Kamil ŻYŁA
Jacek KĘSIK

SHARING AND STORING MEASUREMENT RESULTS USING WEB APPLICATIONS BASED ON WebML MODELS

ABSTRACT *This article introduces method of publishing and storing metrological data using MDE techniques and web applications. Thanks to MDE, researchers do not need to master special skills regarding relational databases, server side programming and other advanced programming issues.*

Keywords: *web measurements sharing, WebML, MDE.*

1. INTRODUCTION

Internet gives many opportunities to share and store results of studies. It provides methods to store data in an environment easing sharing and distributed analysing of such data. Quite important factors are also the confidentiality, security and availability of data.

Kamil ŻYŁA, Msc. Eng.
e-mail: kamilz@cs.pollub.pl

Jacek KĘSIK, Ph.D. Eng.
e-mail: kesik@cs.pollub.pl

Institute of Computer Science, Lublin University of Technology

PROCEEDINGS OF ELECTROTECHNICAL INSTITUTE, Issue 252, 2011

WebML is a notation for creating models presenting database structure, content management and communication within web pages. Its implementation in WebRatio allows creation of web applications based on easy to understand and build WebML models. Created apps can be deployed on any web server handling JSP pages.

This article introduces method of publishing and storing metrological data using MDE techniques and web applications. Thanks to standard set of WebML units, researchers can easily develop a site providing the data they need in a way they like. No advanced knowledge of web-design is required.

2. WebML AT A GLANCE

WebML is in short “a visual notation for specifying the content, composition, and navigation features of hypertext applications, building on ER and UML” [1]. It is an example of Model Driven Engineering technology and can be seen as a high level visual description of a MVC organized web application.

WebML consists of three models: Data Model, Hypertext Model, Presentation Model. Data model describes application data design and is similar to a standard ERD model used in conceptual database design [1]. It utilizes a graphical representation of entities with attributes connected with relationships. Hypertext model describes a composition and navigation of the web-application. It arranges application into views containing areas and pages. Pages contain units responsible for the data presentation. The business logic is projected by the operation units placed either within or outside the pages. Both units types can be connected with links transferring activation and parameters. The common set of units is available for use. The third layer is the presentation model describing the layout of the data presentation units on the pages, the pages layout and the layout of the whole application [2].

3. CREATING APPLICATION MODEL

The data acquired during research, e.g. metrology experiments, is usually a set of periodically probed values. Current equipment can mostly provide these data stored as files. WebML provides an Excel unit, which after short configuration can read data sets from xls files [3].

Hypertext model can use a group of units capable to read such files, extract the measures and store them in relational database. Once it is stored in database, it can be presented in any ways needed, e.g. to the authenticated users. WebRatio provides an authentication grid and proper units to login/ /logout, making it easy to configure rules of access to the data [4].

3.1. Creating data model

Data model of typical application for sharing and storing measurement results consists of three main parts:

1. Personalization subschema – contains entities necessary for allowing restricted access to the stored data (Fig. 1).

Fig. 1. Exemplary personalization subschema

2. BLOBs subschema – contains entities necessary for storing binary large objects e.g. files with: measures, description of experiment, multimedia content etc. (Fig. 2).
3. Series subschema – contains entities being the result of mapping the structure of particular measures (Fig. 3).

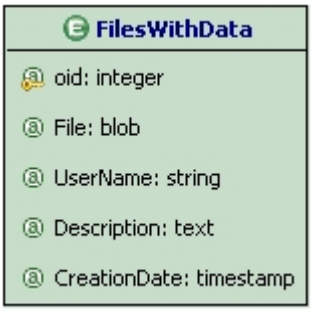

Fig. 2. Exemplary BLOBs subschema

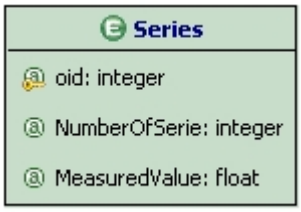

Fig. 3. Exemplary series subschema

Presented entities have to be mapped to one of the supported database types (WebRatio Synchronization Wizard). Depending on deployment method,

resulting SQL code can be immediately executed or saved as SQL script for manual execution. It is also important to: put appropriate JDBC drivers into WebRatio Drivers directory (the name of driver file is important) and set data types sufficient for stored values – especially the max size of BLOB.

3.2. Managing the content of Excel files

WebRatio implements Apache POI framework adding ability of managing Microsoft Office documents. Hypertext model components palette contains Excel Unit that allows to read and write data held in xls files. Developer has to specify the mode and source type of data.

Hypertext model presented in the Figure 4 consists of one page containing form (Entry Unit) for uploading xls files with measurement results and Power Index Unit displaying uploaded data. Excel Unit (called *Open Excel File*) is configured to read data from pointed file, more precisely from the 1st sheet, column A, rows starting from 1 (properties set in Cell Info subcomponent of the unit). *Add Data Serie* Unit saves values from the file to the database table called Series. It is also possible to delete data from the Series table – Query Unit called *Delete content of series* provides oids of records to be deleted by the *Delete Series* Unit.

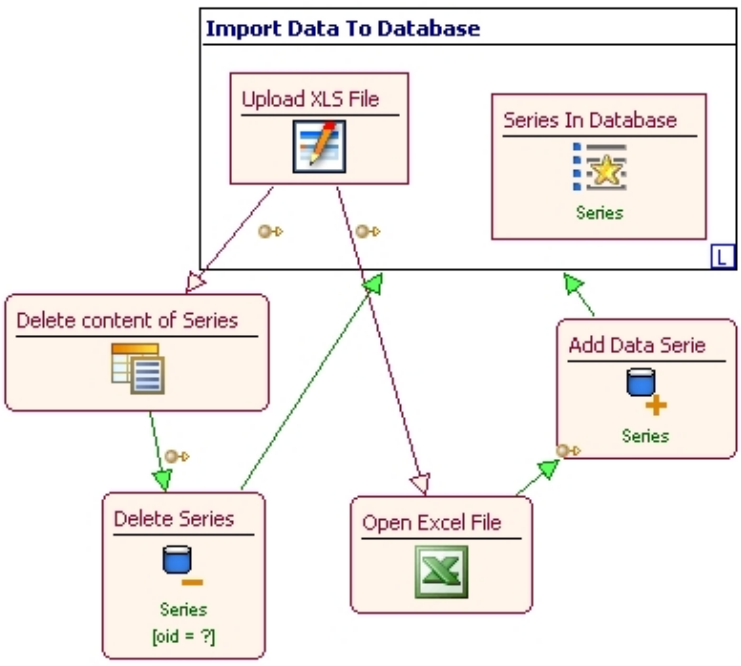

Fig. 4. Hypertext model of a page managing data from xls file

3.3. Exporting data from database

Data stored in database or results of its computation can be easily exported by applying special predefined Excel layout template. The first step is to create page presenting some data in a specific way (name of the page shouldn't contain spaces). The next step is setting Layout → Style page attribute as Excel.

Figure 5 presents hypertext model of *Export Data From Database* page which allows to filter some of stored measurement results taking into account id of serie and bottom limit of the value. The *Results* unit presents results that meet the criteria. The same kind of data is presented by the *Values* page applying Excel style. As a result of clicking Download link, save dialog will appear. Concluding, content of the *Values* page won't be displayed by a web browser, it will be downloaded as xls file.

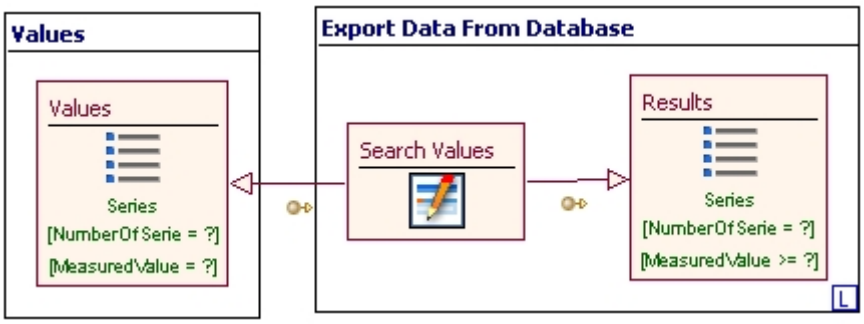

Fig. 5. Hypertext model exporting data as xls file

3.4. Managing BLOBs

Sometimes the number of measurement results exceeds limitations of the xls file, or the description of experiment, videos presenting the experiment, devices technical data, documentary photos etc. should be published. It is not possible to store or publish such data using Excel Unit. It has to be done by uploading data using Entry Unit (can be validated) and storing it as BLOBs inside database table or as files in domain of application. Storing files in database makes them protected by the database safety mechanisms, but it also slows it down. Thus significantly large objects should be kept as files in application domain.

Managing blobs involves two WebML models (data and hypertext model) and following activities:

1. Creation of entity holding BLOB value.
2. Setting storage type – how to store. When database is chosen there is no need to set upload path or the policies.
3. Choosing upload policy – what to do, when the same file name as uploaded one already exists.
4. Choosing delete policy – what to do, when the record holding link to file is deleted.
5. Choosing upload path – where to save the file.
6. Creating hypertext model performing operations on stored files. BLOBs (besides images) are presented by content units as links to files. BLOBs should be treated as regular entity attributes.

Figure 6 presents model of page for performing CRUD operations on BLOBs. *Upload File* Entry Unit allows saving files in database. It also registers the time of uploading (provided by the *Current Time* unit) and the data of uploader (provided by *Get User Data* basing on UserCtxParam value – assuming that only registered and logged in users can upload files). *Uploaded Files* Power Index Unit presents the list of uploaded files (including date, uploader name and description of file) and allows their deletion.

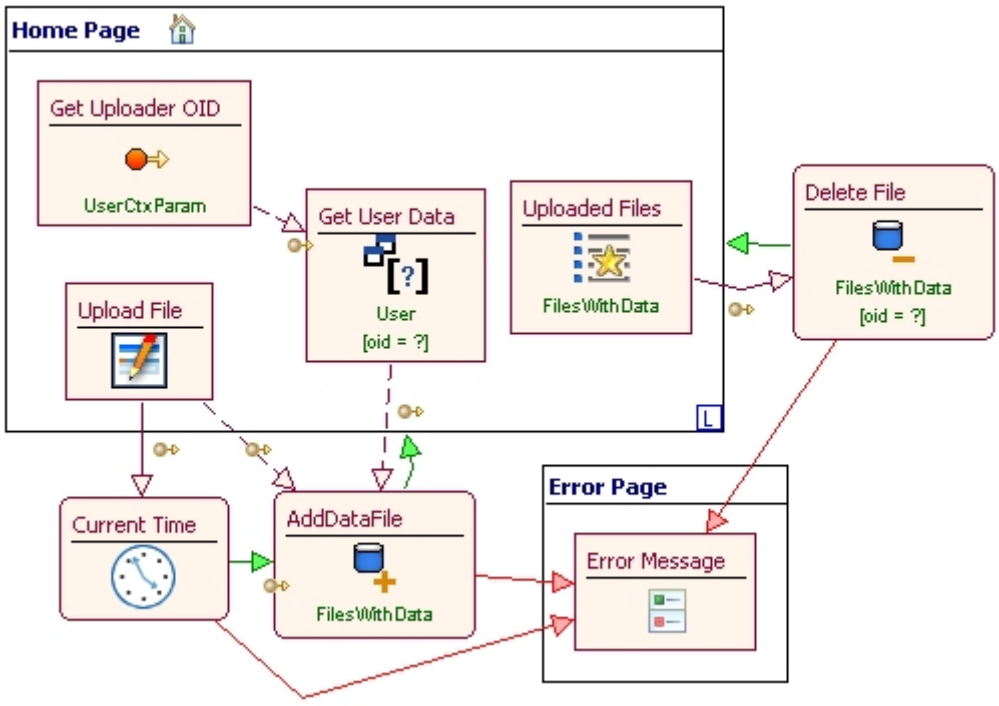

Fig. 6. Hypertext for managing Binary Large Objects

3.5. Limiting access to the data

The access management is both complicated and easy task. One has to first grasp the idea of this management, used in WebRatio. It bases on the personalization subschema, presented in the Figure 1. The module entity is a list of any areas, pages or even single units, access to which has to be somehow restricted. The group entity can store definitions of user groups, having access to specific sets of modules. Finally the user entity is a list of users, which can belong to one or many groups. The group has two connections to module, the 1:N one describes the groups default module while the N:N connection describes the other protected modules, accessible by this group. The group has the access to all its protected modules at the same time. When the logged in user tries to access restricted area, it is checked whether he belongs to any of the groups having the access-rights to that area. User can belong to many groups but can utilize the rights of the only one of them (the current one) at the same time. The user default group is set as current at login time.

The presented subschema is automatically created during project setup. The developer needs only to mark the areas as protected and assure the propagation of that to the module entity. It is done by accessing the special administration web-page, available during the development, and syncing the application current state with the module entity. Creation of groups and users adequate to the application has to be however modeled manually as a part of it. It is not a trivial problem, but some solutions are already made for the beginners. The WebRatio Wiki contains an example of a model for creating groups of users by the admin [5]. It can be easily implemented in any application. Users can be created in the same manner or the more sophisticated solution, including user registration, can be modeled.

A simplified version of it has been shown on the Figure 7. The *Administration* protected area enables creation of groups and users. The *Add group* entry unit allows creation of a group connected to the units chosen from the list provided by the *Modules* Selector unit. The group creation operation is done in a sequence by the *Add group* create unit followed by the *Connect to modules* Connect unit. User creation is done in the same way.

The unprotected *Open area* contains *Login* entry unit which activates login unit, thus enabling the user to log in. Once logged in the user is moved to his default group's default module and a context parameter *UserCtxParam* is set to his ID. This parameter can be e.g. utilized by the Get unit to sign the stored files as shown on Figure 6.

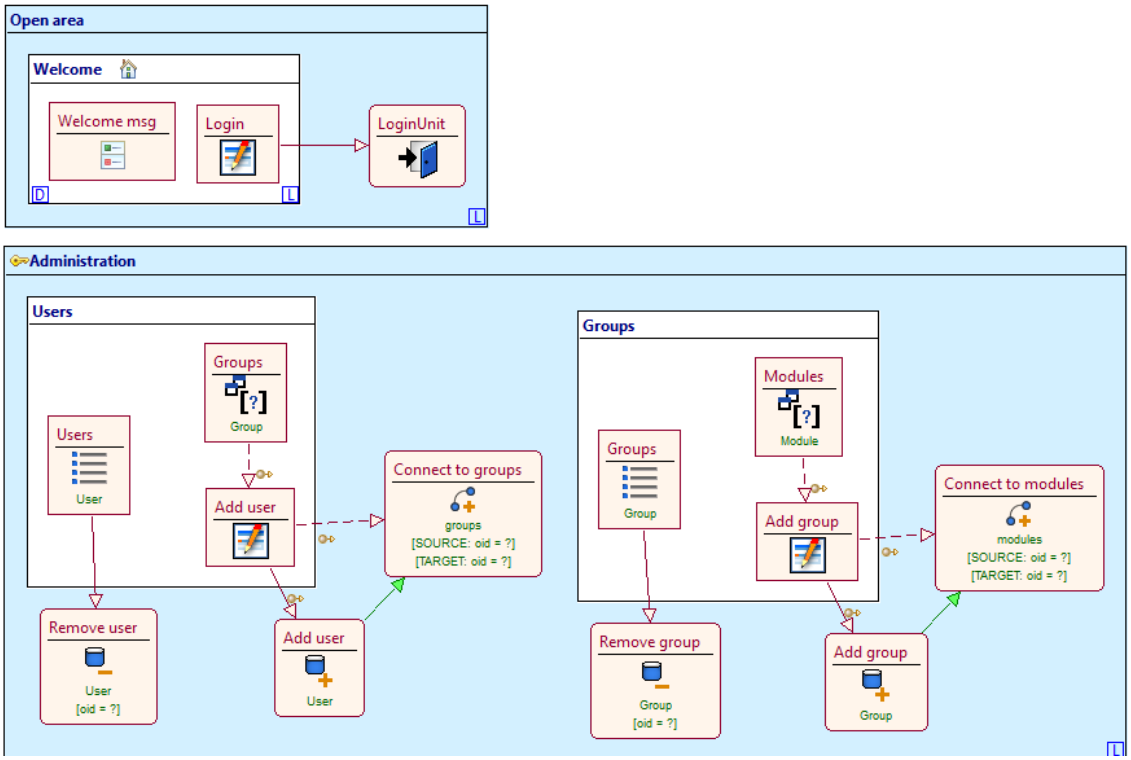

Fig. 7. Example of a simple model managing users and groups

3.6. Setting layout

Developer can use predefined page styles or create its own. Creating the 2nd one is a challenge that requires defining templates for pages, links, units etc. with usage of wr tags and Groovy programming language, so it is advised only for advanced programmers. The 1st one is the simplest one – developer chooses layout from the list of predefined ones (small amount of styles, although sufficient in most of cases). Figure 8 presents the look of the Home Page (model presented in the Fig. 6) with WebRatio style applied. As it can be clearly seen, simple model replaces many lines of complicated code in server-side and client-side programming languages.

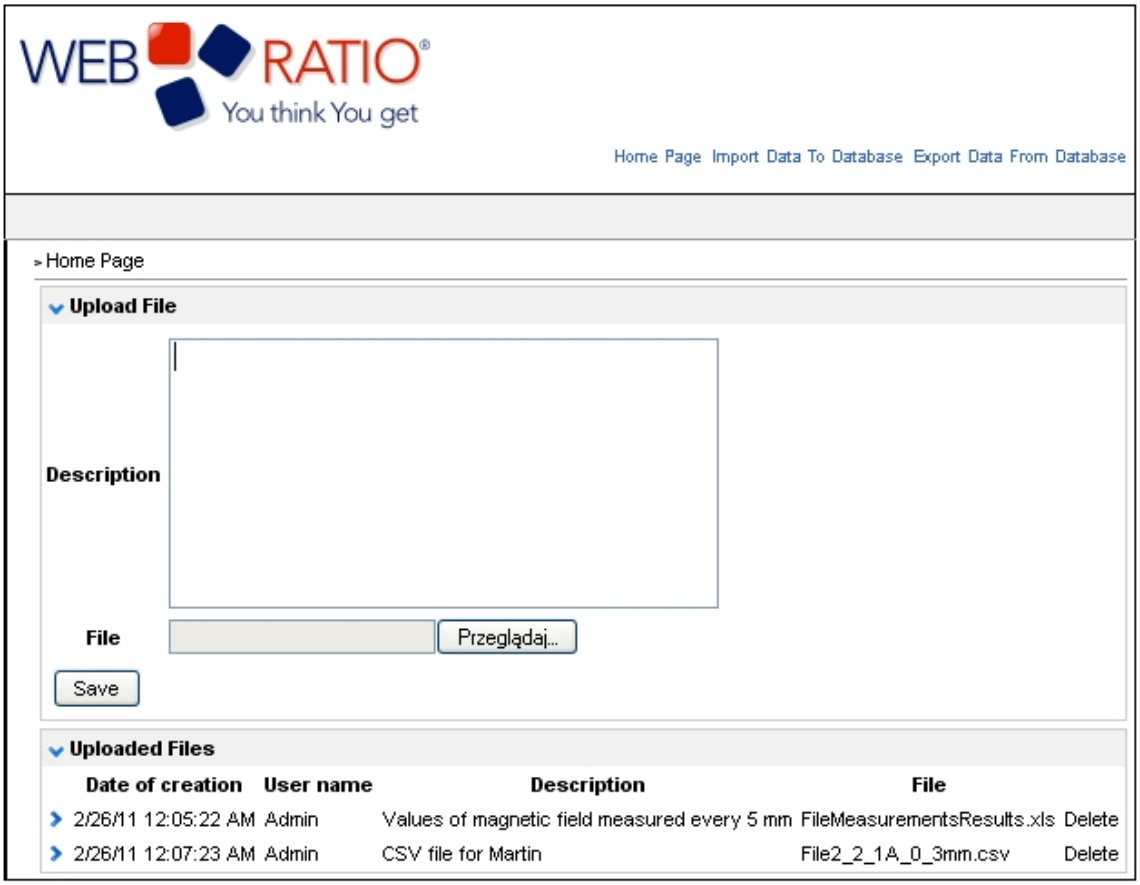

Fig. 8. The look of page for managing BLOBs (WebRatio style applied)

4. DEPLOYMENT OF APPLICATION

The application is generated automatically from the graphical model and can be deployed on any web server capable of running JSP applications (e.g. JBoss or Apache Tomcat). The database can be stored on any common database server type like MySQL, PostgreSQL or Oracle.

While using Webratio the simplest scenarios of application deployment are following:

1. Deployment on local machine using Apache Tomcat web server and Apache Derby database provided with WebRatio installation file (the easiest way to run created application). Developer has to start Apache Tomcat and click on Generate Full Web Project (placed in the WebRatio toolbar). Application home page is available after typing http://host_ _address_or_name/project_name (project name is case sensitive) in a web

browser address bar. Running application can be accessed by other computers from the network as long as host is running.

2. Deployment on remote machine using one of supported databases and application servers e.g. by copying generated files of application into appropriate web server directory and executing manually SQL script (should be adjusted to the particular database – different SQL dialects used by databases) using tools accessible on the remote host. The deployment can be further automatized by enabling creation of war packages or remote generation of the application directly on server [3, 4].

Moreover the newest version of tool provides long-awaited features facilitating debugging the final version of the application while running it by a web server [3]. Also in case of uploading large files (session may expire) or working with complex application (web server may reach the limit of assigned memory) additional configuration of web server may be needed.

5. SUMMARY

The main advantage of the presented MDE approach is the possibility to quickly develop a solution for a distributed team access to the experiment data without the professional web design knowledge. The approach is not perfect for all cases, although can be a quick alternative for using more advanced and expensive equipment capable of providing data via internet.

The other advantage is the possibility to automatize synchronization with periodically updating data by utilizing Jobs feature, made available to model in WebRatio.

One of the disadvantages is a need to create a database structure suited to the each experiment data. On the other hand the hypertext model needs only minor changes in the data acquisition and presentation part, while the rest can be easily copied between experiments. In a case of a set of similar experiments, one can also develop a common database for all the experiments, giving better scope of data access.

The other disadvantage, not directly related to WebML approach, is the limitation of the excel file size. A long experiment can reach the limitation of rows or columns of xls file.

LITERATURE

1. Ceri S., Fraternali P., Bongio A., Brambilla M., Comai S., Matera M.: Designing Data – Intensive Web Applications, Elsevier Science, 2003.
2. Żyła K., Kęsik J.: Usability comparison of WebRatio and symfony for educational purposes, Prace Instytutu Elektrotechniki, zeszyt 247, Warszawa, 2010.
3. WebRatio 6 Help, Web Models, 2011.
4. WebRatio User Guide, Web Models, 2011.
5. WebRatio Wiki, http://wiki.webratio.com, Web Models, 2011.

Manuscript submitted 04.07.2011

PRZECHOWYWANIE I UDOSTĘPNIANIE WYNIKÓW POMIARÓW METROLOGICZNYCH ZA POMOCĄ APLIKACJI WEB WYTWORZONYCH NA PODSTAWIE MODELI WebML

Kamil ŻYŁA, Jacek KĘSIK

STRESZCZENIE *Artykuł przedstawia metodę publikacji i przechowywania danych metrologicznych w Internecie, wykorzystując techniki MDE (Model Driven Engineering). Z pomocą MDE, a konkretnie WebML, naukowcy nie muszą posiadać umiejętności tworzenia relacyjnych baz danych czy oprogramowywania logiki biznesowej aplikacji WWW, aby zbudować i wykorzystywać taką aplikację udostepniającą wyniki pomiarów poprzez Internet w odpowiedni sposób i odpowiedniej grupie współpracowników.*

Jacek KĘSIK, Ph.D. Eng. – adiunkt, Computer Science and Numerical Analysis Department, Institute of Computer Science, Lublin University of Technology.

Kamil ŻYŁA, M.Sc. Eng. – Junior Assistant Professor, Computer Science and Numerical Analysis Department, Institute of Computer Science, Lublin University of Technology.