

# Using XML for flexible data entry in healthcare

## example use for pathology

### Ralf Schweiger

Institute for Medical Informatics, Giessen, Germany  
ralf.schweiger@informatik.med.uni-giessen.de

### Ali Tafazzoli

Institute for Medical Informatics, Giessen, Germany  
ali.g.tafazzoli@informatik.med.uni-giessen.de

### Joachim Dudeck

Institute for Medical Informatics, Giessen, Germany  
joachim.w.dudeck@informatik.med.uni-giessen.de  
<http://www.uni-giessen.de>

#### **Abstract:**

*This paper describes a pragmatic, generic and flexible approach for the management of XML structured data at the example of pathology reports. The flexibility of this approach is based on a template concept. The template describes the documents of a given (clinical) domain in terms of structure and user interface requirements. The template enables a so called document manager to provide a corresponding user front and storage back end (XML document base). With the template approach, the application engineer merely needs to create and change templates in order to meet the latest documentation and application needs of a clinical domain. This presentation outlines our approach to more flexible clinical data structures.*

## Introduction

In a clinical environment we often have to face the problem of unstructured textual resources like pathology reports, treatment guidelines or scientific publications in a university hospital. The lack of structure limits the automatic identification and extraction of the information contained in these resources, i.e. many potential application services such as context sensitive searching, dictionaries, statistics and knowledge based functions are difficult to provide. As a consequence, the information and knowledge (relationship between information) contained in clinical documents remains unused to a large extent.

Since Feb 1998, the XML (Extensible Markup Language) is a standardized format for the structuring of Web resources . With XML, single elements are marked inside textual documents. This markup principle might be used as well for clinical resources. In a pathology report e.g. we might want to mark the patient data and the findings (e.g. by <finding>malign tumor</finding>) which allows an application service to automatically identify and relate these elements inside a pool of pathology reports. The markup consequently solves the problem of identifying the information inside textual resources and enables the development of application services on top of it. In addition, XML does not only enable but also accelerate the application development. The XML technology provides the software engineer with a variety of software components for the parsing and transformation of XML structured data. The transformation facility e.g. can be used to quickly translate XML structured data into standard messages like HL7 or to provide different views on a document (text without markup, marked elements only). The ability to provide new functionality for XML structured data with little investments is an important feature of the XML technology. In that sense, XML follows the trend of emerging component strategies.

For the acquisition and maintenance of XML structured clinical data we need user interfaces which make the XML format transparent to the clinical user (pathologist, pharmacist, surgeon, nurse). XML is still regarded as a message format , rather than a storage format . As a consequence, the development of user interfaces for the management of XML structured data has not yet been of prior interest inside the XML technology. For this reason, we have designed and implemented a pragmatic, generic and flexible approach to provide simple user interfaces for the management (creation, search, update etc.) of XML structured data. This presentation wants to communicate the central concepts of our approach.

## Method

The central aspect of the user interface is the transparency of the XML markup for the clinical user. The user is interested in content issues only, not in formatting issues. The XML transparency is achieved with with a software component which will be referred to as document manager. The document manager provides the user with an HTML form that corresponds to the XML document (pathology report) and which enables the pathologist to create, specifically select and update XML structured data. For the construction of new reports, the document manager needs to know both, the document structure (nested, repeatable, optional elements) and the form structure (user prompts, different form objects, connections to remote data servers). For pragmatic reasons, we merged both kind of construction information into a simple XML document which we call a template. Figure 1 illustrates the overall architecture of our approach. The document manager uses the XML template for both, the automatic generation of an HTML form (user interface) and the generation of an XML document (storage format of the data entered by the user). As a consequence, we only need to add new templates or change existing templates and the document manager will automatically provide/adapt the corresponding front & back end. The application engineer can focus on XML templates, i.e. on structural and modelling issues rather than on logical and programming issues. With such an approach changing user requirements can be quickly satisfied and the clinical user no longer has to put up with inflexible data structures.

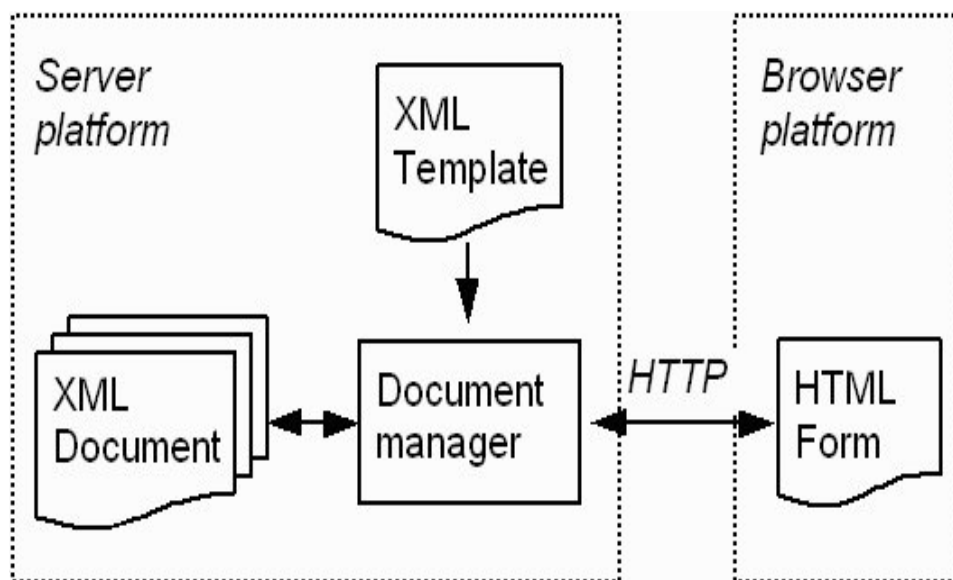


Figure 1. Overall software architecture

*Architecture for the management of XML documents based on a Web platform with a server back end and a browser front end. The template describes and relates both, the front end (form) and the back end (document). The application developer merely provides a template and the document manager will automatically generate a corresponding front & back end.*

Following we will give a respective example for the XML template, the HTML form and the resulting XML document.

## XML template

Figure 2 presents a simplified XML template for pathology reports. The template contains XML elements such as <Patient>, <Specimen>, <MacroscopicFinding> and <Morphology>, i.e. XML elements in the template represent domain specific markup. Relationships between the elements are established by element nesting. The construct <Morphology><Code/></Morphology> e.g. communicates the code of the morphology. Empty elements such as <Code/> represent a data entry point in the corresponding form.

To each domain specific XML element in the template we associate a number of domain independent XML attributes such as **occur**, **type** and **prompt**. The default occurrence is exactly once and might be changed to e.g. occur="omit" which declares an optional element. The type attribute might be applied to empty template elements, i.e. data entry points in the form and determines the form object to be used for the data entry. Possible form objects are selections (<Sex>), multi line text (<MacroscopicFinding>) and single line text (default). A selection is a comma separated list of the possible values, with a prompt and an optional code for each value (prompt/code). The user prompt of an XML element defaults to the element name and might be changed to a more verbose text. The number of attributes associated with a template element is open and can be extended and adapted to the current needs. We have implemented for example a **get** attribute whose value is a URL (Uniform Resource Locator) that identifies a remote data server. Such a mechanism allows the document manager to retrieve and integrate existing data from remote locations using different transfer protocols (HTTP, IIOP etc.).

```
- <PathologyReport prompt="Pathologie-Bericht">
  <RequestDate prompt="Eingangsdatum" />
  - <Patient>
    <FamilyName prompt="Nachname" />
    <GivenName prompt="Vorname" occur="repeat" />
    <DateOfBirth prompt="Geburtsdatum" />
    <Sex prompt="Geschlecht" type="mann/m, frau/f" />
  </Patient>
  <Specimen prompt="Probe" occur="repeat" />
  <MacroscopicFinding prompt="makroskopischer Befund" type="narrate" />
  <MicroscopicFinding occur="omit" prompt="mikroskopischer Befund" type="narrate" />
  <Assessment prompt="Beurteilung" type="narrate" />
  - <Morphology occur="repeat omit">
    <Code />
    <Description prompt="Beschreibung" occur="omit" />
  </Morphology>
</PathologyReport>
```

Figure 2. XML template

Simple XML template for pathology reports.

## HTML form

The XML template is used by the document manager to generate a corresponding HTML form. The user may repeat and omit elements, composed or not, due to their occurrence defined in the template. The <Morphology> group e.g. can be repeated as often as needed. The empty XML elements in the template

such as <Code/> correspond to data entry points in the HTML form. Since no template attribute is given for the <Code> element, the default prompt (element name) and the default form object (single line text) apply.

In the form of Figure 3 the user has selected the second <Morphology> element (radio button) and the corresponding operators appear in the upper frame. The clinical user can close complex elements (composed elements, multi line text) with a click on the respective radio button and the text contained in the closed element appears on the right hand side. Moreover, the document server joins elements without data into a single row. Such transparency functions make the entry form more compact and are useful for larger documents. Due to the occurrence defined in the template the user may now create a new morphology element (newEl) or delete the selected element (delEl). The putDoc (put Document) operator is used to store the current document in the document base of the given domain, e.g. the pathology domain. The getDoc function allows the user to retrieve a list of those documents that match with the entered data and to select a document from the retrieved list for update purposes. The getDoc function consequently renders a context sensitive search in the document base of the given domain.

An example shall illustrate the search functionality. The pathologist might be interested in those pathology report(s) which contain the term “Basaliom” in their assessment part (prompt Beurteilung). For this purpose he starts with an empty HTML form and simply enters the search term into the specific context. By pressing the getDoc button he will receive a list with all matching documents, i.e. pathology reports. He can then select a document from the list and update the corresponding data in an HTML form. The user can also enter several search terms in the same or in different contexts to make the search even more specific. Context sensitive search techniques as possible with XML are an advantage over brute force algorithms which often provide irrelevant search results.

Operator:

Pathologie-Bericht

Eingangsdatum

Patient

Probe

Probe

makroskopischer Befund

mikroskopischer Befund

Beurteilung

Morphologie

Morphologie

Code

Beschreibung

Morphologie

Figure 3. HTML form

HTML form generated from the XML template in Figure 2. The HTML form does not contain XML markup, only data that are related to the clinical domain.

## XML document

The data in the HTML form can now be stored as XML document (Figure 4). The translation between the HTML form and the XML document is done by the document manager. According to a recommendation of , the document manager generates an XML element representation of the content. We can, however, generate

any other representation of the data. We simply have to provide corresponding XSLT stylesheets that describe the transformation between the default representation and the target representation. The document manager includes an XSLT engine which performs the transformation specified in the XSLT stylesheet.

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
- <PathologyReport domain="pathology" file="pathology3815">
  <RequestDate>19990323</RequestDate>
  - <Patient>
    <FamilyName>Müller</FamilyName>
    <GivenName>Anne</GivenName>
    <DateOfBirth>19670721</DateOfBirth>
    <Sex>f</Sex>
  </Patient>
  <Specimen>1: PE Stirn links</Specimen>
  <Specimen>2: PE Nasenflügel rechts</Specimen>
  <MacroscopicFinding>1: In der mediokaudalen Präparatehälfte zeigt sich die Haut sehr unscharf begrenzt auf einer Fläche von mindestens 0,5 cm Durchm. livide verfärbt und etwas aufgeworfen. 2: Ein ovaläres von lateral nach medial 2,2 cm durchm. und von kranial nach kaudal 1,6 cm durchm. Hautexzidat.</MacroscopicFinding>
  <MicroscopicFinding>Auf fortlaufenden Schnittstufen von 1.6. bis 1.9. erkennt man multizentrisch lokalisierte mikrofokale, im weiteren Verlauf auch netzförmig konfluierende, von der Epidermis ausgehende basaloide Tumorzellkomplexe, die eine Palisadenstellung der peripheren Zellreihen aufweisen und von spaltförmigen Hohlräumen umgeben sind.</MicroscopicFinding>
  <Assessment>Es handelt sich um ein multizentrisch wachsendes oberflächl. Basaliom im Bereich der linken Stirn (1), sowie um ein solide zystisches und skerodermiformes Basaliom im Bereich des rechten Nasenflügels (2). Ferner handelt es sich um eine aktinische Keratose mit bis zu leichten Epitheldysplasien in beiden Entnahmelokalisationen.</Assessment>
  - <Morphology>
    <Code>ICD-O M-8091/3 (1)</Code>
    <Description />

```

Figure 4. XML document

XML document that corresponds to the HTML form in Figure 3. This is the storage format for the data that have been entered by the clinical user into the HTML form. The XML format enables an application to reliably identify (search, extract etc.) marked information elements.

## Discussion of the approach

Our method was introduced as a pragmatic, generic and flexible approach for the management of XML structured data. In this section, we want to resume these terms as different viewpoints of our approach and detail their meaning in the style of a discussion.

### Pragmatic viewpoint

In our approach, we regard XML as a storage format rather than a message format for clinical data. In that sense, the document approach “competes” with a clinical database approach. However, we regard the document approach as a complementary rather than a substitutional approach to clinical databases. How can the different approaches complement each other? Our approach might be regarded as a migration path from unstructured textual clinical documents towards structured clinical databases, i.e. as a transitional solution rather than a final solution to the problem of structuring clinical data. XML turned out a proper means to accomplish such a migration, proper in a healthcare specific sense and proper in a technological sense.

Conceptually, XML regards the document as the central data unit. In a clinical environment, documents are used as communication units (doctor’s letter, pathology report etc.), i.e. we often have to start from textual documents in the healthcare domain. The document approach consequently reflects the healthcare situation better than a database approach which often requires clinical documents to be de- and recomposed. On the other hand, databases are technologically matured, perform better in the management of huge amounts of data and increasingly provide interfaces for XML structured data. In that sense, XML documents might be seen as an intermediate step from free text documents towards clinical databases. Such an intermediate step will facilitate the transition towards more structured clinical data.

Technologically, XML provides the engineer with a variety of standard software which supports the processing (parsing, creation, transforming etc.) of XML documents. Public software with standardized interfaces is also a pragmatic reason for using XML as a storage format. With XML it is simple to add structure (markup) to existing textual resources and to refine the structure in a gradual fashion, without heavy investments from the very start.

Another pragmatic viewpoint of our approach is the XML template. There are standard languages for both, the description of document types (DTD, XML ) and the description of user interfaces (HTML, XUL, ). However, there is no easy way to relate them with each other. User interface languages provide the developer with powerful constructs for the front end development, i.e. for the acquisition and the presentation of data. However, the back end logic for storing and retrieving the data is left to the engineer and requires still a lot of implementation effort. The standard representations DTD and XML Schema, on the other hand, describe a document type rather than a user interface and are designed for validation purposes. As a consequence, it is difficult to accommodate elements of the user interface such as prompts, form objects and connections to remote data servers in these standard document descriptions.

Our XML template has been designed with the goal not only to accommodate both kind of information (structure of the document & structure of the form), but also to relate the information with a minimum set of XML markup. We have introduced only a few XML attributes which might be even omitted if the default values apply. The result is a compact and straight forward XML template that mainly contains markup that is related to the clinical domain. It is not to mention that the XML Schema approach is superior to the XML template approach as long as the description of document types is concerned. The same is true for HTML which is more flexible in the description of user forms than it is the XML template. In that sense, the XML template is a pragmatic solution for the ease of implementation.

We are currently about to implement a solution that works with a standard XML Schema instead of a nonstandard XML template. However, we still have to investigate the possibilities of how to accommodate the user interface requirements within an XML Schema. One possible approach is to merge the XML Schema namespace with the namespace of a user interface markup language like HTML.

### **Generic viewpoint**

We have introduced the XML template approach at the example of pathology reports. In fact, the document manager in Figure 1 can manage any number of templates and related documents. We only need to design templates for laboratory reports, nursing documentation, questionnaires for quality assurance or other clinical domains and purposes. The document manager can handle a repository of clinical documents and will provide basic application services for the maintenance and context sensitive selection of the XML structured clinical data. On top of such a repository we might further implement integration services which affect documents from different domains. One example for such an integration service is the electronic patient record which selects the documents of a given patient from several domains (lab, pathology).

### **Flexibility viewpoint**

Flexibility in this context means the ease to meet the latest documentation needs and application requirements. Flexibility plays an important role in a domain that is as dynamic as the healthcare domain. The pathologist for example might want to introduce new elements into his documentation model. New application services might also require a refinement of the documentation model. Applications with a fixed data structure can hardly meet this challenge and often lead to unstructured documentation, low application acceptance or high cost for application maintenance. Our solution to this problem is an approach that allows

the application engineer to quickly adapt the application (user front & storage back end) to the latest needs. The application developer simply has to provide and change XML templates and the document manager will do the rest. Ease of configuration is the key issue to flexible data structures and more structured clinical data.

## Conclusion

XML turned out a means to quickly introduce structure into clinical documents and to refine the structure step by step in order to meet the latest documentation and application requirements. As soon as we use XML as a storage format we also have to think about user interfaces that make the storage format transparent for the clinical user. We have developed a flexible approach that is based on a template concept and that allows the application engineer to quickly adapt both the data acquisition and the storage structure to the changing clinical needs. Such an approach can be completely based on standards and allows an evolution from unstructured over semi-structured towards completely structured clinical data. This seems to be a promising approach for many clinical domains where only little structure has been captured so far.

## Bibliography

- [ARC] Wigefeldt T, Larnholt S, Peterson H: Development of a standardized format for archiving and exchange of electronic patient records in Sweden. Proceedings of Medical Informatics Europe 1997, 252-256.
- [DOM] World Wide Web Consortium: Document Object Model (DOM) Level 1 Specification Version 1.0. W3C Recommendation 1 October 1998.
- [EDI] CEN/TC 251, Task Force XML.
- [EPR] Schweiger R, Buerkle T, Ruan W, Dudeck J: XML: Evolution towards a structured electronic patient record. Electronic Patient Records in Medical Practice, Proceedings of IMIA Working Group 17, Rotterdam 8-10 October 1998.
- [MSG] Morgenthal JP: Enterprise Messaging with XML. Component Strategies, May 1999, 54-57.
- [Schema] World Wide Web Consortium: XML Schema Part 1: Structures. W3C Working Draft 25 February 2000.
- [TAM] Software AG: Transaction Architecture for the Management of INternet Objects (TAMINO).
- [UIML] User Interface Markup Language (UIML). <http://www.uiml.org> .
- [XML] World Wide Web Consortium: Extensible Markup Language (XML) 1.0. W3C Recommendation 10 February 1998.
- [XSLT] World Wide Web Consortium: XSL Transformations (XSLT) Version 1.0. W3C Recommendation 16 November 1999.

## Authors

### Ralf Schweiger

Information Scientist in Medicine  
Institute for Medical Informatics  
Postal Address:  
Heinrich-Buff-Ring 44  
35392 Giessen  
Germany  
Telephon: +49(641)9941370  
Fax: +49(641)9941359  
E-mail: ralf.schweiger@informatik.med.uni-giessen.de

**Ralf Schweiger** - Ralf Schweiger joined the department for Medical Informatics at the university of Giessen in 1996. He is member of a group that explores and implements methods for the integration of clinical information systems. His main research interests are middleware architectures, telemedicine methods and the Internet technology with a focus on XML.

### Ali Tafazzoli

Information Scientist in Medicine  
Institute for Medical Informatics  
Postal Address:  
Heinrich-Buff-Ring 44  
35392 Giessen  
Germany  
Telephon: +49(641)9941382  
Fax: +49(641)9941359  
E-mail: ali.g.tafazzoli@informatik.med.uni-giessen.de

**Ali Tafazzoli** - Ali Tafazzoli joined the department for Medical Informatics at the university of Giessen in 1994 where he works in a working group which has a nationwide coordinating role for various issues related to the tasks of clinical tumor registries, such as data analysis, training of registrars and the definition of documentation standards. His main research interests are the integration of organspecific documentation, concepts and standards for data exchange and arden-syntax-based decision support.

### Joachim Dudeck

Information Scientist in Medicine  
Institute for Medical Informatics  
Postal Address:  
Heinrich-Buff-Ring 44  
35392 Giessen  
Germany  
Telephon: +49(641)9941350  
Fax: +49(641)9941359  
E-mail: joachim.w.dudeck@informatik.med.uni-giessen.de  
Web: www.uni-giessen.de

**Joachim Dudeck** - Prof. Dudeck is head of the department for Medical Informatics at the university hospital in Giessen. He has been involved in the development of medical data dictionaries, knowledge based hospital information and application systems. Since 1993 he is chair of the HL7 user group in Germany and became involved in XML applications in particular in healthcare and as interchange format in messaging. He was also chairman of the CEN TC 251 XML Task Force. The institute was partner of the ISIS XML/EDI Pilot Project.