribute>
   <xs:attribute name="Name" use="required">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:minLength value="1"/>
      <xs:maxLength value="24"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:attribute>
  </xs:complexType>
 </xs:element>
</xs:schema>
```

Xdefinition

```
<xd:def xmlns:xd="http://www.syntea.cz/xdef/2.0"
        root="Accident"
        name="TraficAccident">

<Accident DateTime="required datetime('d.M.y H:m')"
          RefNum  ="optional num(10)">
  <Description> optional string() </Description>
  <Vehicle xd:script="occurs 0..; ref Vehicle"/>
</Acccident>

<Vehicle  Brand ="required string(1,80)"
          RegNum="required string(1,12)">
  <Person xd:script="occurs 0..; ref Person"/>
</Vehicle>

<Person Role = "optional string(0,36)" >
  <xd:choice>
    <Individual xd:script="occurs 0..1; ref Individual"/>
    <Enterprise xd:script="occurs 0..1; ref Enterprise"/>
  </xd:choice >
</Person>

<Individual Name   ="required string(1,24)"
            Surname="required string(1,36)"
            PIN    ="optional num(9,10)"/>

<Enterprise Name    ="required string(1,50)"
            IdNumber="optional num(8,10)"/>

</xd:def>
```

Xdefinition here describes a more complicated situation, which also includes enterprises besides individuals.

Comparison of XSD Schema and Xdefinition

Let's compare both descriptions from the perspective of the criteria which we enumerated at the beginning.

XML Schemata

Characteristics:

- Creates its own mark up language for the definition of client mark up languages
- Described mark up language is hidden in the values of attributes and elements

Main Advantages:

- This is the existing **standard** with a wide range of available products for processing
- The actual logics of the description can be processed using common XML technologies (Xpath, XSLT etc.)

Disadvantages:

- **Difficult to read** , it becomes an exclusive domain for programmers (or analysts) in implementation
- It is not suitable for discussing with partners or for project management, in this phase it is necessary to replace with minimally demonstrative examples or another alternative description
- For documentation it is necessary to also choose illustrative examples or another alternative description and schemata to consider an annex

XSD Schema can be called, with a certain degree of exaggeration, Esperanto created for the purposes of describing other languages (English, Czech, German)

Xdefinition

Characteristics:

- It does not create in the true sense of the word its own mark up language for the definition of client mark up languages
- The actual description of the client object is performed by the language of this object
- The actual description is inserted into places of values, elements and attributes

Main Advantages:

- **Legibility** of the description enables its setting during the following activities
  - Discussions with partners
  - Analytical phase of the project
  - Implementation phase of the project
  - Is comprehensible for project management
  - Useable for documentation
- The description is **obligatory**, i.e. it is machine processable.

Disadvantages:

- This is not an accessible, standardised technology
- For machine processing, available XML technology is not enough, it is necessary to develop propriatory tools

Xdefinition is an attempt to agree in English, Czech or German about how to write correctly in English, Czech or German

The name of the article hints at our answer to the question as to whether to go down the path of the universal language or to try to train descriptions as extensions in existing language.

Example: Description of Meta-Language Using Xdefinition

The poor legibility of the XML Schema is often compensated for by using alternative description. NG Relax introduces double syntax. The description of BPEL language is delivered using a meta-language, obligatory syntax using the XSD Schema forms an annex of the standard.

For illustration let's mention the ability of using the Xdefinition description technology as an alternative meta-language (Web Services Business Process Execution) Version 2.0.

Processes in BPEL contain activities. One of the activities is <receive>, which describes the behaviour of the case of the procedure when receiving news. Description of the activity <receive> in the informal syntax used is as follows (Chapter 5.2 OASIS wsbpel-v2.0-OS 11 April 2007):

```
<receive
   partnerLink ="NCName"
   portType="QName"?
   operation="NCName"
   variable="BPELVariableName"?
   createInstance="yes|no"?
   messageExchange="NCName"?
   standard-attributes>
   standard-elements
   <correlations>?
      <correlation set="NCName" initiate="yes|join|no"? />+
   </correlations>
   <fromParts>?
      <fromPart part="NCName" toVariable="BPELVariableName" />+
   </fromParts>
</receive>
```

The definition above is not an XML document and thus it cannot be processed like an XML document. In machine processing this definition is rewritten in the XML Schema (XSD). However, the XML Schema is not appropriate for explaining what the <receive> activity does.

A description of the same activity in the Xdefinition form:

```
<receive
  partnerLink     ="required NCName()"
  portType        ="optional QName()"
  operation       ="required NCName()"
  variable        ="optional BPELVariableName()"
  createInstance  ="optional list('yes','no')"
  messageExchange ="optional NCName()"
  xd:scrip="ref standardAttributes">
  <xd:any          xd:script="ref standardElements"/>
  <correlations    xd:script="occurs 0..1">
    <correlation  xd:script="occurs 0.."
      set         ="required NCName()"
      initiate    ="optional list('yes','join','no')" />
  </correlations>
  <fromParts       xd:script="occurs 0..1">
    <fromPart      xd:script="occurs 1.."
      part        ="required NCName()"
      toVariable ="required BPELVariableName()" />
  </fromParts>
</receive>
```

Example: Event Model and Process Processing

On the analytical level it is often necessary to specify some processes connected with data validation. An example is processing references to the code-books.

Part of the Xdefinition formal specifications is the validation process event model. An example is the **events of OnTrue** or **OnFalse** suspended on a type control element or attribute. The OnTrue event is called if validation goes well, the OnFalse event if it does not go well. This model then makes it possible to define the process processing of validated data in relation to these events.

The following example demonstrates the validation of input data, where the RoleCode attribute is at the input. This is the code for the role of Persons in road operation (instead of there being text, as in the introductory example). This code must be verified with respect to the code-book and to replace the id in relation to this code-book.

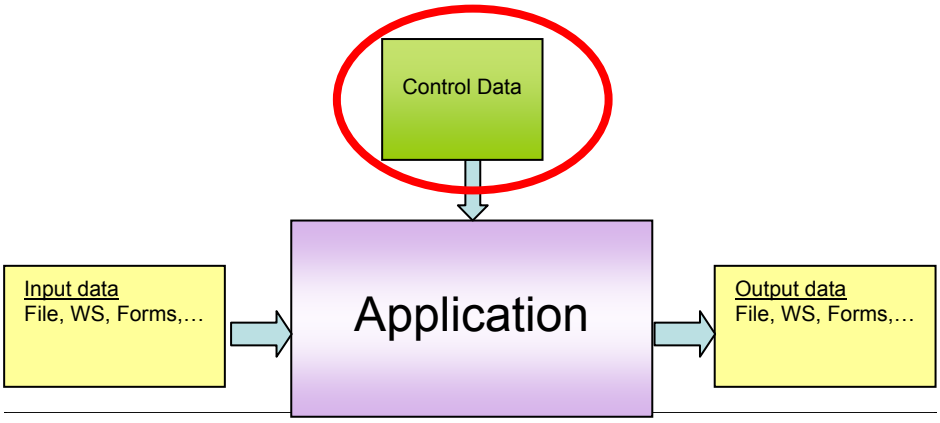
- **CodeBook ('CC_Role')** specifies code-book to which the code points
- **OnTrue ReplaceWithDBId();** calls the procedure for replacing the code with reference to the database
- **OnFalse error();"** determines a flawed reaction, if the code is not found in the database

```
<xd:def xmlns:xd="http://www.syntea.cz/xdef/2.0"
        root ="Accident"
        name ="TraficAccident">

  <Acccident DateTime="required datetime('d.M.y H:m')"
             RefNum  ="optional string(1,26)">
    <Description> optional string() </Description>
    <Vehicle xd:script="occurs 0..; ref Vehicle"/>
  </Accident>

  <Vehicle Brand ="required string(1,80)"
           RegNum="required string(1,12)">
      <Person xd:script="occurs 0..; ref Person"/>
  </Vehicle>

  <Person RoleCode ="required CodeBook('CC_Role');
                     OnTrue ReplaceWithDBId();
                     OnFalse error();" >
    <xd:choice>
      <Individual xd:script="occurs 0..1; ref Individual"/>
      <Enterprise xd:script="occurs 0..1; ref Enterprise"/>
    </xd:choice>
  </Person>

  <Individual Name   ="required string(1,24)"
              Surname="required string(1,36)"
              PIN="optional num(9,10)"/>

  <Enterprise Name    ="required string(1,50)"
              IdNumber="optional num(8,10)"/>

</xd:def>
```

Flexibility and Robustness of Application Managed by Monitoring DataL

Let's return to the introductory consideration of IS in an ever-changing world, where the flexibility and openness of applications play a main role. These characteristics determine the application's adaptability and thus its viability.

The primitive flexible application model can be understood thus:



In the schematic we can see the application first processes control data (parameters, customisation, meta-data, configuration etc.) and thereby set its behaviour for processing input and output data. Control data today mostly has an XML format. The wealth of control data determines the application's level of flexibility.

The actual processing of control data works according to the same model. Control data is the input, which then again has its own definition (i.e. its own control data).

BPEL machine is an example of this kind of application. Control data is BPEL definitions, inputs and outputs are calling and called web services. The BPEL definition has its own control data – such as BPEL Schema, describing the syntax of the BPEL language.

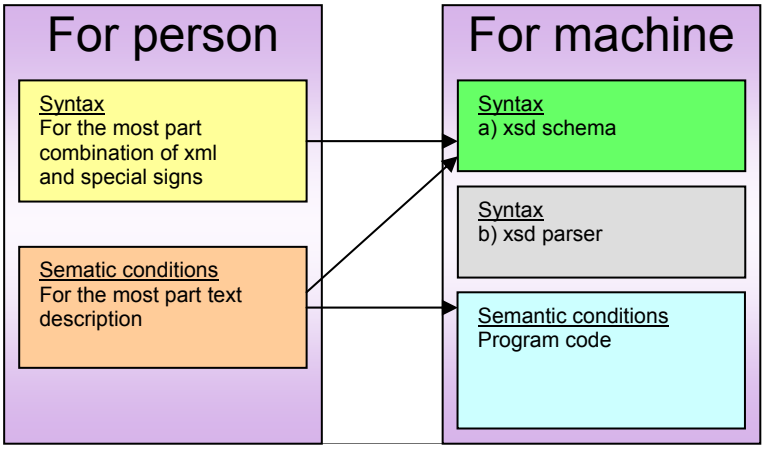
Let's compare the following approaches to a solution:

XML Schema

The description of the syntax of control data is intended for people and thus must be easily legible. Therefore specific notation must be used for this description (i.e. for the given XML language). Limiting semantic conditions are described by accompanying text.

So that it is possible to implement processing of control data, it is necessary to rewrite the specific notation of the specific XML language to another, such as XSD schema. Here it is possible to add certain schematic conditions. Such XSD schema define control data. Implementation of processing original control data means "programming" its formerly mentioned transformation.

The **consequence** is that there exist **four sources** for processing and maintenance of control data, which must be kept consistent:
1. definition in specific notation,
2. description of semantics and limiting conditions,
3. obligatory XML Schema (XSD)
4. code realising the conditions and transformation.



Xdefinition

The Xdefinition technology attempts to achieve greater source integrity for processing and maintenance. One Xdefinition is used for both human and machine definition of the language. Xdefinition technology enables the inclusion of multiple semantic conditions directly in to Xdefinition and thereby minimizing the code realising semantics.

As a result this there exist only three sources, which it is necessary to keep consistent:
1. obligatory Xdefinition (the same for people and machines)
2. description of semantics,
3. reduced code realizing conditions and transformation, which it was not possible to include in the Xdefinition.