Medical Infobahn for Europe
A. Hasman et al. (Eds.)
IOS Press, 2000

Clinical Trial Management and Remote Data Entry on the Internet based on XML Case Report Forms

Peter Wübbelt[1], Guillermo Fernandez[2], Joachim Heymer[3]

[1]Department of Biometry; Hannover Medical School, Carl Neuberg Str. 1, D-30625 Hannover, Germany
[2]Superior Technical School of Telecommunication Engineering, Universidad Politécnica Valencia, Cami de Vera, 46022 Valencia; Spain
[3]Avatar Software, D-30625 Hannover, Ebellstr. 28, Germany

The Internet and the World Wide Web have recently been introduced into the management of some aspects of clinical trials such as remote randomisation and data entry and the distribution of information on trial progress. A few larger trials use electronic Case Report Forms (eCRFs) built with HTML. This article describes a more flexible approach using the Extensible Markup Language XML.

Key Words: Remote Data Entry RDE; Electronic Data Capture; EDC; Clinical trials; Internet; Extensible Markup Language; XML; Extensible Stylesheet Language; XSL; Servlets; Java

1. Introduction

The clinical development of a drug is a complex and costly process which must be conducted under numerous bonds and regulations. In particular, data management is quite demanding, both in terms of the time required and the human resources dedicated to this process.

Pharmaceutical companies have been trying to improve the overall quality, timelines and efficiency of their clinical development process. Traditional data management in clinical trials is carried out by performing the following tasks: [1]:

1. collection of clinical trials data by the use of paper case report forms (CRFs);
2. delivery by a courier or postal system;
3. manual coding;
4. data entry - single or double;
5. computerised consistency and missing data checks;
6. and manual creation and processing of data request forms (DRFs)

About 10 years ago the pharmaceutical industry started using systems for Remote Data Entry (RDE) in achieving three main aims:

1. cutting clinical trial duration time,
2. saving resources,
3. and improving data quality.

In a survey of investigators recently published in Germany, the majority of the participants believed that the use of RDE has benefits compared to the paper based system and most of them would use a RDE system again [2]. An important component of a RDE-System is a (client) database in the participating centre, which is replicated with a central database in an organisation responsible for data management (i.e. the pharmaceutical company or a Contract Research Organisation (CRO)).

The Internet influenced the RDE-approach in such a way that a Web browser is directly presenting the electronic CRF (eCRF) using HTML forms and the data is immediately stored in the central database. The largest Internet based clinical trial is the INternational VErapamil/trandolapril STudy (INVEST) to investigate the safety and efficacy of two different initial medication strategies for hypertension [3]. INVEST will randomise 27,000 patients with hypertension and coronary disease to either a calcium antagonist or non-calcium antagonist-based medication strategy and each patient will be treated for at least 2 years.

INVEST and other groups [4] present solutions which demonstrate that HTML and the Web browser can be used as a graphical user interface (GUI) for database applications in clinical trial management. The disadvantage of using HTML is the very time consuming realisation of extensible and flexible data systems. These limitations have been partly overcome in the EXtensible Markup Language (XML). The following chapters will describe the disadvantages of HTML-based CRFs and present a more flexible and powerful approach using XML and Java-Servlets.

2. The HTML-based eCRF

The general architecture of the system developed by the Department of Biometry Hannover Medical School (see [5] for descriptions in German language or [6] for a similar approach described in English language) is based upon the classical World Wide Web two-tier model, where Web-clients access the application, located on a web server, with an ordinary Mozilla-compatible web browser program.

The main advantage of this approach is the independence of particular client programs. A web browser can be considered as a part of the operating system or at least it can be installed easily. A second advantage is that it is not necessary to convince the department of a hospital which is responsible for security to configure their firewall to allow (point-to-point) communication with non-standard protocols.

The web server usually runs on a set of common gateway interface (CGI) [7] programs, often written in the Perl [8] programming language and backed by a relational database management system (RDBMS) engine. Data flow confidentiality is ensured by the use of the Secure Socket Layer (SSL) communication protocol. Additional JavaScript [9] code embedded in the HTML forms is used to carry out data validation checks on the client side. In some systems the data validation checks are carried out only using CGI but we believe that this approach is not very user friendly. Using CGI, data checks can only be done after the whole CRF was filled out and sent to the server. Using database applications the user expects an error message as soon as any error can be detected. When the user gives an answer to an input field where the patient's weight should be filled in and he is entering a value of, for example, 800 kg a notification should be presented as soon as the user leaves the input field. JavaScript can solve this problem on client-side and can present an alert box in case anything is wrong.

Having a closer look at Internet applications one will find two oneway streets: one connection to the web-server taking data from the presented forms and another connection from the web-server where the collected data is usually presented in a textual style (figure 1).

In database applications we normally see the items which is already stored in the database in the same form elements which were used to gather the data. A system designer can use server side JavaScript (SSJS) [10] or, for example, PHP [11] as a server side, HTML embedded scripting languages to achieve the desired behaviour. The mixture of the logical structure of the forms and the data which should be stored into the database makes it difficult to provide highly interactive GUIs or eCRFs for clinical trials.

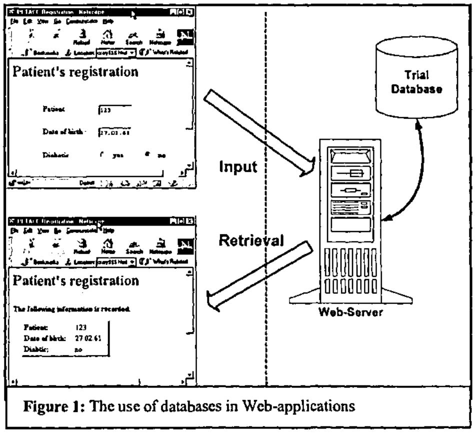

Figure 1: The use of databases in Web-applications

Using XML based documents the process of gathering information on eCRFs and storing them in a relational database can be done much more efficient by the inherent splitting of the data into the logic and the presentation of the data.

3. The Extensible Markup Language XML

XML is based on the Standard Generalised Markup Language [12, 13]. It is a simplified form of SGML that contains tried-and-true features of SGML but with reduced complexity.

Figure 2 shows a simple XML document representing a patient's ID in a clinical trial. It can be seen that the document is well structured and it is easy to store the content in a relational database. The XML document is only presenting the structured *content* of the application (the content of the eCRF). Another SGML derivate, the Extensible Style Language XSL is used to describe the appearance (radio buttons, pulldown lists, ...) of the document and can also be used to completely rearrange the input elements for a particular purpose. If necessary XSL is also able to restructure XML into other document formats (i.e. PDF). Since most available web clients use HTML as their lingua franca, one needs a stylesheet to convert XML in HTML (more precisely, one needs to convert to XHTML which is the XML form of HTML). The Java Apache Project offers the tool *Cocoon* [14] as a publishing system which is able to separate web development in three different layers: content, style and logic. Cocoon is even able to discriminate between different browsers, allowing different stylesheets to be applied. This separation can be very useful in applications where eCRFs are used and is described in further details.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet href="example.xsl" type="text/xsl"
media="Mozilla"?>
<?cocoon-process type="xslt"?>

<demography>

  <patient>
            <initials>P.W.</initials>
            <birthday>12.05.1970</birthday>
            <gender>M</gender>
            <dateOfAssesment>16.01.2000</dateOfAssesment>
  </patient>

</demography>
```

Figure 2: Example of an XML-file

4. XML-based CRFs

Figure 3 describes how XML-documents are used in our approach to realise eCRFs. The XML document presents the content and Cocoon is used to convert the document into HTML. The style of the document (the input fields, the check-boxes etc.) is described in the associated (XML) stylesheet. The user fills in the eCRF, edit checks are done using JavaScript routines and after pressing the submit button the data is sent to a servlet written in Java. This servlet acts as a DOM document transformation system. The DOM (Document Object Model) defines interfaces for document elements (that is tags), text nodes (where the actual text inside the tags is kept) and quite a few other things not discussed here. Figure 4 is a diagram of the servlet that can transform one XML document into some other formats programmatically.

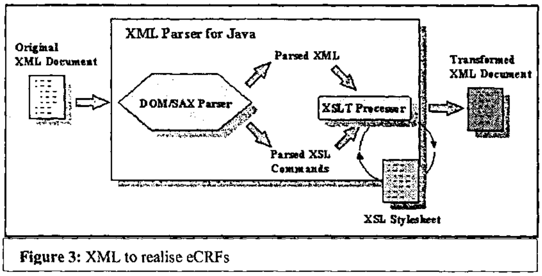

Figure 3: XML to realise eCRFs

The servlet uses a DOM parser to parse an XML file, and the parser returns a tree that is the exact representation of the XML in the file. This can be done only in just a few lines of code. The document tree can be traversed and can be modified - either by deleting nodes or updating their values etc. The modified tree can be used to construct a new XML file which can be converted into HTML and presented to the user with the help of Cocoon. The tree is also used to store the data into an Oracle database either using an XML tool or using a JDBC connection.

In case the data of a patient should be updated or completed, the patient's data which are already stored in the database are retrieved by the servlet and the XML document is created with the help of the DOM tree. The only variation is that the values in the DOM tree are different from the empty XML document providing the default values for the eCRF (either empty fields or predefined values). Again, an XSL stylesheet is used by Cocoon to generate the HTML page. The programmer has to write much less code compared to a PHP script because a lot of tools (the parser, the JDBC protocol, the Cocoon transformation) are available and support the developer. Another advantage is the independence of this solution from the hard- and software. Any database can be used if a JDBC driver is available. Also any Web-server which is able to run servlets can be used (in fact Cocoon is also a servlet) and any browser can be used because the XML document can be converted to HTML.

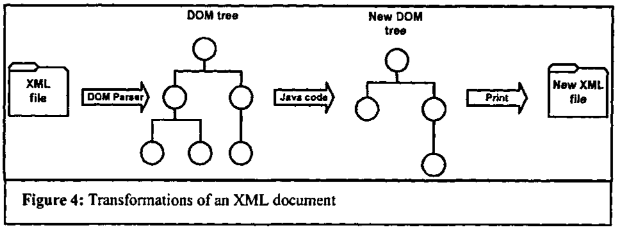

Figure 4: Transformations of an XML document

5. *Conclusion and further aspects*

The Internet and the World Wide Web have recently been introduced into the management of some aspects of large-scale clinical trials such as the dissemination of information on trial process, randomisation and monitoring processes and remote data entry. Guidance documents already reflect long-standing regulations covering clinical trial records and also addresses requirements of the electronic records and electronic signature rules [15].

HTML based systems have already been set up using CGI for server side tasks and/or JavaScript for client side edit checks. The web based approach can be improved by using XML even though browsers which can display XML are still rare. The benefit of using XML is the splitting of the application into logic, style and presentation. Further advantages in the near future might be the use of the XML Schema Language [16] which discusses datatypes that can be used in a XML Schema. The XML Schema Requirements document defines requirements to be fulfilled by this specification. Important aspects of the XML Schema Language are:

1. it provides primitive data typing, including byte, date, integer, sequence, SQL & Java primitive data types, etc.;
2. it defines a type system that is adequate for import/export from database systems (e.g., relational, object, OLAP);
3. it distinguishes requirements relating to lexical data representation vs. those governing an underlying information set;
4. it allows creation of user-defined datatypes, such as datatypes that are derived from existing datatypes and which may constrain certain of its properties (e.g., range, precision, length, format). These datatypes can be specified for element content.

A further advantage of using XML is the associated Document Type Definition (DTD). The DTD specifies what elements may exist, what attributes the elements may have, what elements may or must be found inside other elements and in which order. Validating parsers read XML documents, verify that it is well-formed and construct the document structure as a tree of objects (this is also done by nonvalidating parsers), and then go on to determine whether the document tags are legal, whether the attribute names make sense, whether each element is nested properly and so on. A DTD can be useful for example in supporting 'Standard Operation Procedure (SOP)' in a Clinical Trial. In case your demography CRF for instance should always include certain items you can achieve this with the help of a DTD. Although we are not using it at the moment it can be useful, especially if more than one person or more than one centre designs eCRFs. As a lot of the rules mentioned above are incorporated into the XML document which represents the eCRF, the parsers can check those constraints.

References

[1] Kubick W. R.: The Elegant Machine: Applying Technology to Optimize Clinical Trials; In: Drug Information Journal; Vol. 32 pp. 861-869, 1998 (http://www.diahome.org/abstract/dij1257.pdf)
[2] Ahrens K., Schueren van de F.: Der Prüfarzt und Remote Data Entry (RDE) - Eine Umfrage. In: Wurzinger, S. (Hrsg) Walter-Jung, B. (Hrsg.): Gesundheitspolitischer Strukturwandel - Herausforderungen für die Medizinische Dokumentation, 1999, S. 95-99.
[3] Hompage: International Verapamil/Trandolapril Study; http://invest.biostat.ufl.edu/
[4] Santoro E., Nicolis E., Franzosi M.G. Tognoni G.: Internet for Clinical Trials - Past, Present and Future In: Controlled Clinical Trials 19: 194-201, 1999
[5] Papers about Remote Data Entry: http://www.mh-hannover.de/institut/biometrie/Projekte/Webprojekt/
[6] Scognamillo R., Strozzi C., Vincenzi B., Reccia G.: Clinical Trial Management and Remote Data Entry on the Internet; Drug Information Journal; Vol. 33 pp 1061-1065, 1999 (http://www.diahome.org/abstract/d3341061.pdf)
[7] CGI Programming OpenFAQ; http://www.boutell.com/faq/index.html
[8] Wall L., Schwartz R. "Programming Perl", O' Reilly & Associates 1991; http://www.ora.com and http://www.perl.com
[9] A Beginner's Guide to JavaScript; http://jsguide.simplenet.com
[10] Netscape DevEdge Online - JavaScript Developer Central; http://developer.netscape.com/tech/javascript/index.html
[11] Homepage: PHP; http://www.php.net
[12] Homepage: World Wide Web Consortium W3C; http://www.w3c.org
[13] W3C Architecture Domain Extensible Markup Language (XML); http://www.w3.org/XML
[14] Apache XML Project, Cocoon; http://xml.apache.org/cocoon/index.html
[15] FDA Guidance for Industry: Computerized Systems used in Clinical Trials; http://www.fda.gov/ora/compliance_ref/bimo/ffinalcct.htm
[16] W3C working draft XML Schema Part 1: Structures (http://www.w3.org/TR/xmlschema-1) and XML Schema Part 2: Datatypes (http://www.w3.org/TR/xmlschema-2)