The ITB Journal

Volume 3 | Issue 2 Article 8

2002

An XML Framework for the Structured Data Exchange Between Medical Devices

Ulrich Neuhaus

Paul Walsh

An XML Framework for the Structured Data Exchange Between Medical Devices

Ulrich Neuhaus [+], Paul Walsh [#]

Cork Institute of Technology, Rossa Avenue, Bishopstown, Cork, Ireland

[+] ulrich.neuhaus@e-healthcare.de, [#] pwalsh@cit.ie

Klaus W. Wente

Fachhochschule Darmstadt, Schoefferstr. 6. Darmstadt, Germany

k.wente@fbi.fh-darmstadt.de

Abstract

This paper describes an XML framework for the exchange of data between medical devices and software systems. XML technologies offer an extensible data format, which is easy to both create and process and simplifies the exchange of data. Different XML standards are defined within the framework in order to provide a practical solution for both device manufacturers and software developers. The structure and benefits of XML documents are discussed and an XML framework for ophthalmology is described.

Introduction

This paper describes an XML framework for the data exchange between medical devices in the area of ophthalmology. In ophthalmology, an examination contains measurements and maybe images of a patient's eyes. Even for one examination, different devices can be used and the collected information about the patient's eyes has then to be interpreted by a physician. Most of the measurements are done by devices that are connected to a software system. The physician should then be able to get all required information from an electronic patient record system, and use this information to treat the patient accordingly.

In the area of data exchange between medical software systems a wide variety of different standards for different purposes are available, e.g. Health Level 7 [1] for patient information, billing and accounting or DICOM [2] for digital imaging and structured reports. However the communication between medical devices and software systems is an area where virtually every vendor defines and uses their own specific data formats. For example, examination data is stored in a fixed length string, where all information has a known length and position in the string. Another variant on this scheme is a string separated by a control character such as a comma or semicolon, where the length of values is variable but the order is always fixed.

Changes and extensions to both formats require a customisation of the application interface and occasionally the device interface. Because of these reasons the existing software systems implement various interfaces to all or most available devices to connect them to the application. Those implementations work properly, but with every new or changed device, the existing program code gets larger and the maintenance for those interfaces gets expensive. At this point it would be helpful to have one data format which has to be processed.

In this paper, different XML technologies are used for describing the structure of data that is communicated between medical devices and software applications. The use of XML provides both device manufacturers and software developers with a powerful solution, which could solve the existing problems and make data exchange easier and cheaper. Moreover, XML is a standardized format and there are a lot of libraries for creating, accessing and parsing XML documents very quickly, e.g. MSXML Parser [3] or SAX [4]. For the software developer it is easier to process an XML document, because the processing of the received document is always the same. The developer only needs to know which data elements are relevant and can frequently reuse the same code for processing that data. The vendor gets a data format, which is easy to extend for future requirements. The use of a *standardized* format is also a useful selling point. By using a standard format, the idea of connecting a device easily to a software system (like plug and play) becomes possible.

As there are a wide range of medical devices available that cater for every medical field, the system described here is focused only on a subset of devices in the area of ophthalmology. Indeed a lot of standardization work has been done in the area of ophthalmology and the readiness of manufacturers for participating in such a framework is encouraging. If a usable framework is developed and verified, it could be adapted to other medical areas.

XML Technologies.

The XML language [5] makes it possible to structure and process almost any kind of information. All information is stored between markup tags or in attributes. An XML document contains markups and data. A markup is used for describing the document's layout and logical structure, while the data is the main information described in the document. An XML document is normally human readable.

An XML document is only valid if the document is well formed. This means, that all information is stored between markups and every opened tag is closed. This function is offered by an XML schema. An XML schema [6] is a document that formally describes an XML document and can be used to prove if the information inside markups is valid.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Autorefractor PatientGuid="471143" ExaminationDate="2002-08-01"
3   AFType="manual" DocumentID="898890">
4           <AFLeft>
5                   <Geometry>
6                           <Sphere>120</Sphere>
7                           <Cylinder>100</Cylinder>
8                           <Axis>90</Axis>
9                   </Geometry>
10          </AFLeft>
11          <AFRight>
12                  <Geometry>
13                          <Sphere>100</Sphere>
14                          <Cylinder>140</Cylinder>
15                          <Axis>70</Axis>
16                  </Geometry>
17          </AFRight>
18          <Remarks>some remarks</Remarks>
19  </Autorefractor>
```

Figure 1: XML document for an auto refractor

Figure 1 shows a valid XML document. The tag "Autorefractor"[1] is the root element and contains four attributes (PatientGuid, ExaminationDate, AFType and the DocumentID) and three child elements. These elements store the measurement values "AFLeft" and "AFRight" from the examination and some free text "Remarks". Every XML Document can be displayed as a tree-structure with one root and several child elements (see Figure 2). It is not possible for a document to have two root elements; if this happens the document is not valid.

Within the XML Schema, the structure of an XML document is described. For each element and attribute the name and the data type are defined. The schema also includes information about:

- optional and required attributes and elements;
- the order and the occurrence of elements;
- the range of values for attributes and elements.

[1] An Autorefractor is a device for measuring the refractive power of an eye.

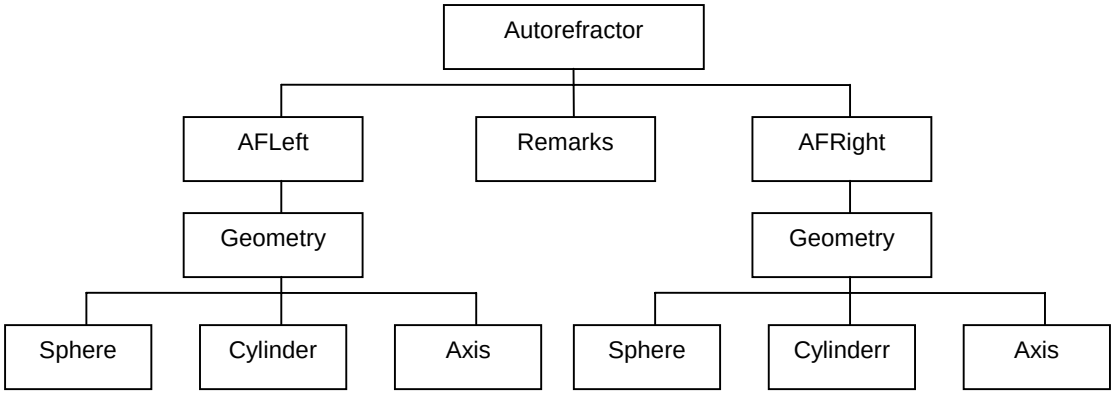

Figure 2: XML tree-structure

Schemas are used in two different ways. The creator of a XML document uses the schema to retrieve all information necessary for creating a valid document. The software developer or the application uses the schema to validate the document against it.

A language for accessing elements in an XML document is also specified. Simple access to any element in the document is offered by XPath [7]. With XPath a query language is available that is similar to the functionality of SQL for relational databases, but is much easier to use. Every element and attribute in the document is accessible by a path. In Figure 3, the first query gets the value of the element sphere; the second query retrieves the value from the attribute PatientGuid.

/Autorefractor/AFLeft/Geoemtry/Sphere

/Autorefractor/@PatientGuid

Figure 3: Simple XPath Queries

Schema Definition for Medical Devices

Initially all information must be defined using XML schemas. Four different types of XML schemas, shown in Figure 4, were developed.

Different approaches were used for defining these schemas. The examination schemas were defined by using the information about the data format from the specifications of the respective device, e.g. measurement and range of values. The examination schema for a perimeter[2] was also developed along with the device manufacturers Interzeag [8] and Oculus

[2] A perimeter is a device for measuring the visual field of an eye.

[9]. After collecting the required information, it was converted to a tree-structure to meet the XML requirements.

The examination document is defined by the examination schema. This document stores all data acquired during the examination. The data is divided into the results for the right and left eye. The schema includes three schemas for the patient, the device and the self-defined data types. These additional schemas are optional. When the examination document is created, it may also contain a sub document defined by the patient schema and one defined by the device schema.

In this scope, the examination document contains only the measurements from one device. If a physician is performing several examinations on a patient, the results from all used devices are stored in a new examination document.

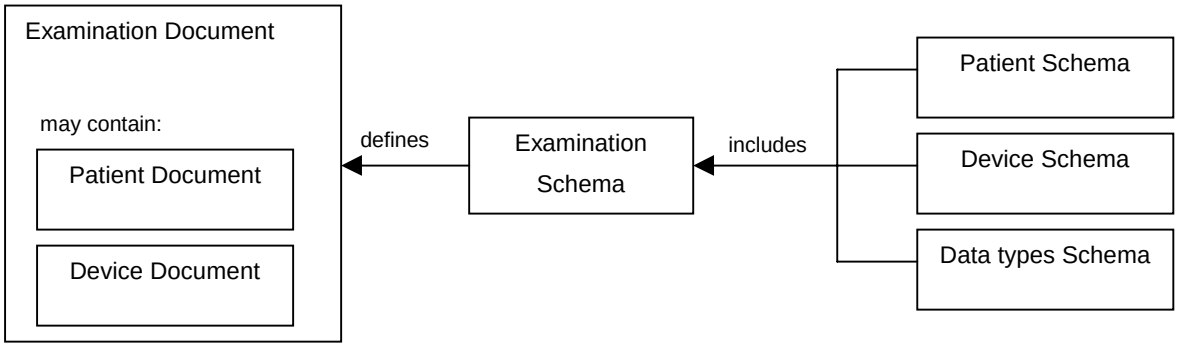

Figure 4: Schema types

- **Patient Schema**

  In the patient schema, personal and contact information about a person is stored, e.g. patient ID, name, and address. Most of the elements in this schema are optional, because not all information is needed for an examination. Only the patient ID is required for assigning the examination results to the correct patient in an application. The patient schema is referenced in every examination schema as optional data.

- **Device Schema**

  The device schema includes all relevant information about the respective device, e.g. device type, software and firmware revisions. The device schema is also linked to every examination schema as optional data.

- **Examination Schema**

  A schema for every device is developed. In this schema all possible values and elements that could be transferred from the device to the application are defined. Schemas are available for the most recent ophthalmologic devices, e.g. auto refractor. A special

section is defined in every schema for vendor specific extensions. This takes advantage of the extensibility of XML. All data stored in this section won't be validated, because the structure of those elements are not defined in the schema and so are unknown to it.

- **Data type Schema**

  An additional schema is developed for elements, which are used in several examination schemas, e.g. the type geometry is composed of three elements: axis, sphere and cylinder.

Figure 5 shows the XML Schema for an Autorefractor. In line 7 to 9 a prefix is defined for each of the three included schemas. The inclusion is done in line 10 to 12. Every element is composed of a name and a data type. It is possible to use simple data types (e.g. float or string), data types from other schemas (e.g. line 13, the definition of an element from the type "DeviceInformation") and to define complex data types composed of simple or other complex types (e.g. line 18 and 19, the "AutorefractorDataset" contains the elements "Geometry" and "VisualAcuity").

In line 35 the root element is defined and all the elements, which are possible as child elements, are listed. Also the attributes are defined and for the attribute "AFType", the potential values are defined (line 54 to 56).

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <xsd:schema targetNamespace=
3   "http://schemas.ectalk.org/xeot/1.0/base/autorefractor"
4   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
5   xmlns="http://schemas.ectalk.org/xeot/1.0/base/autorefractor"
6   xmlns:Autorefractor="Autorefractor"
7   xmlns:ECTALK="http://schemas.ectalk.org/xeot/1.0/base/ectalkdatatypes"
8   xmlns:DI="http://schemas.ectalk.org/xeot/1.0/base/deviceinformation"
9   xmlns:PATIENT="http://schemas.ectalk.org/xeot/1.0/base/patient">
10          <xsd:import namespace="http://schemas.ectalk.org/xeot/1.0/base/deviceinformation"/>
11          <xsd:import namespace="http://schemas.ectalk.org/xeot/1.0/base/ectalkdatatypes"/>
12          <xsd:import namespace="http://schemas.ectalk.org/xeot/1.0/base/patient"/>
13          <xsd:element name="DeviceInformation" type="DI:DeviceInformation"/>
14          <xsd:element name="Patient" type="PATIENT:PatientInformation"/>
15          <xsd:element name="VisualAcuityBinocular" type="xsd:string"/>
16          <xsd:element name="Remarks" type="xsd:string"/>
17          <xsd:element name="PD" type="xsd:float"/>
18          <xsd:element name="AFRight" type="AutoRefractorDataset"/>
19          <xsd:element name="AFLeft" type="AutoRefractorDataset"/>
20          <xsd:complexType name="ExtType">
21                  <xsd:sequence>
22                  <xsd:any namespace="##any"
23                          processContents="lax" minOccurs="0" maxOccurs="0"/>
24                  </xsd:sequence>
25          </xsd:complexType>
26          <xsd:element name="Ext" type="ExtType"/>
27          <xsd:complexType name="AutoRefractorDataset">
28                  <xsd:sequence>
29                          <xsd:element ref="Geometry" minOccurs="0"/>
```

```
30                          <xsd:element ref="VisualAcuity" minOccurs="0"/>
31                  </xsd:sequence>
32          </xsd:complexType>
33          <xsd:element name="Geometry" type="ECTALK:Geometry"/>
34          <xsd:element name="VisualAcuity" type="ECTALK:VisualAcuityComplex"/>
35          <xsd:element name="Autorefractor">
36                  <xsd:complexType>
37                          <xsd:sequence>
38                                  <xsd:element ref="AFLeft" minOccurs="0"/>
39                                  <xsd:element ref="AFRight" minOccurs="0"/>
40                                  <xsd:element ref="Remarks"/>
41                                  <xsd:element ref="PD" minOccurs="0"/>
42                                  <xsd:element ref="VisualAcuityBinocular" minOccurs="0"/>
43                                  <xsd:element ref="DeviceInformation" minOccurs="0"/>
44                                  <xsd:element ref="PatientInformation" minOccurs="0"/>
45                                  <xsd:element ref="Ext"/>
46                          </xsd:sequence>
47                          <xsd:attribute
48                                  name="PatientGUID" type="xsd:anyURI" use="required"/>
49                          <xsd:attribute
50                                  name="ExaminationDate" type="xsd:date" use="required"/>
51                          <xsd:attribute name="AFType" use="required">
52                                  <xsd:simpleType>
53                                          <xsd:restriction base="xsd:NMTOKEN">
54                                                  <xsd:enumeration value="manual"/>
55                                                  <xsd:enumeration value="auto"/>
56                                                  <xsd:enumeration value="skiascopy"/>
57                                          </xsd:restriction>
58                                  </xsd:simpleType>
59                          </xsd:attribute>
60                          <xsd:attribute
61                                  name="DocumentID" type="xsd:anyURI" use="required"/>
62                  </xsd:complexType>
63          </xsd:element>
64  </xsd:schema>
```

Figure 5: XML Schema for an auto refractor

Data Exchange

The data exchange between a device and an application is shown in Figure 6. In this communication, the respective XML schemas are stored in the device and also in the application.

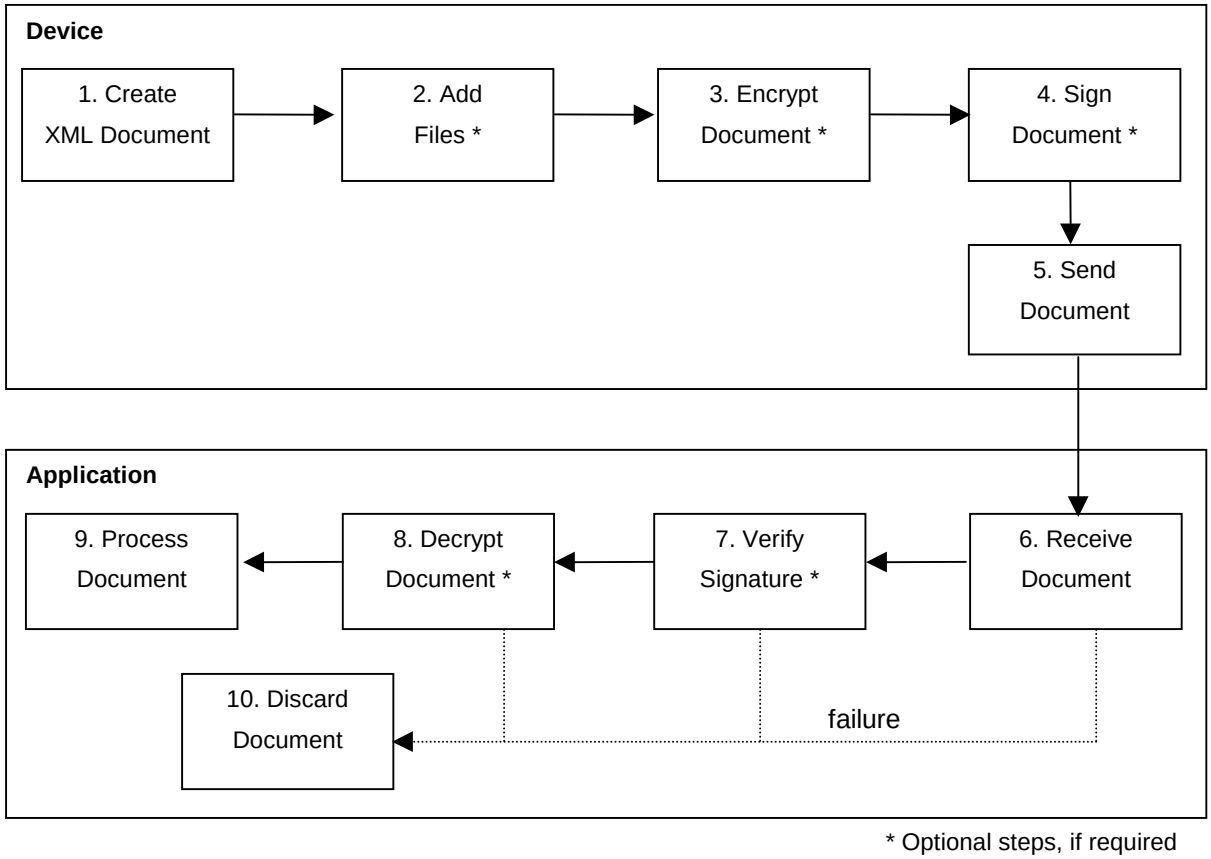

Figure 6: Data exchange

1. Create XML – For every examination a new document is created. At this point, all results are collected from the device application and the document is created, according to the respective schema.

2. Add Files – This step is optional and only required, if binary files (e.g. pictures) should be transferred within the document. The files will be added as string values.

3. Encrypt Document – Because of the sensitivity of all medical information these aspects can become very important, especially when the information is transferred over the Internet or other public networks. Parts or the whole document can be encrypted.

4. Sign Document – In this step the document could be digitally signed to proof authenticity. Steps 3 and 4 are optional. Both steps require a special implementation on the device to allow these security extensions.

5. Send Document – When the document is finally created, it can be sent to the receiving application. Currently the framework does not define how the XML document is to be transferred, e.g. using http. The decision of how this is done is left to the device manufacturer.

6. Receive Document – The document is received by the application and prepared for further processing.

7. Verify Signature – When the document is digitally signed, the signature is verified and if successful the processing is continued.

8. Decrypt Document – If the document is encrypted, the application needs to decrypt it. Steps 7 and 8 also require a special implementation in the application for verifying and decrypting the document. These steps are also optional.

9. *Process Document* – Once all preceding steps are successfully completed, the main processing of the document can begin. The document is verified against the schema, to ensure, that all information in the document is correct and can be processed. It is up to the application, if the information is displayed or stored in a database.

10. Discard Document – If an error occurs in one of the steps, the document is discarded and the examination has to be sent again.

Conclusions

In this paper we have described the structure of an XML framework for data exchange between ophthalmologic medical devices and software systems. The use of XML technologies for structuring has the following advantages:

- it facilitates the use of a standardized data format;
- it is easy to create and process XML data with the existing libraries and additional standards;
- it provides an open data format, which is extensible.

At this stage of the research we have collect all required information about the data output of the devices and defined with this information a base set of XML schemas.

At the moment, one manufacturer (Laser Diagnostic Technologies [10]) is using an earlier version of the XML Schema defined in the framework described in this paper and several manufacturers are interested. The framework contains 15 manufacturer-independent schemas for different medical devices and one vendor specific schema.

In the next phase of research more steps are required. For both, device manufacturer and software developer, a reference implementation is needed. This implementation should show how to create and process XML documents and how to use the XML schemas for validation. Furthermore, the implied security aspects in the data exchange (encrypting, decrypting and signing XML documents) and the use of external resources, which may only be stored in the examination document as an URL, have to be worked out in more detail. For this step, the existing guidelines from the W3C about XML Encryption [11], XML Signature [12] and

XML Linking Language [13] must be incorporated into the framework. Finally, the completed framework has to be verified by manufacturers to ensure a usable solution.

References

[1] Digital Imaging and Communication (DICOM)
http://medical.nema.org/
[2] Health Level 7 (HL-7)
http://www.hl7.org
[3] Microsoft XML Parser 4.0 (MSXML)
http://msdn.microsoft.com/library/en-us/xmlsdk/htm/sdk_intro_6g53.asp
[4] Simple API for XML (SAX)
http://www.saxproject.org/
[51 Extensible Markup Language (XML) 1.0 (Second Edition)
http://www.w3.org/TR/2000/REC-xml-20001006
[6] XML Schema, Part 0, 1 & 2
http://www.w3.org/XML/Schema
[7] XML Path Language (XPath)
http://www.w3.org/TR/1999/REC-xpath-19991116
[8] Interzeag International AG
http://www.octopus.ch
[9] Oculus Optikgeräte GmbH
http://www.oculus.de
[10] Laser Diagnostic Technologies (LDT)
http://www.laserdiagnostic.com
[11] XML Encryption
http://www.w3c.org/Encryption/2001/
[12] XML Signature
http://www.w3c.org/Signature/
[13] XML Linking Language
http://www.w3.org/XML/Linking