Uppsala Master's Thesis
in Computing Science 138
Examensarbete DV3
1998-12-08
ISSN 1100–1836

Investigator's Workbench

Fredrik Duprez

Computing Science Department
Uppsala University
Box 311
S-751 05 Uppsala
Sweden

This work has been carried out at
PharmaSoft AB
Box 1237
S-751 42 Uppsala
Sweden

Supervisor: Ola Strandberg, PharmaSoft AB
Examiner: Mats Nordström, Computing Science Department, Uppsala University

Passed:____________________________________

Abstract

The administration of applications for approval of a new drug is taking more and more time, both for the drug industry as well as the regulatory agencies. On one hand pharmaceutical companies want the submissions to be handled and approved as fast as possible, and on the other the regulatory agencies have to adapt to the rapidly increasing number of submissions. Both parties would thus benefit from a more efficient storage, retrieval and navigation of the material, leading to easier access.

The current review environment was investigated at both the Swedish Medical Products Agency and the Dutch National Institute of Public Health and the Environment. The different attempts in making standards for electronic submissions were tested and discussed. One of these attempts is called MERS, *Multi agency Electronic Regulatory Submission Project*. MERS is a collaboration between the European authorities and PharmaSoft and uses SGML to structure the information. Using the structure provided by MERS a prototype was made that shows that by marking up information with SGML/XML, it should be possible to develop a more efficient review environment than there exists today.

Contents

List of Figures

List of Tables

1 Acknowledgments

This is a Dissertation submitted for the Degree of Master of Science in Computer Science at the *Computing Science Department* at *Uppsala University*, Sweden, July 1998 and the work has been carried out at *PharmaSoft AB*, Uppsala. The text was written in LaTeX 2_ϵ [12].

I would like to thank Stan van Belkum at RIVM for the valuable input as well as the generous hospitality. I also would like to thank Sven-Erik Hillver at the MPA and Peter Salomons at RIVM for suggestions and comments in the early stages of the work.

At PharmaSoft I would like to thank Per Manell for his generosity and enthusiasm, which has made me feel very welcome. I also want to thank my supervisor Ola Strandberg for his support during the project.

Fredrik Duprez, Uppsala, December 1998.

2 Introduction

The development of a new drug is a time consuming process, where the pharmaceutical company must conform to a number of demands regarding how to conduct and document quality and efficacy tests of the product. When a drug is considered as good as ready for the market a *New Drug Application* (NDA), is submitted to the regulatory agency in the area where the product is to be sold. An NDA is divided into several parts and is usually delivered on paper in a number of binders. The usual amount of information is between 100–200 binders.

In the United States the agency is called FDA[1]. The EU has agencies in all member states, but it is possible to submit an NDA to one of them saying that the product is meant for the whole EU market, and then it's up to the agency in question to make sure that the product conforms to the rules of all EU member states.

Since the evaluation of an NDA is a rather time-consuming process, there have been several projects trying to make the evaluation process easier to handle.

Today there is a catch 22 situation, where both the authorities and the companies submitting the information want a standardized way to hand in the submission and a faster way to investigate it, but no one wants to take the initiative. PharmaSoft is commited to developing tools for making the overall work flow easier.

One of the basis for easier handling of the NDA is that the information is stored in a structured way. This will allow the information to be searched more efficiently, and make it possible to access different parts concurrently. Since there are strict rules defining how an NDA must be assembled, it is possible to structure it using SGML - *Standard Generalized Markup Language*.

SGML is an ISO standard [11] and stores the data in plain text, which makes it platform independent. This makes it easy to comply to the strict archiving requirements set by the authorities. At the same time SGML allows the data to be stored in a database for efficient access, which is the key to reduce the time needed to review an NDA.

There has been an increasing workload concerning the reviewing of new and modified medical substances over the last years. The regulatory agencies in Europe gets funding for the review work on the basis of the number of NDAs they investigate. This has led to a situation where the regulatory agencies have to make the turnaround time shorter, and thus decrease the resources needed to investigate an NDA. Some of the time consuming processes today is to remember if an NDA regarding a similar substance has been reviewed before and if so in an efficient way compare the decisions the agency made in that case with the new information. The same problem arises when a variation, i.e. an update for an existing approved drug, comes in.

2.1 The drug development process

The process of developing a new drug is divided into several steps, and the time span from initial research to an approved new drug is between ten to fifteen years. A description of the steps can be found in [17] and are presented briefly in this section.

[1] U.S. Food and Drug Administration

The two main steps are *preclinical studies* and *clinical trials*, which in turn are divided into substeps.

The preclinical studies usually start with a search for an active substance, i.e. the effective part of a drug. This step includes literature- and patient studies, synthetization in laboratory scale as well as the application for a patent. The following step contains studies of the effects on animals, determining dosage requirements as well as synthetization in a larger scale. The last step is submitting a report called *Investigational New Drug* (IND) to the authorities. If the IND is approved, the clinical trials can be initiated.

The clinical trials has three initial phases followed by the submission of a *New Drug Application* (NDA), and if the NDA is approved a fourth phase will follow. The first phase is studies on healthy human beings as well as continued studies on animals. The second phase is patient studies on a limited amount of patients (50–200 individuals). The third phase is comparative studies on a large number of patients (500–5000 individuals). The documented results of the preclinical studies together the clinical trials is submitted to the authorities as an NDA. If the NDA is approved the fourth phase, that deals with long term effects among other things, can commence.

When the NDA has been approved and the drug reaches the market the development continues, trying to maintain the quality as well as lowering the production costs. This usually mean that the assembly process may change slightly for instance. Since the authorities only have approved exactly the methods stated in the NDA a *variation* must be submitted for each change made in the manufacturing process. These variations can be considered as a new version of relevant parts of the original NDA.

2.2 The structure of a New Drug Application

The NDA, also called submission in this report, is divided into three main parts: quality, safety and efficacy. The quality part deals with the chemical quality of the drug. It is divided into a chemical and a biological part. This part addresses things like the chemical structure of the drug, how it is produced etc. The safety and efficacy parts are both divided into a preclinical and a clinical part, where the preclinical part deals with the effects on microorganisms and animals and the clinical part refers to clinical studies on humans. All of these parts have official names on the form Part X, where X is a Roman number between one and four.

The quality part, which is discussed in this report, is called Part II and has a number of specified subsections, which are described in table 1 on the facing page.

2.3 The investigators' review environment

The investigator must have an efficient tool available for conducting the work. The tool should provide a user specific environment with the possibility to adapt to the users preferences and needs. For instance when reviewing a certain part of a submission the environment should provide the guidelines associated with that part.

The demands were discussed with reviewers at the Swedish MPA and the Dutch RIVM. These discussions resulted in an overview of what's available today as well as the reviewers' wishes concerning the ideal environment.

Part name	Subsection name	Description
Part IIa	Composition	This subsection deals with the chemical composition of the drug
Part IIb	Methods of Preparation	This subsection briefly covers the preparation methods that didn't work, but focuses on the current method. This part also covers how the production result is validated
Part IIc	Control of Starting Materials	This subsection describes how the active ingredient is obtained. If some kind of reference sample has been used (e.g. an extra purified batch).
Part IId	Control tests of Intermediate Product	Usually a rather small section. One example is the manufacturing of plasters, which are manufactured as two separate parts and then glued together.
Part IIe	Control of Finished Product	Describes how the final result is controlled
Part IIf	Stability	The minimal requirements are six months of accelerated tests with extreme heat, cold etc. In addition twelve months of real time data, which is allowed to be interpolated if the accelerated tests indicates that it is reasonable

Table 1: The subsections of the quality part of an NDA.

The review work consists of navigating through the information provided, using existing guidelines that defines what kind of properties the different parts of the submission must obey. Using these guidelines the investigator should produce a recommendation stating whether or not the drug can be approved.

Today the investigator's review environment consists of the submission in paper form and a word processor used to write the report with. This is a very inefficient way of conducting the work, especially when relevant parts of old submissions are required. Because of this a number of initiatives have been made to set a standard for handling the submissions electronically. Three examples of such projects are DAMOS, MERS and SEDAMM.

2.3.1 One initiative - MERS

The first SGML based project for structuring a drug submission is called MERS - *Multi agency Electronic Regulatory Submission Project*. MERS aims towards setting a standard for storing drug submissions using SGML. The advantages would be a platform-independent storage, that would allow easy navigation and various ways to present the information. MERS has so far resulted in a DTD[2], that specifies a subset of the submission, namely the Quality part.

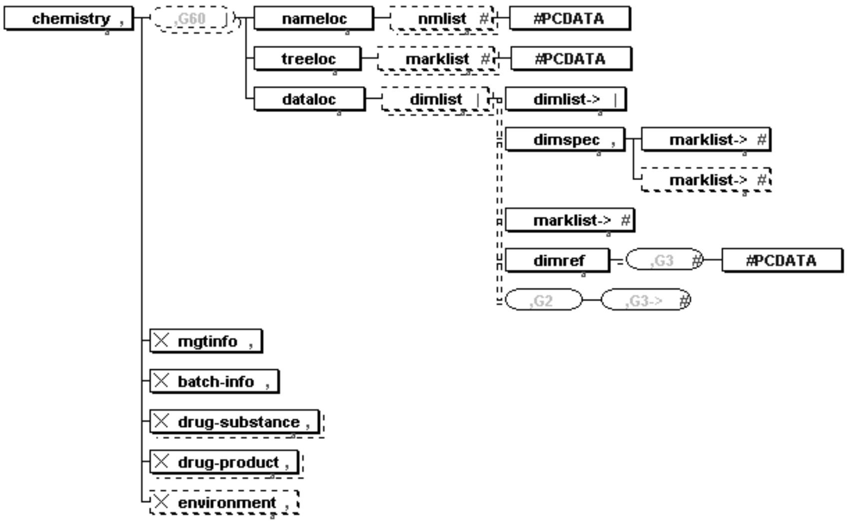


Figure 1: Graphical overview of the MERS-DTD

The MERS DTD does not use the subsections described in table 1 on the preceding page to structure a submission. However the granularity is so fine that it is possible to map the information carrying parts of the MERS DTD onto the submission structure. The differences has to do with the fact that submissions look slightly different in the US and the EU. The MERS DTD has a structure that the FDA uses, and the submission structure is the one used in the EU. The parts that have to be in the submission as well as the argumentation that leads to the conclusions that the drug has the desired effect is essentially the same in both regions, making it possible to use the same structured storage for them. This also has the advantage that a submission can be written once, and automatically transformed into the standard used by the reviewing agency.

2.3.2 Other initiatives

Today there are a few other electronic submission initiatives:

- DAMOS - Drug Application Methodology with Optical Storage (A German made system), stores the submission in TIFF-format[3], i.e. scanned images of the pages. This is the most used system today. It is very closely connected to a viewing software called PharmBridge, which is used to gain some functionality.
- SEDAMM - Soumission Electronique de Dossiers d'Autorisation de Mise sur le Marché (A French made system), stores in HTML. This is today just an effort to create a standard. There have been no submissions so far using this standard.

[2]Document Type Definition. A description of the DTD concept can be found in section 2.9.1 on page 11.
[3]TIFF is a tag-based image file format. It is a platform independent format that is widely used in desktop publishing.

- PDF-format[4]. It uses the Adobe Acrobat software to view and navigate the submission. This approach makes the table of contents click-able as well as allowing annotations in the text. This is no standard, so the submission tends to look different depending on who submitted it.

DAMOS is free to use, so our system has to be made available for free or at a low cost to compete. We must also prove that our system provides performance that is better than the existing systems, in terms of scalability and usability. The main advantage of this system would be the independent data format, which will provide a large flexibility in terms of which client program to use when the information has to be accessed.

2.4 Applications used today for the review environment

2.4.1 PharmBridge

One used and approved environment is PharmBridge, from Image Solutions Inc., a software that is tightly connected with DAMOS. PharmBridge has added a navigational framework on top of the TIFF files that DAMOS uses. This makes it possible to hyper link between different pages in a dossier. The links are made by the manufacturer of the drug and cannot be changed by the reviewer.

The possibility to make annotations is included, but it is not possible to print a report containing the original information with the annotations added.

One of the obvious problems is the fact that all the information value has been removed from the document. If say the original submission is written in a word-processor of some sort, the information could more or less easily be transferred to another machine readable format with the possibilities to e.g. do searches on the material. A quite ironic feature of the PharmBridge software is the ability to use OCR[5] software in order to regain the information that got lost in the making of the TIFF files.

This is agreed by the reviewers to be the best system that is used today. However a number of complaints have been stated. Since the annotations can not be printed together with the original document, the report containing the annotations must be composed from scratch with the possibilities of errors when transferring the text manually to another document. The readability of the pages is not that good, and it's not possible to change the font, or size of the text since each page is a digital photocopy of a paper.

2.4.2 Adobe Acrobat

With Adobe's Acrobat Reader a serious attempt was made to set a standard for WYSIWYG portable documents. This software uses a file format called Portable Document Format (PDF), which is widely used on the Internet today.

[4] Portable Document Format. PDF was developed by Adobe and is aiming towards maintaining the look and feel of the original document when viewed or printed. The only software that can view or print PDF files comes from Adobe.

[5] Optical Character Recognition - the process of converting a printed document to computer readable text

The information is machine readable and thus searchable. A problem though is the fact that this format is centered around being WYSIWYG, so there is no context sensitive information. The document format is only accessible using Adobe Acrobat Reader.

Since there is no standard today regarding how submissions using Adobe Acrobat Reader should be structured and hyperlinked, this environment is not approved by the authorities.

2.5 The purpose of this project

The inefficient way submissions are handled today motivates an investigation whether a more useful system could be built on top of a structured model like MERS.

The vision is to create a knowledge base that can be used by all parties in the process of developing and using a drug, i.e. the pharmaceutical industry, the authorities involved, the pharmacies, doctors as well as the patients. All the information regarding a certain drug would then be possible to store in a virtual document. Various tools could be used by the different people involved to view and be able to alter the information depending on their role. The pharmaceutical company would need some kind of editor to be able to create the type of document needed. The investigator at an authority would need a tool that could present the information in a standardized but customizable way as well as letting him/her make annotations at appropriate places when reviewing the submission.

The structured storage would allow the same document to be used under a drug's whole life cycle, from a New Drug Application to variations and safety updates. An investigator could for example be able to look at the original submission when a variation comes in, and even follow a possible chain of variations in an easy way.

This work will be mostly about how to use the information encoded in the MERS-DTD to achieve some sort of Investigator's Workbench. With Investigator's Workbench we mean a tool for the investigator at an agency where he/she can access the information in the submission, make annotations, get suggestions on where to look for related information and so on.

We want to show that semantically marked information can be used to achieve among other things, the possibility to pick out certain pieces of information and store it into another database, keeping the knowledge about what kind of information is stored. E.g. you can from a generated fact sheet showing tabulated information about a drug substance get a question whether you want to store this information in your own substance database, with all the data fields already entered.

2.6 The scope of the project

A complete system handling everything from how the information should be stored, transferred between authorities and pharmaceutical companies and a efficient review environment is much too complex for this project, so some limitations had to be made.

This project will concentrate on trying to show that using information encoded with MERS a review environment could be built with support for easy navigation. Actual database storage needs to be thoroughly investigated, an investigation that was left out of this project. The database storage was however investigated enough to show that the structure provided by

MERS can be utilized by an object oriented database model. An object oriented database with special support for structured documents was tested and the results are discussed in section 7.

2.7 Possible architectures for the structured storage

Taking only the structured storage into account, a few alternative system designs were sketched on:

Editable SGML in the database

If the submission can arrive in electronic editable form, it would be possible for the authorities to have a database that is used both for storing the submission as well as adding annotations to it. This would also make possible what is called *the interactive IND/NDA*, a concept described in [16]. The interactive IND/NDA process as described in this article exploits the possibilities to submit individual parts of an NDA as the company finishes them. It would also be possible for the reviewing agency to demand for more information regarding a certain part without the company having to hand in the whole NDA. Figure 2 shows an overview of this architecture.

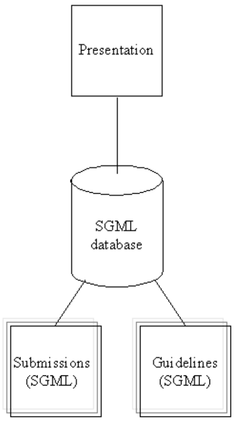


Figure 2: Structured storage, editable SGML

Submission in read-only format

Depending on the legal issues that has to be taken into account, it's possible that the submission has to be made on a non-editable medium, e.g. CD-ROM. Figure 3 on the following page shows an overview of this architecture.

This does not prevent the authorities from copying the submission and add it to their own database as an editable structured document. If this is done, the authorities can add annotations and edit parts of the text and return it to the company as a complete document.

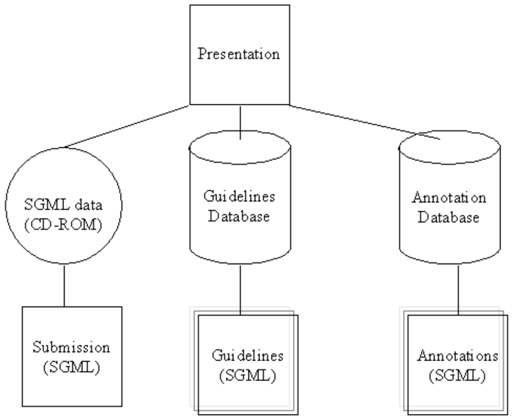


Figure 3: Structured storage, read-only submission

2.8 Possible architectures for the presentation

In order to present semantically marked information, there must be some sort of mapping from the actual data to a readable form. Stylesheets is the common name for this kind of mapping and there exist a couple of different stylesheet languages that are associated with SGML and other structured formats. Two applications using stylesheets are Panorama and Microsoft Internet Explorer.

Panorama

Panorama from SoftQuad is a more or less fully featured SGML editor. It is capable of viewing and navigating through SGML documents conforming to the HyTime Standard. It uses it's own stylesheets to present the SGML information.

Microsoft Internet Explorer

Microsoft Internet Explorer is a web browser that makes it possible to use the well known HTML format for presenting and navigating through information. It is produced by Microsoft and the current version is 4.01. In this respect it does not differ from other web browsers on the market, but it's built in support for XML makes it unique today. The XML support makes it possible to use XSL stylesheets to present XML data directly in the browser. This possibility is further investigated in section 6.

The most desirable solution

The ideal electronic format allows all submissions made to be indexed and searchable. To achieve this the review environment must be able to access the information and make cross-references between e.g. drugs with the same active substance.

A solution which uses an editable submission complying to a structured data format has the advantage that the format can be used as the transfer format between authorities and pharmaceutical companies.

2.9 Overview of different technologies

2.9.1 SGML

SGML is an acronym for *Standard Generalized Markup Language*, and specifies the syntax and semantics of e.g. HTML. SGML is a meta language that focuses on the structure of the information, i.e. disregarding the visual properties of a document. The information is structured using elements which are marked up using tags. Tags are often used in pairs, making it easy to see where a certain piece of information starts and ends. A tag usually starts with a "<" character and ends with a ">" character. In between there is plain text describing the contents of the tag. An end tag contains the same text as the start tag, but is preceded by a slash (/). For example a tag pair describing an e-mail address could look like:

```
<EMAIL>fredrik.duprez@pharmasoft.com</EMAIL>
```

A tag-pair including the text contained within is called an SGML element.

Elements can have attributes specifying special information. The syntax for attributes is as follows:

```
<EMAIL ATTRIBUTE_1=''VALUE_1'' ... ATTRIBUTE_N=''VALUE_N''> ... </EMAIL>
```

An SGML document consists of three formal parts, the SGML declaration, the document type definition (DTD) and the document instance. The declaration defines among other things which delimiter characters to use. The properties of the declaration is usually the same for all documents in an SGML installation. The DTD is written in SGML and defines the structure of a SGML document. The order of elements and the possibilities to nest elements is defined here. The document instance is an application of the rules in the DTD, i.e. the elements mentioned in the DTD are used according to the semantics of the DTD. To check that an SGML document instance is valid a parser is used. The parser takes the document and validates it against the DTD.

Figure 4 on the next page shows a non-complete DTD with definitions of some of the SGML elements. The DTD is divided into four main parts which will be described a bit further below.

The first part consists of external entity declarations except elements and DTDs. An entity is a mechanism for replacing text within an SGML document. For instance an entity, *today*, could be declared as the corresponding weekday, e.g. Monday. With this declaration the SGML document will contain a reference to this entity by containing the text *&today;*, which means that this text will be replaced by the parser according to the entity definition. The replacement text could be defined within the entity declaration:

```
<!ENTITY today ''Monday''>
```

```
<!--*********************************************************************-->
<!-- NAME OF ENTITY:-//MERS//DTD Quality v3//EN                          -->
<!--*********************************************************************-->

<!--          Produced by using SGML COMPANION (R)(TM), S.I.D.E.         -->
<!--          PharmaSoft AB.                                             -->
<!-- DOCUMENT ELEMENT
 chemistry                                                               -->
<!--                                                                     -->
<!--*********************************************************************-->

<!--*********************************************************************-->
<!-- EXTERNAL ENTITY DECLARATIONS EXCEPT ELEMENTS AND DTDS               -->
<!--*********************************************************************-->

...

<!ENTITY  % ent4         PUBLIC "ISO 8879:1986//ENTITIES Added Latin 1//EN">
%ent4;

...

<!--*********************************************************************-->
<!-- ELEMENT DECLARATIONS                                                -->
<!--*********************************************************************-->
...

<!ELEMENT   batch-info       - - ((nameloc|treeloc|dataloc)*,
                                   introduction?,batch-usage-subst,
                                   batch-usage-prod,formulation-table?,
                                   subst-batch-information?,
                                   subst-batch-results?,
                                   prod-batch-information?,
                                   prod-batch-results?)
                                             --(960930) Batch information-->
...

<!ELEMENT   chemistry        - - ((nameloc|treeloc|dataloc)*,
                                   mgtinfo,batch-info,drug-substance+,
                                   drug-product+,environment?)
                                                              --chemistry-->
...

<!ELEMENT   drug-product     - - ((nameloc|treeloc|dataloc)*,
                                   pharmaceutical-form,dev-pharm,
                                   Manufacturing,regspecs,
                                   batch-analysis.prod,ingredspecs,
                                   container-closure,stability.prod,
                                   labelling,microbiology?)
                                                          --DRUG PRODUCT.-->
<!ELEMENT   drug-substance   - - ((nameloc|treeloc|dataloc)*,
                                   nomenclature,characteristics,
                                   physical-chem,structproof,source+,
                                   mfg-processes,impurities,ref-standard+,
                                   regulatory-controls,
                                   batch-analysis.subst,
                                   packaging,stability.subst)
                                                         --DRUG SUBSTANCE-->
...

<!ELEMENT   environment      - - ((para|table|figure|dformgrp)*,section*)*
                                                            --environment-->
...

<!ELEMENT   mgtinfo          - - (date,type,submitter+,drug-name+)
                                                                --mgtinfo-->
...

<!ELEMENT   treeloc          - O (marklist*)
                                       --Tree location-addressing element-->
...

<!--*********************************************************************-->
<!-- ATTRIBUTE DEFINITION LISTS                                          -->
<!--*********************************************************************-->

...
                                                                        >
<!ATTLIST   chemistry
                                                             --chemistry--
applic NAMES  #IMPLIED
                                                                --applic--
hytime NAME  #FIXED "hydoc"
                                                                --hytime--
id ID  #IMPLIED
                                                                    --id--
                                                                        >
...

<!--*********************************************************************-->
<!-- EXTERNAL ENTITY DECLARATIONS: DTDS AND ELEMENTS                     -->
<!--*********************************************************************-->
<!ENTITY  % mersform        PUBLIC "-//MERS//ELEMENTS Formula//EN"        >
%mersform;
<!ENTITY  % calstab      PUBLIC "-//MERS//ELEMENTS CALS table fragment//EN">
%calstab;
```

Figure 4: Example of the parts of a DTD

The text could also be placed in an external file accessible using a SYSTEM or PUBLIC identifier. A SYSTEM identifier is usually a path to a file in the local file system, while a PUBLIC identifier is a name that is resolved by the SGML parser and points to different things depending on the environment.

The second part contains the element declarations. The root element of this DTD is called CHEMISTRY. The two minus-symbols (-) succeeding the name says that this element must have both start and end-tags, i.e.

```
<CHEMISTRY> ... </CHEMISTRY>
```

must be in the document instance. Now follows the rule saying which elements can appear within a CHEMISTRY element and in which order. A comma sign states that an element comes after another, e.g mfg-info precedes batch-info in the example. A plus-symbol (+) after an element says that the element appears at least once and might appear up to an infinite number of times. An asterisk (*) after an element says that the element might appear zero times up to an infinite number of times.

The elements can be grouped separated by pipe-symbols (|) which mean that only one of the elements in the group can exist in the document instance. However it is possible to use the plus or asterisk after a group as well.

The third part contains attribute definition lists. Here the names of the attributes as well as the type of information they contain is stated. The type of information contained in an attribute could for instance be an ID, which means that the parser must check that the attribute value in the document instance is unique. It could also be plain text.

The fourth part uses entities to incorporate more elements and DTDs into the original DTD. This means that the definition of a CALS TABLE is a DTD in it self located in a place pointed out by it's PUBLIC identifier.

SGML is an established ISO standard [11], and is described in [8].

2.9.2 HyTime

In order to handle hyperlinks within and between SGML documents an additional framework is needed. HyTime, which is a short form for *Hypermedia/Time-based Structuring Language*, is an ISO standard [10] which addresses this. The standard itself is quite large containing a great variety of aspects. In this project only a small subset has been used, namely the hyperlinking within a document. HyTime uses SGML syntax, so an SGML parser validating the document against a DTD will have no problems with the extra elements used for anchors and references.

2.9.3 DSSSL

SGML makes it possible to structure a document according to it's contents, concentrating on the type of information. When looking at the document it should be possible to have a set of rules telling how different elements should be presented. The most used standard for this in the SGML community is DSSSL, *Document Style Semantics and Specification Language.*

DSSSL is an ISO standard [9] and there exist a number of applications on the market that uses DSSSL to transform SGML documents to document formats that concentrate on the appearance, e.g. $\TeX$ and HTML.

One of the downsides with DSSSL is that element reordering isn't possible. That possibility is needed if a SGML storage without redundant information is requested.

2.9.4 XML

XML is an acronym for Extensible Markup Language, and it is called extensible since it is not a fixed format like HTML. It is designed to enable the use of SGML on the World Wide Web. XML is not an ISO standard, but it has a formal definition [19] made by the World Wide Web Consortium. XML is designed to be a simplified subset of SGML, but there have been some additions made that are not conformant with the SGML specification.

The purpose of developing XML is to get a standardized way of presenting data on the Web, making it possible to separate the way the information should be presented from the actual information. XML has a more stringent definition in the sense that elements must either have corresponding start- and end-tags or be declared as empty elements using a new syntax not available in SGML. An empty element must not be followed by text, but only an empty element, a start-tag or an end-tag.

An empty element, here called EMAIL, is written as

```
<EMAIL ATTRIBUTE_1=''VALUE_1'' ... ATTRIBUTE_N=''VALUE_N''/>
```

i.e. by putting the slash character immediately before the end delimiter in the document instance.

XML can be used with or without a DTD. If it is used without a DTD the requirements on the document instance is that it should be well formed. Well formed documents consists of combinations of corresponding start- and end-tags and empty elements.

2.9.5 XSL

Extensible Style Language, XSL, is an upcoming standard [18] describing rules for presenting XML documents. The standard is presented by the World Wide Web Consortium. XSL can be used for mapping different XML elements to a display style, but it can also be used for rearranging elements in the original document before the information is displayed.

3 Functional requirements on a structured storage

There are a few things that have to be fulfilled by the storage. These requirements have to be supported in the database system to be used, and will be investigated in this project. Some of the key features are discussed in [5] and [6] and are presented briefly below.

3.1 Authentication

A document that is submitted to an external party must be tagged with a signature that guarantees that the document has arrived without any changes done to it during the transport.

If a quote is to be made then the submitter might require that a whole paragraph is used instead of a small part of text. If this is the case that paragraph must be tagged with a signature that can be used to see whether the quote is legitimate or not.

3.2 Changes in data format

All non-textual pieces of information in an SGML document are treated as external entities, which means that pictures for example are stored separately as binary data. If an image format changes we want to be able to check out that component, convert it and check it into the database again.

The text format used is also important. Should we use the UNICODE[6] standard or stick with standard 8-bit ASCII[7] and use textual entities which we easily can change the meaning of? You could for example use the entity *°* to describe a degree character which the repository could convert to a desired code when the document is to be exported to another medium.

3.3 Managing hyperlinks

There must be a way to manage hyperlinks between objects in such a way that a link that references to a certain document has to be updated if that document moves or is removed from the storage. This solution will make it possible to avoid duplicate data in the database. The possibility for the reviewing agency to add local hyperlinks when reviewing the NDA is also important, otherwize the submitting company can have the NDA prepared with a fixed set of links and thus preventing the reviewer to get an objective view of the contents.

3.4 Annotations

Annotations, including information about who wrote it, should be connected to the document in some way. Either by adding a link to an external file containing the annotation or by including the annotation in the original document. The annotations in combination with the

[6]Unicode is a 16-bit character coding system designed to be used worldwide. It is currently under development.

[7]American Standard Code for Information Interchange - a standard binary coding scheme for characters.

original text is used by the authorities to produce their report, stating whether the drug can be approved or not, so there is a strong demand for a possibility view and print a document containing all this information.

3.5 Granularity

Since one single drug submission is a large amount of information, the accumulated information when storing many submissions is vast. If the submissions could be divided into smaller logical parts, a database engine would be able to search the information space in significantly reduced time.

3.6 Legal requirements

A drug on the market today usually has a life span of about ten to fifteen years, with occasional variations. To this the development time of the drug when the information is created must be added. The development time is between ten to twelve years, and when estimating the whole lifetime fifty to sixty years are mentioned. This makes it important to store the information in a format that is guaranteed to live that long.

3.7 Why use SGML as the storage format?

When choosing an electronic archiving format it is important to make sure that the information is retrievable after an unknown period of time. With the fast evolution in the computer software market, products that are setting the standard today might have disappeared tomorrow. A storage based on a file format depending on a certain program should not under any circumstances be accepted.

SGML is a well established ISO standard [11] that has been stable since 1986. SGML is based on plain text, which means that we might store a backup copy of the information on paper if we take it to the extreme. It would then be possible to use OCR techniques to retrieve the information and end up with machine readable information. The plain text source makes it possible to write a parser and other processing tools if the software used would disappear.

Since all SGML documents uses a DTD their structure can be validated at all times.

PDF format is dependent on specific software in order to access the information, which makes it unfit for this kind of data. Since the PDF format isn't human readable, the information might be worthless if the software disappears or doesn't run on future computers.

DAMOS is also dependent on specific software if the benefits of making annotations and the possibilities to follow hyperlinks is to be utilized. In addition to this DAMOS has removed information value from the original application by simply store digital photocopies of pages, making the transfer to an electronic medium pointless.

4 Formal demands for the review environment

Using the structured storage an infrastructure must be built that makes it possible to use the information in a way that aids the reviewer in his/her tasks. Some work has been done in this field, e.g. [3] uses a HTML form to gather information about what the user wants and then the system retrieves the compiled information. In [1] and [2] a general approach towards generating a complete document from several microdocuments is investigated.

The storage should be a database with support for structured information like SGML. It should be possible to check components in and out independent of each other. Navigation could take advantage of the structure of the data making it possible to make context-sensitive searches within different parts of the document as well as within the corresponding parts of different documents.

4.1 Microdocuments

The concept of microdocuments is described in [4] as a reasonably self-contained fragment of the document. In SGML that corresponds to one or more elements within a document instance. One or more sections within a paragraph could for instance be viewed as a microdocument.

The subset to be displayed should not be too large. In [7] these matters are discussed and has been taken into consideration.

4.2 Data gathering and retrieval

The principle of gathering microdocuments is very appealing when there is a structured SGML storage to fetch the information from. There should be a possibility to get individual pieces from the database, splice them together, and deliver them as one unit. This unit has to contain the information needed to distinguish between the different pieces and thus be able to locate their place in the database.

This technique could be used on all three parts used:

- The SGML data for the submission
- The SGML data containing the corresponding guidelines
- The XSL stylesheets for the corresponding elements in the submission as well as the guidelines.

Ideally the database could deliver the fragments in XML format, so that the system could retrieve all pieces and splice them together into one XML document for the current view of the submission, one for the corresponding guidelines and one XSL stylesheet for each. Now the system just has to convert the documents into HTML for presentation in a web browser.

By using different attributes in the HTML tags, there would be a reference back into the database about each piece. This would make it possible to utilize the context sensitivity in the generated HTML.

A motivation for using XML at the user level is that Web browsers soon will be able to handle XML directly, removing the need for converting the data to HTML. Word processors are also evolving towards the ability to export documents in XML format.

4.3 A context sensitive editing environment

If a paragraph is to be edited in the submission an editing environment should be available, which makes sure that the edited information doesn't break any rules set by the definition of a submission.

4.4 Using agents to propose suitable actions

The user would demand a certain part of the submission to view, and it would be presented in a way that he/she prefers. Using SGML markup, the corresponding part of the official guidelines could also be extracted and presented by some kind of interactive agent saying that guidelines are available for this particular part.

When reviewing a drug that has the same active substance as one already in the repository, the agent should inform the reviewer that more information is available.

Using agents would also make it possible to log the current users activities and learn his/her personal preferences.

4.5 Maintaining information value

The only approved format today, DAMOS, has sacrificed the information value when transferring the submission onto an electronic medium. If the information used to draw a chart is stored within the submission, the reviewer would have a possibility to rearrange the chart in different ways, making it easier to compare the results. This is a very much asked for feature that clearly would speed up the review time of a submission.

It is also desirable for the reviewer to be able to extract information from the submission and either use it as a quote in an investigational report or store different data in a local database. When quoting, digital signatures could be used to guarantee that the quote is legitimate, i.e. not modified and sufficiently long for the context to be clear.

5 Modeling the review environment in combination with the repository

In this section a model of the review environment and the structured storage will be presented. Parts of this model has been realized in a prototype which is described in the following chapter.

5.1 The repository

The demands for the repository is that it should be able to handle and store the submissions, guidelines as well as the information needed to transform them into a presentable format. It must also have access control and be able to log who edited something. The repository should have the possibility to check out subsets of the submissions as microdocuments making it possible for multiple reviewers to investigate a certain submission concurrently.

In order to exploit the hierarchical structure of the SGML documents an object-oriented approach could be used.

5.2 The review environment

The review environment should provide active help for the reviewer, being able to give suggestions on the appropriate course of action as different parts of the submission are viewed. It should also be possible to extract information from a submission in order to store it in a local database.

The presentation of the information could be made with HTML in a web browser in order to have a well known interface to text with hyperlinks. Using dynamic HTML it would be possible to have references back to the microdocument originally fetched from the database so that editing of that specific part could be done.

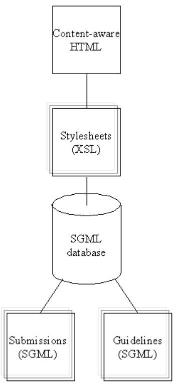


Figure 5: The review environment in combination with the repository

6 Prototype made

In order to test how information could be presented by the user a prototype was made which uses the Microsoft XML parser that can be integrated with Internet Explorer 4.0. The idea is to show that it is possible to generate different reports depending on the different agencies' preferences. Figure 6 shows how the information could be presented.

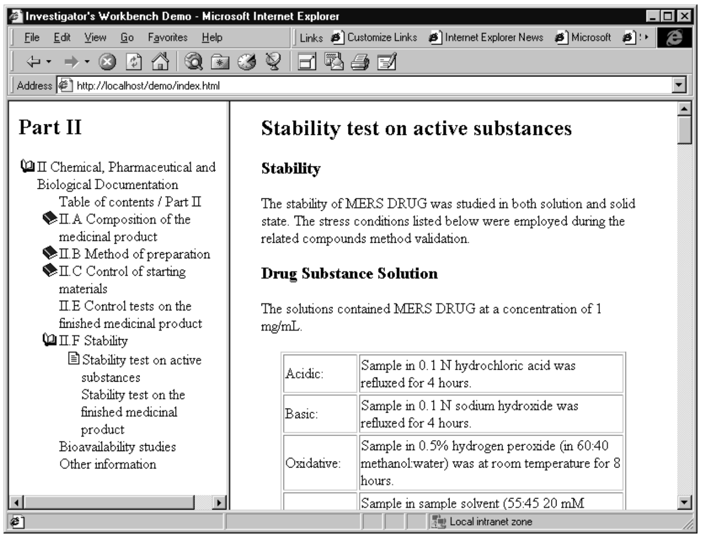


Figure 6: Screenshot of the prototype

The prototype was demonstrated at the International Reviewer Forum in London the 27th of April 1998. The participants of the conference were reviewers from different countries in the EU. The purpose of the demonstration was to get feedback from the different reviewers about the user interface and the general idea to use dynamically generated HTML instead of scanned papers.

6.1 Choosing an environment

The reason for using XML and Internet Explorer is the rapid evolvement in the market towards XML native applications, where today only Internet Explorer has enough support for XML. Internet Explorer is free and widely used, so that means that the potential customers might already have the tools needed to make use of XML data. This environment also makes it possible to create a review environment exactly as the customer wants it without him/her having to know about the structure behind the user interface.

The possibility to reorder elements is needed if you want to be able to present things differently depending on the viewer. This kind of functionality is missing in DSSSL, so an approach using SGML and DSSSL could not have been realized.

The possibility to make tailored documents from different microdocuments means that the generated document would need a generated DTD to be validated against. The generated document would however not need to be validated if it's sufficient to have references back to the original place in the database where the different microdocuments were fetched from. This also makes DTD-less XML interesting at the user level.

6.2 Generating the SGML data

The raw text containing a subset of the quality part of a fictional submission was provided by the FDA. With help from the Swedish MPA the information was encoded according to the MERS DTD.

Tables were included in the MERS DTD by using a subset of the HTML version 4.0 standard [20]. This was accomplished by adding an external entity TABLE to the MERS DTD and let it point to this definition.

The finished SGML document was then converted using SX (an SGML to XML converter) into a DTD-less XML document. A small part of the generated XML is shown in Appendix A. The generated XML document could now be parsed and processed using Microsoft's ActiveX XSL processor version 1.0 and the resulting output was HTML that could be presented in a web browser.

In order to process the generated XML document a few XSL stylesheets were written. Documentation regarding XSL syntax can be found in [15] and an example of one of the stylesheets used is in Appendix B. The stylesheets made it possible to traverse the XML tree and pick out elements in a desired order. The selected elements were then transformed into suitable HTML elements according to the rules given in the XSL stylesheets. An example of how the resulting HTML looks like in a web browser is shown in Appendix C.

The table of contents was implemented as an expanding/collapsing outline. The tree structure follows the subsections described in table 1 on page 5. The sublevels are taken from a dummy submission. The outline is essentially an unordered HTML list, but with the use of DHTML [13] and JScript I could exploit some features in the Document Object Model [14] to make it dynamically expandable.

HyTime linking within the generated HTML document was realized by generating name anchors at every table and figure. The generated names could be accessed at parse-time by the XSL processor, making it possible to convert the references in the original document to HTML references.

Figures were included in the generated HTML as clickable thumbnails, to ensure that e.g. a chart wasn't too large to fit in the current window. When the user clicked on an image a new browser window, stripped from navigational buttons and toolbars, was opened. This imageviewer window had the possibility to show the image in original format as well as constraining it to the window size. Figure 7 on the facing page shows a screenshot of the imageviewer window. To avoid cluttering with a lot of opened windows, all images were opened in the same imageviewer window.

The prototype does not address the problems with linking between documents or how to generate table of contents from the original document.

A possibility to extract information from the XML data and store it in a local database was visualized with a button in one part of the submission, see Appendix C. When the button was pressed, selected information from the page was extracted from the original submission

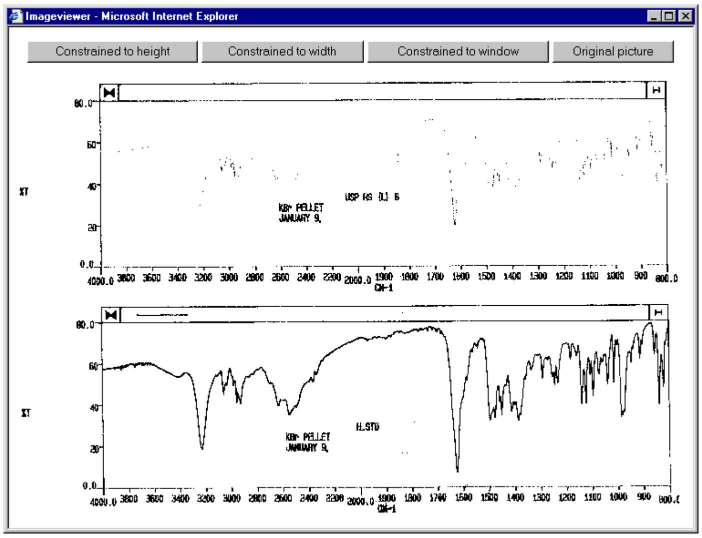


Figure 7: Screenshot of the imageviewer window

and presented in a dialog box. The dialog gave the reviewer the possibility of storing the data locally after making sure that the information was the same as in the original submission. A screenshot of the dialog box is shown in figure 8.

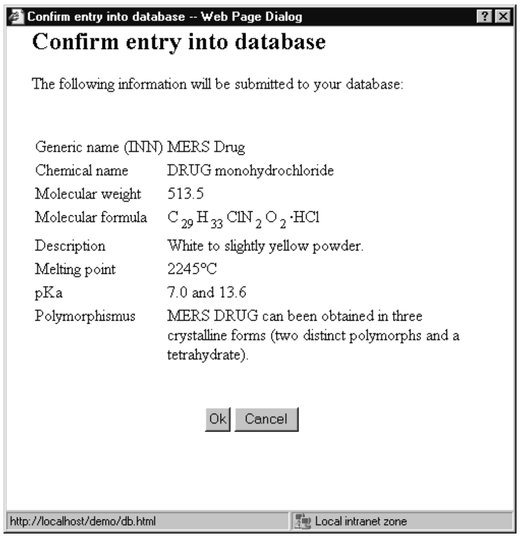


Figure 8: Screenshot of the database entry dialog box

The possibility to use annotations was implemented as a small icon next to the original text. When the icon was clicked, the annotation text was shown as emphasized text with a different color than the rest of the document. A click on the icon when the annotation was shown resulted in hiding the annotation again. This made it possible to print the document with or without the annotations. Figure 9 on the following page shows a screenshot of an annotation.

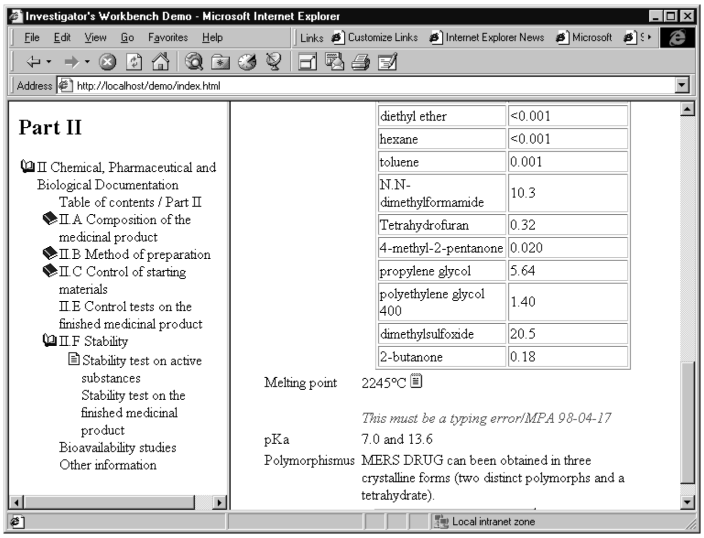


Figure 9: Screenshot of an annotation

6.3 Conclusions

Using the metaphors of the web browser for following hyperlinks was acclaimed by the reviewers that saw the prototype. The possibilities to alter the font-size and view images in a customized way also caused a positive reaction. The possibility to extract information from the submission and store it in a local database also suited the reviewers' needs.

If the database provides enough support for generating the appropriate XML data and XSL stylesheets, this solution would be possible to realize.

7 Evaluation of a database manager

The second quarter of 1998 POET software intends to release POET Content Management Suite, hereafter called CMS, an add-on to their object-oriented database system. POET has focused on efficient support for structured documents, i.e. SGML and XML. A part of this project has been to evaluate and test the beta version of the software to see if it might be used as the base for a structured storage of submissions. To fully exploit the features of the CMS the Software Development Kit, Content SDK, had to be studied.

7.1 Concepts

Some terminology used in POET has to be explained in order to understand how the database utilizes the hierarchical structure of the SGML data stored. The most vital concepts include the publication specification, the SGML document tree and the SGML class hierarchy.

7.1.1 The Publication Specification

The CMS uses a Publication Specification in which it specifies which Components a document could be divided into. A Component is a mapping to an element in the document, so the Component essentially tells which elements can be checked in or out independent of the original document. Using Components makes it possible to select the level of granularity to use for a certain document. Several Components can be checked out and edited independent of each other.

7.1.2 The SGML Document Tree

One common metaphor used for structured documents is that of a tree. The documentation to the Content SDK refers to the hierarchical structure of an SGML document as the *document tree*. In the Content SDK this document tree represents the entire contents of the database, which can contain multiple documents. The root Object is called *PSDbGlobal* and each database is guaranteed to contain only one instance of PSDbGlobal. A Publication Specification is identified using an object called *PSDbPublicationSpecification* and each document has an object called *PSDbPublicationVersion* as the root object of the document.

A number of classes are used to represent the different nodes and leaves of the SGML document tree. This hierarchy of classes is called the *SGML Class Hierarchy*.

7.1.3 The SGML Class Hierarchy

To support SGML in CMS, it has mapped the structure of an SGML document onto a class hierarchy. This makes it possible to traverse a document by e.g. following a certain type of element. This architecture is available to the programmer, so that any part of the SGML document can be accessed efficiently.

An SGML document imported into the database gets divided into Components according to the publication specification. These components are mapped onto PSDbComponentVersion

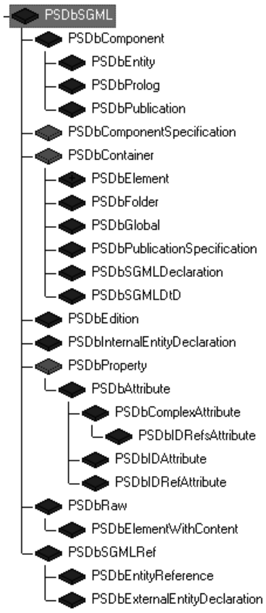


Figure 10: The SGML Class Hierarchy in CMS

objects. The SGML elements that aren't specified as components gets mapped onto a PSDbElement object. This different handling has to do with the fact that the SGML structure is dynamic in a component, meaning that when a component is checked out, new elements might be added or removed before the component is checked back in. If the structure has changed a new PSDbComponentVersion object is needed to store the component. Otherwise rollback to previous versions wouldn't be possible.

7.1.4 POET Content SDK navigation API

This API[8] exposes the SGML class Hierarchy to the programmer allowing detailed navigation in the SGML tree. The navigation API consists of a number of C++ classes that makes the SGML class hierarchy accessible. There is also a number of support functions for navigating the tree.

7.1.5 POET Web Factory

POET Web Factory is a concept for integrating access to a POET database via the web. Using an executable file analogue to a CGI[9] script, the contents of the database can be accessed. The information gets streamed to the web browser and it is possible to process the stream through a filter, i.e. SGML content can be converted to XML before reaching the browser.

[8] Application Programming Interface

[9] Common Gateway Interface - a technique where an application running on the web server sends a text stream, usually a HTML document, to the web browser

7.2 Tests performed

The software was installed and different functionality was tested. Checkin/checkout of individual components using the interactive client software was performed, as well as some traversal of the SGML class hierarchy using small test programs written in C++.

7.2.1 Individual checkout/checkin of individual components

The test document was imported into the database with a relatively high granularity and components on different levels were checked out and checked back in.

One of the problems here is that the microdocuments exported might have references to other parts of the main document, confusing the SGML parser when the component is to be checked back in. This would not be acceptable in the final release.

7.2.2 POET Web Factory

In combination with Microsoft's ActiveX XSL processor it should be possible to extend the made prototype into generating the XML data on the fly from the database's SGML data.

In order to test this, a web server was installed and configured to work with the POET Web Factory. A HTML page containing the Microsoft XSL processor as an ActiveX component was designed analogous to the prototype, but instead of using a filename for the XML file, a special URL was used. This URL made it possible to make POET retrieve the SGML document from the database, and deliver it as a character stream that had passed through SX, the SGML to XML converter.

8 Conclusions and future work

8.1 Annotations should be part of the original document

When using an object oriented database and SGML as the document storage, there is a clear advantage to include the annotations in the document. The DTD can easily be adapted to accept annotations in different contexts. Since the document can be checked out from the storage in small specified pieces, there is no extra cost in size when handling a large document. Using a chain of document versions, it is possible to look at a version of the document before and after the changes were made.

8.2 Hyperlinking within and between documents

Since the database engine had problems with internal hyperlinks when a Component was to be checked back in if the linkend was not within the current microdocument, this will have to be tested again with the final release of the software.

8.3 Handling the large volumes of data

This project only addressed a minor part of a submission, and the data used to test the prototype was not really complete. A complete submission with a number of variations and safety updates is in it self a much larger data volume. With the addition of a number of different submissions it's clear that the database to be used must be tested thoroughly with much larger amounts of data in order to see if it is suitable for the needs. Also when migrating to an electronic medium the possibilities to incorporate multimedia, e.g. a film showing the manufacturing process, could make tha data volumes of an application to grow even more.

References

[1] Helena Ahonen, Barbara Heikkinen, Oskari Heinonen, Jani Jaakkola, Pekka Kilpeläinen, Greger Lindén, and Heikki Mannila. Intelligent Assembly of Structured Documents. Technical Report C-1996-40, University of Helsinki, Department of Computer Science, June 1996.

[2] Helena Ahonen, Barbara Heikkinen, Oskari Heinonen, Jani Jaakkola, Pekka Kilpeläinen, Greger Lindén, and Heikki Mannila. Constructing tailored SGML documents. In J. Saarela, editor, *Proceedings of SGML Finland 1996*, pages 106–116, SGML User Group Finland, Espoo, Finland, October 1996.

[3] Helena Ahonen, Barbara Heikkinen, Oskari Heinonen, and Pekka Kilpeläinen. A system for assembling specialized textbooks from a pool of documents. Technical Report C-1997-22, University of Helsinki, Department of Computer Science, March 1997.

[4] Helena Ahonen, Barbara Heikkinen, Oskari Heinonen, and Mika Klemettinen. Improving the accessibility of SGML documents - A content-analytical approach. In *Proceedings of SGML Europe '97 Conference*, pages 321–327, Barcelona, Spain, 13-15 May 1997. Graphic Communications Association.

[5] Ulf Andersson. SESAM - Philosophy and Rules concerning electronic archives and authenticity. Astra Finance, 1996.

[6] Ulf Andersson. *Workshop on Electronic Archiving.* the Swedish National Archives and Astra AB, 1997.

[7] Eric D. Freese. The transformation of sgml documents for presentation on the world wide web.
`http://www.sil.org/sgml/freese.html`.

[8] Eric van Herwijnen. *Practical SGML.* Kluwer Academic Publishers, 1994.

[9] International Organization for Standardization, Geneva, Switzerland. *Document Style Semantics and Specification Language (DSSSL).* International Standard ISO 10179:1996.

[10] International Organization for Standardization, Geneva, Switzerland. *Hypermedia/Time-based Structuring Language (HyTime).* International Standard ISO 10744:1997.

[11] International Organization for Standardization, Geneva, Switzerland. *Information Processing, Text and Office Systems, Standard Generalized Markup Language (SGML) = Traitement de l'information, systemes bureautiques, langage standard généralisé de balisage (SGML). First edition, 1986-10-15.* International Standard ISO 8879-1986. Federal information processing standard; FIPS PUB 152.

[12] Leslie Lamport. *LaTeX - A Document Preparation System.* Addison-Wesley Publishing Company, 1986.

[13] Microsoft Corporation. *DHTML References*, 1998.
`http://www.microsoft.com/msdn/sdk/inetsdk/help/dhtml/references/dhtmlrefs.htm`.

[14] Microsoft Corporation. *Document Object Model References*, 1998. http://www.microsoft.com/msdn/sdk/inetsdk/help/dhtml/references/domrefs.htm.

[15] Microsoft Corporation. *XSL Tutorial*, January 1998. http://www.microsoft.com/xml/xsl/tutorial/default.asp.

[16] Loren Miller, Jill Buckley, and Richard Crawley. The interactive IND/NDA. *Drug Information Journal*, 30:593–599, 1996.

[17] Riksrevisionsverket. Den statliga läkemedelskontrollen. Technical Report 1986:58, Revisionsavdelning 3, 1987.

[18] World Wide Web Consortium. *A Proposal for XSL*, August 1997. http://www.w3.org/TR/NOTE-XSL.html.

[19] World Wide Web Consortium. *Extensible Markup Language (XML) 1.0*, February 1998. http://www.w3.org/TR/REC-xml-19980210.

[20] World Wide Web Consortium. *HTML 4.0 Specification*, April 1998. http://www.w3.org/TR/REC-html40/.

A Example of an XML file

```xml
<?xml version="1.0"?>
<CHEMISTRY HYTIME="HYDOC">
   <DRUG-SUBSTANCE>
      <NOMENCLATURE>
         <CHEM-NAME TYPE="IUPAC">DRUG monohydrochloride</CHEM-NAME>
         <OFFICIAL-NAME TYPE="INN">MERS Drug</OFFICIAL-NAME>
      </NOMENCLATURE>
      <CHARACTERISTICS>
         <MOLWGT>513.5
            <XREF HYTIME="CLINK" LINKEND="DRUG-SUBSTANCE"><ANCHOR>drug-substance linktext</ANCHOR></XREF>
            <XREF HYTIME="CLINK" LINKEND="FIGUVSCAN"><ANCHOR>figurelink</ANCHOR></XREF>
         </MOLWGT>
         <MOLFORM>C<SUB>29</SUB>H<SUB>33</SUB>ClN<SUB>2</SUB>O<SUB>2</SUB>HCl</MOLFORM>
         <STRUCTFORM>
            <FIGURE HYTIME="ILINK">
               <TITLE>The structural form of the drug substance</TITLE>
               <GRAPHIC BOARDNO="215a" ROTATE="0" SCALE="0"></GRAPHIC>
            </FIGURE>
         </STRUCTFORM>
         <STEREOCHEM></STEREOCHEM>
      </CHARACTERISTICS>
      <PHYSICAL-CHEM>
      <INTRODUCTION>
         <SECTION>
            <TITLE>Physicochemical Properties</TITLE>
            <PARA>The physicochemical characteristics (i.e. solubilities, pKa and physical characteristics)
            of MERS DRUG are included in this section. </PARA>

            <PARA>MERS DRUG can been obtained in three crystalline forms (two distinct polymorphs and a tetrahydrate). A
            detailed discussion concerning polymorphism is included in this section confirming the drug substance supplied by our
            supplier (SGML LTD.) is form I. This form corresponds to that of MERS DRUG USP reference standard.</PARA>
         </SECTION>
      </INTRODUCTION>
      <PHYSFORM>White to slightly yellow powder.</PHYSFORM>
      <SOLID-STATE-PROP>
         <PARTSIZE>As MERS DRUG is only slightly soluble in water, this material will be micronized (to increase solubility) prior to
         use in the manufacturing of MERS DRUG tablets.</PARTSIZE>

         <CRYSTAL-FORM>
            <POLYMORPH>MERS DRUG can been obtained in three crystalline forms
            (two distinct polymorphs and a tetrahydrate).</POLYMORPH>
         </CRYSTAL-FORM>
         <OTHER></OTHER>
      </SOLID-STATE-PROP>
      <SOLUBILITY>The solubility of MERS DRUG in various solvents is
      listed below.
         <TABLE WIDTH="90%" ID="TABSOLUBILITY">
            <CAPTION>The solubility of MERS DRUG in various solvents.</CAPTION>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP" WIDTH="50%"> Solvent</TD>
            <TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">Solubility in g/100mL solution</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">water (pH = 1.7)</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">0.14</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">citrate-phosphate pH 6.1</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">0.008</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">citrate-phosphate pH 7.9</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">0.001</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">methanol</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">28.6</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">ethanol</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">5.37</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">2-propanol</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">1.11</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">dichloromethane</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">35.1</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">acetone</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">0.20</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">ethyl acetate</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">0.035</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">diethyl ether</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">0.001</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">hexane</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">0.001</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">toluene</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">0.001</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">N.N-dimethylformamide</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">10.3</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">Tetrahydrofuran</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">0.32</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">4-methyl-2-pentanone</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">0.020</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">propylene glycol</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">5.64</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">polyethylene glycol 400</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">1.40</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">dimethylsulfoxide</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">20.5</TD></TR>
            <TR><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">2-butanone</TD><TD ROWSPAN="1" COLSPAN="1" VALIGN="TOP">0.18</TD></TR>
         </TABLE>
      </SOLUBILITY>
      <PKA>7.0 and 13.6</PKA>
      <MELTPOINT>2245C</MELTPOINT>
      <ISOMERISM></ISOMERISM>
      <HYGROSCOPICITY></HYGROSCOPICITY>
      <SPECIFIC-GRAVITY></SPECIFIC-GRAVITY>
      </PHYSICAL-CHEM>


   </DRUG-SUBSTANCE>
</CHEMISTRY>
```

B Example of an XSL stylesheet

```
<xsl>

  <!-- Root rule - start processing here -->
  <rule>
    <root/>

    <HTML>

     <BODY BGCOLOR="FFFFFF" LINK="#993366" VLINK="#663366">
      <children/>
     </BODY>
    </HTML>

  </rule>

  <!-- All undefined elements gets a blue color -->
  <rule>
    <target-element/>

      <SPAN color="blue" title="='&lt;' + tagName + '&gt;'">
          <children/>
      </SPAN>
  </rule>

  <rule>
    <element type="MOLWGT">
      <target-element type="XREF"/>
    </element>

    <empty/>

  </rule>

  <rule>
      <target-element type="GRAPHIC"/>

    <empty/>
  </rule>

  <!-- Make a tabular framework to store the result in -->
  <rule>
    <target-element type="DRUG-SUBSTANCE"/>

    <DIV ALIGN="CENTER">
    <TABLE BORDER="0" WIDTH="90%" CELLSPACING="0">
    <TR>
    <TD>
    <H2>
     <eval>this.children.item("NOMENCLATURE",0).children.item("OFFICIAL-NAME",0).text</eval>
    </H2>
    <H3>
     Chemical and Physical Properties
    </H3>
    </TD>
    </TR>

    <TR>
    <TD>
    <TABLE BORDER="0">
    <select-elements from="descendants">
       <target-element type="GRAPHIC"/>
    </select-elements>

    <select-elements from="descendants">
      <target-element type="OFFICIAL-NAME"/>
    </select-elements>

    <select-elements from="descendants">
       <target-element type="MOLWGT"/>
    </select-elements>

    <select-elements from="descendants">
       <target-element type="MOLFORM"/>
    </select-elements>

    <select-elements from="descendants">
       <target-element type="CHEM-NAME"/>
    </select-elements>

    <select-elements from="descendants">
       <target-element type="PHYSFORM"/>
    </select-elements>

    <select-elements from="descendants">
       <target-element type="SOLUBILITY"/>
    </select-elements>
```

```
  <select-elements from="descendants">
     <target-element type="MELTPOINT"/>
  </select-elements>

  <select-elements from="descendants">
     <target-element type="PKA"/>
  </select-elements>

  <select-elements from="descendants">
     <target-element type="POLYMORPH"/>
  </select-elements>

  </TABLE>
 </TD></TR>
 <TR><TD>
   <P ALIGN="CENTER">
   <INPUT type="BUTTON" onclick="copyToDB();" VALUE="Submit to Database"/>
   </P>
 </TD></TR>

</TABLE>
</DIV>
</rule>

<rule>
  <target-element type="POLYMORPH"/>

    <TR>
   <TD VALIGN="TOP">Polymorphismus</TD>
   <TD>
       <children/>
   </TD>
  </TR>
</rule>

<rule>
  <target-element type="PKA"/>

  <TR>
   <TD VALIGN="TOP" WIDTH="25%">pKa</TD>
   <TD>
       <children/>
   </TD>
  </TR>
</rule>

<rule>
  <target-element type="MELTPOINT"/>

  <TR>
   <TD VALIGN="TOP">Melting point</TD>
   <TD>

     <SPAN ID="mp">
        <children/>
     </SPAN>

     <SPAN ID="foo" onClick="parent.annotate(document, this);" class="clsContent" style="cursor:hand;">
       <IMG SRC="images/annotation.gif"/>
     </SPAN>
     <SPAN ID="fooa" class="clsAnnE" >
     </SPAN>

   </TD>
  </TR>
</rule>


<rule>
  <element type="SECTION">
    <target-element type="TITLE"/>
  </element>

  <H3>
   <children/>
  </H3>
</rule>

<!-- Tag paragraphs with standard HTML P-tags -->
<rule>
  <target-element type="PARA"/>
    <P>
      <children/>
    </P>
</rule>
```

```
<!-- Tag subscripts with the appropriate HTML tag -->
<rule>
      <target-element type="SUB"/>

    <SUB>
      <children/>
    </SUB>
</rule>

<!-- Basic rules for HTML-tables -->
<rule>
  <!-- one of (h3 p div a) -->
  <target-element type="TABLE"/>

    <CENTER>
     <P>
     <A NAME='=getAttribute("ID")'></A>
     <TABLE BORDER="1" WIDTH='=getAttribute("WIDTH")'>
       <children/>
     </TABLE>
     </P>
    </CENTER>
</rule>

<rule>
  <element type="TABLE">
    <target-element type="CAPTION"/>
  </element>

  <P ALIGN="CENTER">
    <FONT size="-1">
        Table <eval>formatNumberList(
                hierarchicalNumberRecursive("TABLE", this), "1", ".")</eval>:
      <children/>
    </FONT>
  </P>
</rule>

<rule>
  <!-- one of (h3 p div a) -->
  <target-element type="TH"/>

    <TH>
      <P font-weight="bold"><children/></P>
    </TH>
</rule>

<rule>
  <!-- one of (h3 p div a) -->
  <target-element type="TR"/>

    <TR>
      <children/>
    </TR>
</rule>

<rule>
  <!-- one of (h3 p div a) -->
  <target-element type="TD"/>

    <TD ALIGN='=getAttribute("ALIGN")' COLSPAN='=getAttribute("COLSPAN")'>
      <children/>
    </TD>
</rule>

<rule>
  <target-element type="SOLUBILITY"/>

  <TR>
   <TD VALIGN="TOP">Solubility</TD>
   <TD>
      <select-elements>
       <target-element/>
      </select-elements>
   </TD>
  </TR>
</rule>

<!-- Entry containing the chemical name -->
<rule>
  <target-element type="PHYSFORM"/>

  <TR>
   <TD VALIGN="TOP">Description</TD>
   <TD>
        <children/>
   </TD>
  </TR>
</rule>
```

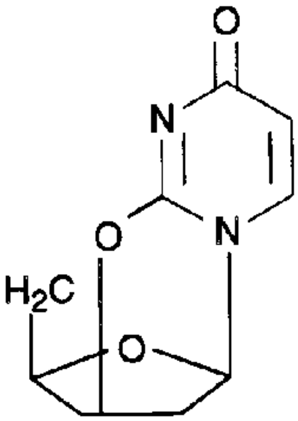
```
  <!-- Entry containing the chemical name -->
  <rule>
    <target-element type="CHEM-NAME"/>

    <TR>
     <TD VALIGN="TOP">Chemical name</TD>
     <TD>
        <children/>
     </TD>
    </TR>
  </rule>


  <rule>
    <target-element type="MOLFORM"/>

    <TR>
     <TD VALIGN="TOP">Molecular formula</TD>
     <TD>
        <children/>
     </TD>
    </TR>
  </rule>


  <!-- Create an entry for the molecular weight -->
  <rule>
    <target-element type="MOLWGT"/>

    <TR>
     <TD VALIGN="TOP">Molecular weight</TD>
     <TD>
         <children/>
      </TD>
    </TR>

  </rule>

  <!-- Create an entry for the official name-->
  <rule>
    <target-element type="OFFICIAL-NAME"/>

    <TR>
     <TD VALIGN="TOP">
       Generic name (<eval>getAttribute("TYPE")</eval>)
     </TD>
     <TD VALIGN="TOP">
         <eval>this.text</eval>
      </TD>
    </TR>

  </rule>


  <!-- Create an entry for the structform image -->
  <rule>
    <element type="STRUCTFORM">
      <element type="FIGURE">
        <target-element type="GRAPHIC"/>
      </element>
    </element>

    <TR>
     <TD VALIGN="TOP">Structure</TD>
     <TD>
     <IMG SRC='="images/" + getAttribute("BOARDNO") + ".gif"' WIDTH="50%"/>
     </TD>
    </TR>

  </rule>

</xsl>
```

C The result of applying the stylesheet

MERS Drug

Chemical and Physical Properties

Structure



Generic name (INN)	MERS Drug
Molecular weight	513.5
Molecular formula	$C_{29}H_{33}ClN_2O_2 \cdot HCl$
Chemical name	DRUG monohydrochloride
Description	White to slightly yellow powder.
Solubility	The solubility of MERS DRUG in various solvents is listed below.

Table 1: The solubility of MERS DRUG in various solvents.

Solvent	Solubility in g/100mL solution
water (pH = 1.7)	0.14
citrate-phosphate pH 6.1	0.008
citrate-phosphate pH 7.9	<0.001
methanol	28.6
ethanol	5.37
2-propanol	1.11
dichloromethane	35.1
acetone	0.20
ethyl acetate	0.035
diethyl ether	<0.001
hexane	<0.001
toluene	0.001
N.N-dimethylformamide	10.3
Tetrahydrofuran	0.32
4-methyl-2-pentanone	0.020
propylene glycol	5.64
polyethylene glycol 400	1.40
dimethylsulfoxide	20.5
2-butanone	0.18

Melting point	2245°C
pKa	7.0 and 13.6
Polymorphismus	MERS DRUG can been obtained in three crystalline forms (two distinct polymorphs and a tetrahydrate).

Submit to Database