Persistent Knowledge Acquisition for Educational Purpose: An Open Distance Learning Website for German Literature and Language Scientists

Alexander Schatten, A Min Tjoa
Institute of Software Technology, Vienna University of Technology
A-1040 Vienna, FavoritenStr. 9-11/188
aschatt@ifs.tuwien.ac.at, tjoa@ifs.tuwien.ac.at

Abstract
Building learning packages and educational software often forces a decision which learning system/software to use, or to develop a system for particular needs. The disadvantage of many systems is, that they are „data islands" in the rapid stream of information flow. Exchange of data especially in learning and multimedia systems was not enough considered in the past, and will be a crucial criterion in the future for preserving valuable information once collected.
Extensible Markup Language (XML) is a very interesting approach to solve this problem of persistent data storage, especially when combined with metadata information and platform independent tools. We used these technologies to build a framework for an Open Distance Learning Course (ODL) „Literatur in der Wiener Moderne" dealing with Austrian Literature. It was desired to record the data required for publication in a highly generic form, which is difficult in the case of lecture data and multimedia information. Nevertheless the aim was to separate data from presentation, for gaining flexibility for future use of the data in arbitrary other (scientific) projects. Moreover as the internet and multimedia technology is evolving with an enormous speed, our approach allows easy adoption of data to future internet or multimedia technology.

1. "Data Islands" and the Need of Data Exchange

1.1. Proprietary Systems, „Learning Software", „Authoring Systems"

The situation of multimedia authoring systems is a very dynamic one and innumerable software vendors try to place their products in this emerging market, with the consequence that many different proprietary systems are available today. The problem is that the ability of data exchange is not always a prior function of these systems. Some producers might have in mind, that a good ability of data exchange allows the users to „easily" migrate to a system of an opponent.
The consequence is, that nearly all systems use proprietary, and even worse, usually binary (non documented) data formats. This is especially true for systems from „smaller" vendors or systems that were designed for certain universities or training institutes.
The lack of common standards makes data exchange between different systems a torture. Sometimes lots of conversion tasks have to be performed manually, if possible at all. At least many of these systems allow to structure the data in a generic way, which offers the possibility for future reuse in different contexts.
Sometimes exchange is only possible in very unstructured data formats like „plain text", HTML etc. Information loss can not be avoided in most of these cases. Figure 1 illustrates this problem: At

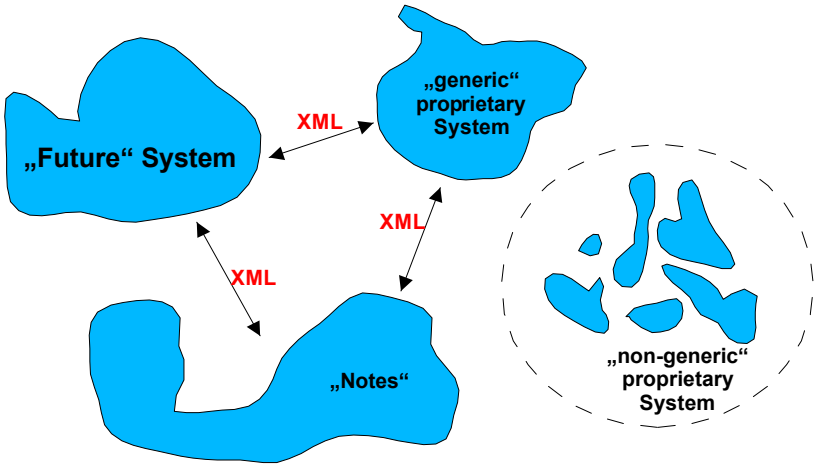


Fig. 1 „Data Islands“: possible communication and data transfer between heterogenous systems.

the moment (blue islands) different systems do exist with proprietary data storage, consistent, and with a (hopefully) generic data structure, but data exchange or use in „Future“ Systems is usually not guaranteed at the moment. As illustrated, XML (as an exchange vehicle for the common representation of the data) could close this gap and offers the possibility to exchange data in heterogeneous environments.

1.2. "non-generic" Approach

Probably the „worst case scenario“ in terms of reusability is the „non-generic“ approach. This means, that documents are created, but information gets lost: Consider *this* document. When printed it offers information for *human readers*, but *machines/algorithms* will not have a chance to reuse information provided on these pages, because it is not clear, what is the *title*, the *abstract*, *keywords*, and so on. One could *guess*, that the first sentence written in large bold face could be the title, but is this certain? In another document the author could use another layout and the first „sentence“ is the author of the article.

1.3. The Generic Approach

Some Multimedia Authoring tools or dedicated „learning software“ like Lotus Notes/Learning Space offers the possibility to collect data in a structured way. The „generic approach“ means, that information is not only „just entered“ into the system (probably in a layout centric way), but also structured with „meta-information“ - *information about the information* is added. This is an important feature, as reusability of data is rather limited if it is entered in some unstructured way without adding information on content structure, which will be described later on.

2. Persistent Knowledge Acquisition

2.1. Introduction

Following the ideas above, it is desired to collect information in a way, that application of information in different styles and different systems is possible, even on systems not yet available (who would have foreseen e.g. WAP cellular telephones three years ago! And today’s HTML pages

are definitely not suited for viewing with these new devices.) On the other hand integration of this information in a bigger context is desired. Such a context could be the Internet. The problem today is obviously that there is lot of information available, but this information is hardly manageable as to few information *about* this information is available. *Is the number 65 on an HTML page a temperature, a stock quote, a price?* Who knows. The human reader will (hopefully) understand it in its textual context, but the machine (e.g. search machine) will not.
At the moment valuable information for educational purpose is created, but usually in a way, that does not allow reuse in a future context. This is especially problematic when dealing with information that has a long lifecycle as for example cultural or historical data.
The idea is to perform the step from creating *educational data islands* to *persistent knowledge acquisition for educational purpose with a high degree of flexibility and reusability*. XML and Metadata initiatives can be a step toward the solution to this problem.

2.2. Generic Data Acquisition

Generic data acquisition is an optimal foundation for building reusable (multimedia) content without loss of information. Moreover it is possible to store generic data very efficiently (e.g. in database systems), with highly flexible access patterns. Of course, GDA demands a clear concept *before* collecting data, to ensure a consistent yet flexible data model.

2.3. Metadata

As mentioned above in acquiring data, meta-information should be considered. Through some initiatives, there are already existing standard for metadata recording (e.g. Dublin Core). However: Tagging information with meta-information enriches the data entered into the system and is the first step in creating persistent knowledge in a broader context.
For example tasks like „Search and Retrieval“, „Library Systems“ and „Agent Technology“ are supported or made available, especially when using standard metadata formats. Metadata can even enrich „conventional“ documents like HTML pages and make the integration into library systems and databases of search machines more efficient.

2.4. XML

2.4.1. Persistent Data-Storage

Extended Markup Language XML is a recent W3C recommendation (Bray 1996, Connolly 1996, 1997). XML is like HTML a SGML application. It showed that HTML has a lot of restrictions while SGML is very powerful but rather complex and was initially not designed for online use.
So essentially XML is much more flexible than HTML but the complexity of SGML was avoided. XML allows to create your “own” markup language, which offers the possibility to define exactly the tags necessary to store the desired information. Syntactically XML is similar to HTML but may not used that “sloppy”. (E.g. no starting tags without end-tags).
XML is a very efficient solution for persistent data storage with integration of metadata. It is a „self-descriptive“ data format, i.e. that it is human-readable and also understandable. The XML tags (comparable to HTML tags) describe the underlying data very clearly. Furthermore it is not a binary, but a text format, hence *platform and system independent*.
A demand in definition of XML was, that it is (especially compared to HTML) easy parseable using an easy, but rigid syntax. The consequence is, that yet lots of parsers are available especially for Java, but also for C++, Perl, Python, Delphi, ... XML is sometimes called the „Java Data Format“. This seems to be true in practice, as Java is highly platform independent as XML is.

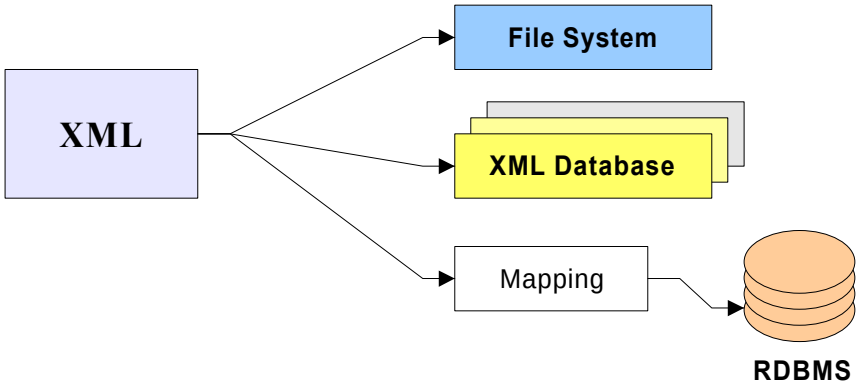


Fig. 2: XML Data Storage

Storage of XML data is possible in different ways (see Fig. 2):

- Storing XML as text file in the „normal“ filesystem is probably the easiest way.
- More enhanced techniques are mapping (hierarchical) XML data into „conventional“ relational database management systems; various tools are available for this task. In the latter option, XML is more a data exchange than a storage format.
- Recently „pure“ XML databases were presented, that allow direct storage of XML data without conversion or mapping to relational systems (Gray, 1999).

2.4.2. Validation

An essential property of XML is data-validation. Using document type definitions (DTDs) or the recent recommendation: schemas, a validation of the document structure (DTD) or even validation of the data types (comparable to RDBMS systems) becomes available. Schemas and DTDs are not necessary (XML can be used without strict declarations), but allow checking of „incoming“ data, e.g. for data exchange between companies in business to business EDI.

2.4.3. Publication

Data Storage and Exchange are one part of the XML story. The other very interesting thing is publication. Basically three different approaches are used:

- „Direct“ publication using stylesheets (e.g. XSL, DSSSL, CSS) and stylesheet engines in the Webbrowser. In this case the XML file is delivered with the according stylesheet and the browser performs the rendering.
- Transformation on the Webserver: On the serverside, the XML file is transformed with the associated stylesheet (or using a Java servlet) to a HTML (or PDF, ...) file. This HTML file is then sent to the browser. The advantage of this method is, that also „older“ browsers can handle the documents.
- „Pre-Transformation“: XML is used for data-storage or as intermediate format and will be „pre-transformed“ into the output publication format (e.g. HTML and PDF). These files are then put on the webserver for static delivery. We decided to use this approach for our Open Distance Learning project.

Moreover „publication tools“ like different office and desktop publishing applications offer XML interfaces. This allows to use for example a DTP software like Frame Maker to publish XML data.

2.4.4. From „search“ to „finding“ machines

As mentioned above, the problem today is not accessing information, which is rather simple using standard internet technology, but *location* of useful information. The point is, that nobody is

interested in *searching machines*, what is needed are *finding machines*. Using metadata and XML the next step in information retrieval is set, as machines can better „understand“ documents and offer more powerful document databases and „finding“ frontends!

2.4.5. Interface to E-Commerce Systems

Most XML systems today are used in e-commerce business to business application. Hence using XML in projects allows easy integration of data into commercial systems like bookstores, data exchange with publishers and off course for import or export of data into other educational systems.

2.4.6. Flexibility/Return on Investment

Summarising the decision to use XML allows to use a broad range of tools and applications. But much more important: XML is an „open“ format, easy in understanding and usage and is well supported by all „big players“ (e.g. Sun, IBM, Lotus, Oracle, Microsoft...) for use on arbitrary target systems (PC, Mac, Unix, Palm, WAP, ...).
This makes XML a good choice especially for data that should be available on future operating systems and devices.

3. Project „Literatur in der Wiener Moderne“

3.1. Introduction

Close contacts between the department of German language and literature of the Paris Lodron University and an increasing number of German Departments in various central east european states have been established within the past 25 years. Thus we have experienced a steadily growing interest for the topic „Literatur in der Wiener Moderne“, focussing a most interesting, shifting cultural scenery between about 1870 and 1910, comprising both the de-sires of incoming students and the demands for outgoing Salzburg professors.
At the same time it became obvious that it would not be possible to cover those demands by means of physical mobility, not only due to limited finances. Therefore it seemed more or less evident to produce an online learning package. The Austrian Ministry of Science and Transport helps to establish a co-operation between the Department of German Language and Literature of University of Salzburg and the Department of Software Engineering (Vienna University of Technology), and in early 1999 those two institutions have started their interdisciplinary work on that specific learning package.
A set of basic ideas were taken into consideration when starting the joint project (following the concepts described above):

- Storage of generic information instead of direct publishing was required for having the option to reuse the data for future projects. So a highly generic representation of the data with a minimal loss of information was pursued.
- Furthermore, for the processing of the data, platform-independent tools were preferred (Java, XML, DOM, JDBC).
- Metadata initiatives like Dublin-Core and RDF are considered for easy interoperability with other information systems (Lassila 1997, DublinCore 1997).

3.2. Data Stored in RDBMS

Information is structured and entered into a relational database management system (RDBMS); We try to collect and structure not only on the surface “hard” structured data, but even “soft” data like lecture texts. Following this approach we avoid the mentioned problems and it is again easier to add meta-information (metadata) like: keywords, subjects, abstracts, links, etc. After having stored the data in the database system further processing of the information is conceivable in a flexible way.
By modelling the information using entity-relationship diagrams, redundant information is avoided. This is not only a “nice side effect” for the technicians, but means also more comfort for the team in Salzburg, as smarter entering of data is practicable. This is especially true for the “multimedia components” (like images, music, URLs, etc.) as each entity has to be entered precisely once, regardless how often it will be used. Moreover particularly these multimedia entities often need maintenance like modifying changed URLs or image filenames and the like. As the access to this data is easily possible through the visual database interface and these entities are stored in a non-redundant way, the change of a URL used many times in the website is one simple change of the database.

3.3. "Middle Format" XML

We registered the following advantages of XML for our project:

- XML is a text-format, hence platform and system independent: This is a relevant as *persistent storage* of the information is desired, independent of special database software or other technological developments/changes
- XML offers the possibility to store the project data without loss of information.
- XML is (contrary to HTML) *easy to pars*e. Consequently lots of free parsers are yet available, many written in Java.
- It is a human-readable and understandable data format. The consequence is, that “every-body” has access to the complete project data without the burden of reading and interpreting binary data formats. This is especially important to store valuable data for future use.
- The RDBMS system is a good tool to store and handle the project data, but data representation in the RDBMS system is not very good suited for publication purpose. The XML files are designed having the destination publication in mind.

3.4. Script Language: Java

Using a RDBMS system for data management, XML as „middle“ format and HTML and PDF as publishing formats we had a conversion problem, that can be divided into (at least) two sub-problems:

1. Generation of XML files out of the database and
2. generation of HTML and/or PDF files out of the XML data.

As mentioned above it seems that Java has adopted XML as “standard” data format, hence lots of free XML parsers of high quality are available. As also the database linking using JDBC (Java Database Connectivity) works reliable, we desired using Java as „scripting“ language.
For generation of XML files, first of all a database connection is required: This connection was performed using JDBC. Having access to the database, the XML library *Project X* from Sun Microsystems (Sun, 1999) was used to build the XML tree. The Project X library then generates the XML data.

The next problem is building the HTML-website. This task is also performed using Java and the Project X library. The XML parser reads the XML files and builds a tree-representation of the data. Using this tree easy access to all parts of the document is conceivable.
Finally the HTML files are generated using HTML-stylesheets to having separated the conversion scripts from design/layout.

3.5. RDF Metadata Initiative

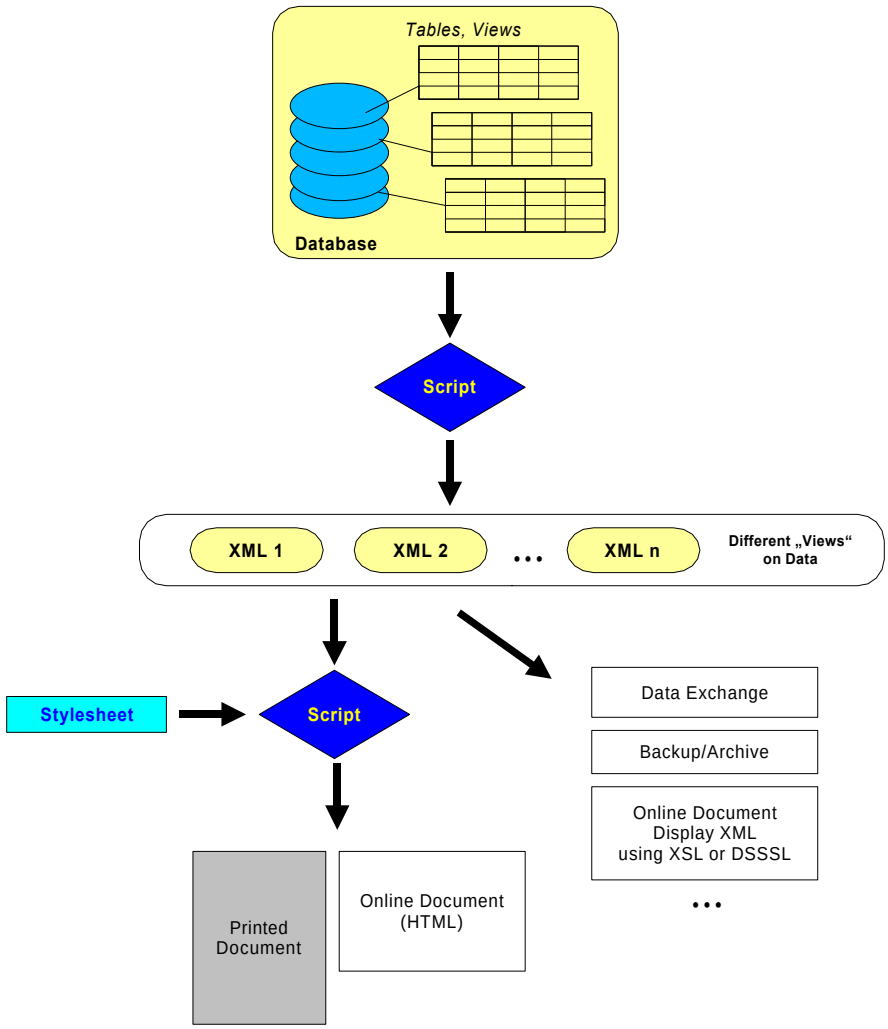
As mentioned above, metadata can be used to „enrich“ even „normal“ HTML pages with additional information. One of the first Metadata Initiatives was PICS: *Platform for Internet Content Selectio*n, that was first of all considered to rate content (W3C, 1996). The successor of this first attempt was the *Dublin-Core Metadata Initiative* (Dublin Core, 1997) and the Proposal for the *Resource Description Framework* (RDF) (Lassilla, 1997). Syntactically RDF is XML based, which is a consistent method designed for future use with XML documents. Nevertheless RDF can be applied to HTML documents too.
Especially in a project where lots of structured information (lexical data, lecture text elements) are available, clear usage of metadata is a wise decision as we hope to be fit for future implementations and developments of electronic library systems.
Also content rating comparable to PICS and providing copyright information is possible using RDF, which is though not that relevant for our work.
But the main aspect will be the knowledge exchange using intelligent software agents. The technology of “intelligent” agents is still a research topic, but will emerge soon as a relevant method for information retrieval in my opinion. As far as we can expect now, RDF will provide the “food” for these software tools.
Moreover the creation of metadata is not a real burden, as our information is well structured in a database system already containing the needed metadata (keywords, subjects, abstracts).

3.6. Design Stylesheets

Separating the process of content acquisition from publishing, allows the integration of design stylesheets and templates. A designer generated a set of HTML stylesheets, which were used for automatically processing of the website.
The consequence of this approach is, that the website has a very consistent layout and modification of the design is easily possible by replacing the style-templates.

3.7. Publishing

The smartest way of publication would have been using XML directly! This is possible in theory as part of the XML activity are definitions of style-languages. Especially two are in discussion: DSSSL (coming from SGML) and XSL.
So the idea is keeping the data without loss of information in XML and publishing this data “through” a style “filter” namely DSSSL or XSL. The problem is, that neither Netscape Navigator not Internet Explorer are capable to perform this task at the moment, but next browser generation (e.g. Gecko rendering engine) should.
Furthermore one other essential element of hypertext, namely linking is currently under heavy discussion, and seems not to be ready for production use at the moment. The consequence was, that we needed to convert the XML data into HTML specially prepared for online purpose (no long text elements per page, clear design for easy online reading ...) and into PDF for off-line use (printing). For future use it is off course desired to direct-publish the XML data using XSL stylesheets.



Nonetheless we separated information from design: A designer generated design-templates for each part of the project and the scripts automatically created HTML pages getting the information out of the XML data and using the design-templates to build the layout.

4. Future Development

We all are positive that already existing well functioning physical academic mobility (SOCRATES, CEEPUS and others) and cross border co-operations will be enlarged and improved by strong virtual supply. Nevertheless this is one of the first steps in this direction. Further German (Austrian) literature projects are planned or already started on the technical basis we developed. Not only adding new content will be part of the future development, also update in new technological directions are desired.

Particularly the next browser generation and their ability to directly render XML will influence the future development from the technological side, as well as upgrades in database management systems and administration of data. Regardless of the direction the internet is developing, we hope to be on the “safe side”, as data is stored and processed using open standards. So at least the core of the project, namely the basic information provided by the German literature scientists should survive the multimedia future.

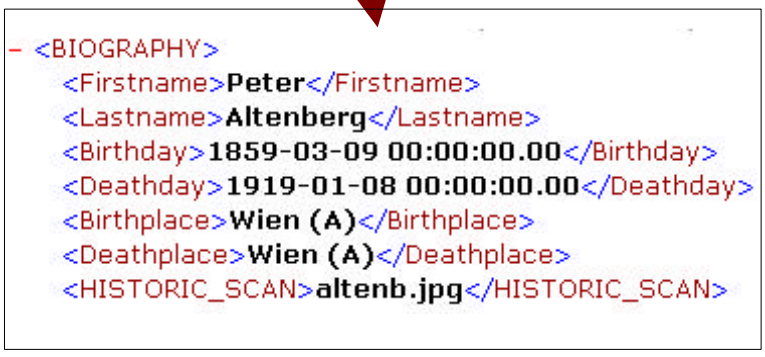
Entering Data in Database Forms

MONOGRAPHY : Tabelle

ID des Artikels	Vorname	Nachname	Geburtstag	Todestag	BIF
1	Ludwig	Anzengruber	29.11.1839	10.12.1889	Wie
2	Johannes	Brahms	07.05.1833	03.04.1897	Han
3	Carl	Karlweis	23.11.1850	27.10.1901	Wie
4	Julius von	Gans-Ludassy	13.04.1858	30.09.1922	Wie

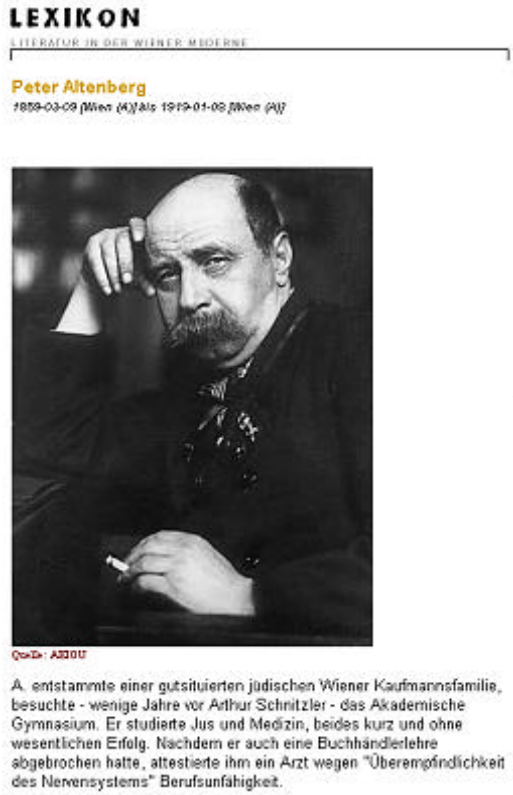
Data Storage in RDBMS

Conversion to XML



```
- <BIOGRAPHY>
    <Firstname>Peter</Firstname>
    <Lastname>Altenberg</Lastname>
    <Birthday>1859-03-09 00:00:00.00</Birthday>
    <Deathday>1919-01-08 00:00:00.00</Deathday>
    <Birthplace>Wien (A)</Birthplace>
    <Deathplace>Wien (A)</Deathplace>
    <HISTORIC_SCAN>altenb.jpg</HISTORIC_SCAN>
```

XML for Publication, Data Exchange
and Persistant Data Storage



LEXIKON

LITERATUR IN DER WIENER MODERNE

Peter Altenberg

1859-03-09 (Wien (A)) bis 1919-01-08 (Wien (A))

Quelle: ASEE

A. entstammte einer gutsituierten jüdischen Wiener Kaufmannsfamilie, besuchte - wenige Jahre vor Arthur Schnitzler - das Akademische Gymnasium. Er studierte Jus und Medizin, beides kurz und ohne wesentlichen Erfolg. Nachdem er auch eine Buchhändlerlehre abgebrochen hatte, attestierte ihm ein Arzt wegen "Überempfindlichkeit des Nervensystems" Berufsunfähigkeit.

Publication in HTML (or PDF, ...)

Fig 3: This figure shows the complete workflow producing the ODL website, starting with database forms and persistant data storage using XML to publishing in HTML or PDF. (Screenshots form Applications and Website.)

5. Go To “Literatur in der Wiener Moderne”

Please visit our site at http://www.sbg.ac.at/lwm.
For further interests in this project please send an e-mail to lwm.interesse@sbg.ac.at

6. References

Bray, Tim and Sperberg-McQueen, C.M. (1996), Initial XML draft, http://www.w3.org/TR/WD-xml-961114.html
Connolly, Dan (1996), Extensible Markup Language (XMLTM) Activity, http://www.w3.org/XML/Activity.html
Connolly, Dan (1997), Extensible Markup Language (XMLTM), http://www.w3.org/XML
Dublin Core (1997), Dublin Core Metadata Initiative http://purl.org/dc
Gray, Carey (1999), Software AG´s Tamino: Butler Group Viewpoint, Analyst Statement, http://www.softwareag.com/tamino/references/Butler_about_tamino.htm
Lassila (1997), Introduction to RDF Metadata, W3C Note, 1997,http://www.w3.org/TR/NOTE-rdf-simple-intro-971113.html
Sun (1999), JavaTM Technology, XML: A Perfect Match, http://java.sun.com/xml/
W3C (1996), PICS Label Distribution Label Syntax and Communication Protocols, Version 1.1, W3C Recommendation, 1996; http://www.w3.org/TR/REC-PICS-labels
Harold, Elliotte Rusty, *XML Bible*, IDG Books (1999)
Leventhal, Michael and Lewis, David and Fuchs, Matthew, *Designing XML Internet Applications*, Prentice Hall PTR, Upper Saddle River (1998)