WODA: A Slim Web Oriented Database

WODA: A Slim Web Oriented Database

TURK, Ziga

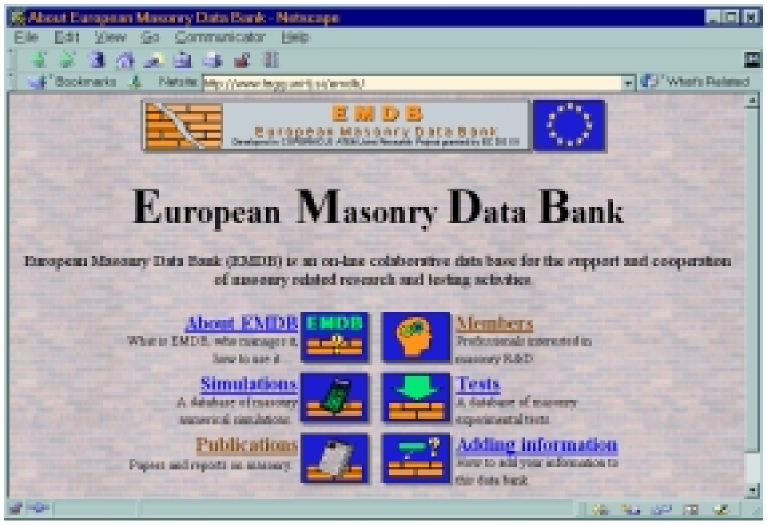
Assoc. Prof., Faculty of Civil and Geodetic Engineering, University of Ljubljana, Slovenia.
E-mail: ziga.turk@fagg.uni-lj.si; Web: www.fagg.uni-lj.si/~zturk/

Abstract

Typical data that is published on the Web fits into a common schema and layout, for example homepages, course information, bibliographies, mailing list archives, documentary data etc. But with the exception of large sites maintained by media companies, databases are not used to deliver such information. Instead, most people use various HTML editors to craft individual Web pages and the file system is used to store them as files. The author claims that the reason for this is that most of the mammoth database systems are overkill - in price, use of system and human resources - to be used for these rather simple tasks. Since 1995, the author has been developing a database system that simplifies the management of multimedia databases - WODA. The acronym stands for Web Oriented Database. Such databases enable the creation, maintenance and use of database systems using a Web browser, and support the conventions of the Web such as integration with Email and the commonly used search syntax. Advanced features of WODA include a software agent and an application generator. WODA is easy to install and databases can be set up in minutes. This, rather than indexing speed, scaling, transaction processing etc. has been the design focus of WODA. WODA development started before the big database players offered useful solutions for the Web. It remains competitive in non-mission critical applications that must be developed rapidly, on a tight budget and where not much time can be spent on maintenance. Most importantly, WODA makes the delivery of Web pages out of a database simple enough for non-commercial web sites and therefore educates towards the proper way of publishing information on the Web.

1. Introduction

The World Wide Web started as a collection of HTML documents. Web documents, as most documents in general, interleave the information content and its layout. While original HTML offered rather poor formatting capabilities the evolution of HTML, until recent work related to XML, went mostly in the direction of supporting more colourful page layouts. Most developers of Web sites, however, quickly realised that as elsewhere in computing, the separation of content and layout could be very beneficial ... it would allow the authors who know about the content, to write that without bothering about the latest HTML extensions. On the other hand, artists could define nice web layouts to follow for different types of information. Big web publishers, such as Ziff Davies, CNN, Cnet etc. were firsts to use databases to store the data to be published on the Web, and special software to present that data in nicely laid out HTML. The databases also allowed for much better security, access privileges and staging of the publication process that html files scattered over the file system.


About European Masonry Data Bank - Netscape
File Edit View Go Communicator Help
Bookmarks Netsite What's Related
EMDB
European Masonry Data Bank

European Masonry Data Bank

European Masonry Data Bank (EMDB) is an on-line collaborative data base for the support and cooperation of masonry related research and testing activities

About EMDB
What is EMDB, who manages it, how to use it

Members
Professionals interested in masonry R&D

Simulations
A database of masonry numerical simulations

Tests
A database of masonry experimental tests

Publications
Papers and reports on masonry

Adding information
How to add your information to this data bank

Figure 1: A typical web site that uses WODA for most of its services.

But the web are not just the big media companies. It is a democratic medium where anyone can publish. But the so-called amateurs (which also include educational institutions, universities, various groups and societies, hobbyists etc.) were given html or site editors, such as Navigator Gold or Front Page. There were used to define more or less static content. A host of freely available CGI scripts was written to handle things like discussion systems, guestbooks, membership lists, collaborative bookmarks etc.

Since 1993 the author has been using the Web to support collaboration in civil engineering (Turk, 1997) and also to provide some services such as the best search engine for shareware at that time (Turk, 1995). Most of the Web services developed were using some CGI scripts and common to them all was handling of structured data; for example a shareware library, ftp sites with shareware, membership lists, home pages of faculty members, published papers including full texts, archives of mailing lists, research project documents, faculty courses, a multimedia collection of slides, questionnaires, conference registrations, paper submissions ... In fact all these services could be viewed as a database problem and the author created a tool to handle them as such - WODA.

If in 1994 concepts like "active server pages" and database systems built right into the HTTPD servers were available, as easily as today, the evolution of WODA would perhaps be different. But at the time writing, a small, lightweight, portable database engine and use it instead of Oracle, Informix or MS Access seemed the most viable solution. First it was called wBASE (web dataBase also revealing authors experience with dBASE). Later it was renamed to WODA (Web Oriented DAtabase). In many Indo-European languages a word like "woda" means water. Name therefore alludes it is caffeine free. WODA is being developed incrementally; new features are being added when there is a need for one in a service being developed. The source code has recently been growing at a rate of about 4 kilobytes a month.

2. Requirements

In this section, the requirements for a *lightweight* web oriented database are listed. The database administrator's (and owner's) requirements:

- The system should be used through the Web using a Web browser.
- Database creator (owner) and administrator don't manage this database as their primary job function. Therefore the creation and management of the database should not take much time.
- Database owner is not a programmer or a database specialist. It should be possible to define and manage the database without programming.
- Database owner is not an artist or an HTML wizard. It should be possible to define the database with limited knowledge of HTML and without the need to hand-code individual screens and forms.
- The system should be open so that the savvy administrators are able to reconfigure or re-program it.
- Desktop applications such as Excel or Access are ubiquitous. It should be possible to move the data from and to desktop applications.
- Applications should be portable. People may want to develop them on a PC and then move them onto a UNIX, NT or whatever platform they rent server space on.
- The data should be stored in a robust and transparent way so that in case of problem tools like vi or notepad could be used examine the problem.
- Users might come from different countries; multilingual features are needed.

End users might enjoy the following:

- The ratio between number of users contributing information and the number of records will be big. Therefore user authentication, record locking and record ownership should be simplified, yet making sure that ones data is not overwritten by someone else's.
- The generated HTML or any other code (Java, JavaScript) should portable and run on as many browsers as possible. Minimal requirement are 2nd generation browsers with tables support.
- Pages should display quickly, look reasonable and include as much information as possible.
- The user interface should be consistent with the common Web style.
- The search syntax must be simple for novices but advanced users must be able to define complex search expressions.
- Data types common to the Web should be supported, such as text areas, radio buttons, URLs, email addresses ...
- It should be possible to store multimedia data without size limitations.
- It should be possible to distinguish between novices and experiences users.
- It would be nice if an agent would be built right into the database and allow users to order customised database updates by email.
- It would be nice if the system would be kind of smart in remembering what a user has been doing.

The following has not been perceived as a priority but did nevertheless receive attention:

- Speed and scalability. Users on the Web are expected to wait a second or two for a reply from a server. The system should respond within that time frame. Services to be developed with WODA were not expected to have millions of hits per day, although some using very same design (Turk, 1995) did and could handle them. Speed degradation when using a large database should be close to linear.
- Security. The system will not be used to handle financial transactions.
- Relations. Although the support for relational databases would be nice, the HTML interface imposes some restrictions on how relational data can be handled. HTML and JavaScript simply do not have the flexibility of a programming language like Java. It is particularly difficult to create form/sub-form type of user interfaces.
- Commerce. Users don't want to see as many ads as possible, but information.

3. Architectural foundation

3.1 Tools and environment

Portability called for a portable programming language. Perl (Wall et al., 1996) was selected for the following reasons:

- All author's CGI scripts at the time were written in Perl;
- Good support for printing text, most notably the block quoting mechanism and variable substitution within strings;
- Powerful regular expressions;
- Perl is interpreted and allows for fast code-test-correct cycles;
- The 'unless' statement and postfix ifs are nice syntactic sugar.

The system interfaces with the Web using a CGI interface. This interface was the only standard way to deliver dynamic content over the web and is supported by all web server software. The main problem of using Perl with he CGI interface is a rather large overhead - time to open another process, start Perl and compile the code. This needs to be done with every request. Recent versions of WODA are compatible with other httpd interfacing mechanisms and will be discussed in section "Implementation issues".

WODA was conceived as a library of CGI functions. Based on the data definition set up by the owner of the database, and parameters passed as a part of the URL, WODA generates appropriate Web pages. In the next sections the supported data models, data storage and the data definition language are presented.

3.2 Data model

In the data models used for today's database applications we can distinguish between tabular data models and object oriented models. The links between records or objects can be set up using relational paradigm (in RDBMS), pointers (in OODBMS) and hyperlinks (in Web oriented databases). Data in a database can belong to simple or multimedia data types. Having in mind the applications listed in section "Introduction" we decided to organise the data in tables, use hyperlinking to relate data in several tables and allow for multimedia data types. More proper relational features were added later.

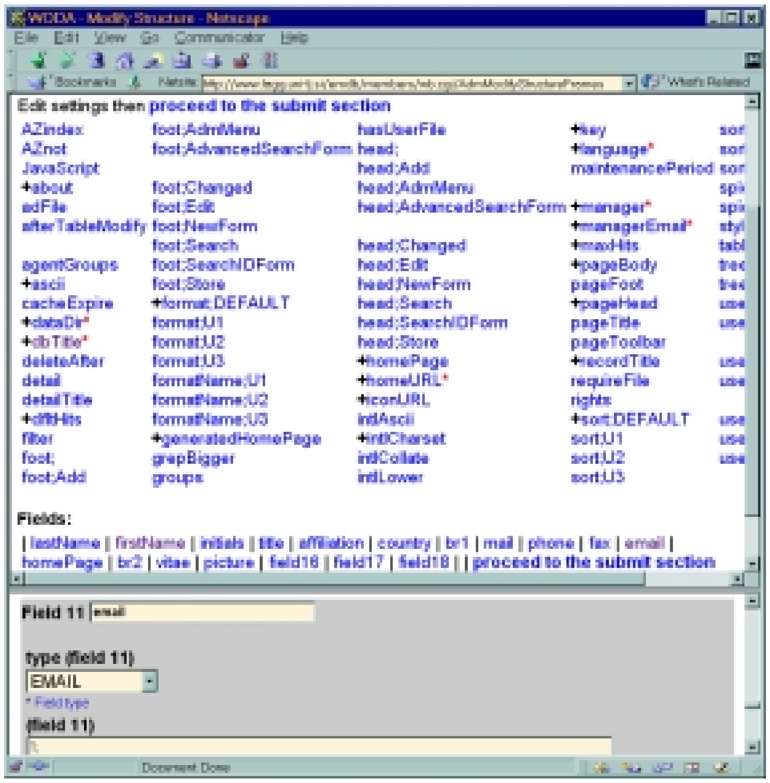
WODA was designed to handle one table at a time. There are several features, however, that allow for establishing relations between tables. A table can contain an arbitrary number of records. Each record can contain any number of fields. Fields may contain textual (in any 8 bit code-page) or binary information. The textual fields of a record are primarily stored as files, each record in a separate file. All records of one table share the same directory that should be reserved exclusively for this table. Binary fields such as pictures, sounds, documents ... are stored each in a separate file within that same directory. The format of the .rec files is very simple, readable and efficient. It starts with \n character (one blank line) followed by a list of name value pairs like:

```
name
Audrey
lastName
Horne
address
Twin Peaks ...
```

These items are written in an ASCII file, one item per line. This format may seem quite inefficient if one thinks of storing little information per record. However, if one thinks of a record storing a whole article, discussion, email etc, record per file makes more sense than record per line.

The name of the file is the same as the primary key of the record. Newlines are escaped. Such organisation is convenient for smaller database of a few hundred records. The efficiency of larger databases, particularly when searching, is vastly improved by caching some information. Cache files are created from the .rec files periodically, either after every update of the database or after every few hours or as a result of a scheduled job. Several such files will be created, pre-sorted to the defined sort expressions. WODA also prepares other files that speed up the generation of some pages, for example the A-Z indexes or hierarchical menus.

Except for the primary key, so far there has not been a need for additional keys. Searching is done using brute force. In use patterns of typical WODA databases it has proven superior to some other approaches. Searching is typically done over all fields. Full text indexes are not pre-created. WODA databases tend to grow slower that the speed of the Web servers. Brute force searching relies on Perl's internal grep function or an external grep program. The latter tends to be faster on large databases. Administrators can set the threshold to decide which search method to use. Similar search approach has been used in the Virtual Shareware Library (Turk, 1995) and tests showed that grep outperformed WAIS(99), SWISH(99), GLIMPSE(99) etc. indexing approaches in a database that included 30 megabytes of text. The reasons are in a relatively big start-up overhead of any of the complex indexing schemes. They would perhaps outperform grep if they would operate as demons loaded at all times, but, as pointed out above, WODA applications do not attract huge numbers of visitors a day, keeping a demon alive would just eat up system resource and installing demons on rented servers is beyond what most users can hope for.


WODA - Modify Structure - Netscape
File Edit View Go Communicator Help
Bookmarks
Netsite
What's Related
Document Done

Edit settings then proceed to the submit section

AZindex	foot;AdmMenu	hasUserFile	+key	sor
AZnot	foot;AdvancedSearchForm	head;	+language*	sor
JavaScript		head;Add	maintenancePeriod	sor
+about	foot;Changed	head;AdmMenu		spi
adFile	foot;Edit	head;AdvancedSearchForm	+manager*	spi
afterTableModify	foot;NewForm		+managerEmail*	styl
	foot;Search	head;Changed	+maxHits	tabl
agentGroups	foot;SearchIDForm	head;Edit	+pageBody	tre
+ascii	foot;Store	head;NewForm	pageFoot	tre
cacheExpire	+format;DEFAULT	head;Search	+pageHead	use
+dataDir*	format;U1	head;SearchIDForm	pageTitle	use
+dbTitle*	format;U2	head;Store	pageToolbar	
deleteAfter	format;U3	+homePage	+recordTitle	use
detail	formatName;U1	+homeURL*	requireFile	use
detailTitle	formatName;U2	+iconURL	rights	
+dfltHits	formatName;U3	intlAscii	+sort;DEFAULT	use
filter	+generatedHomePage	+intlCharset	sort;U1	use
foot;	grepBigger	intlCollate	sort;U2	use
foot;Add	groups	intlLower	sort;U3	

Fields:

| lastName | firstName | initials | title | affiliation | country | br1 | mail | phone | fax | email | homePage | br2 | vitae | picture | field16 | field17 | field18 | | proceed to the submit section

Field 11 email

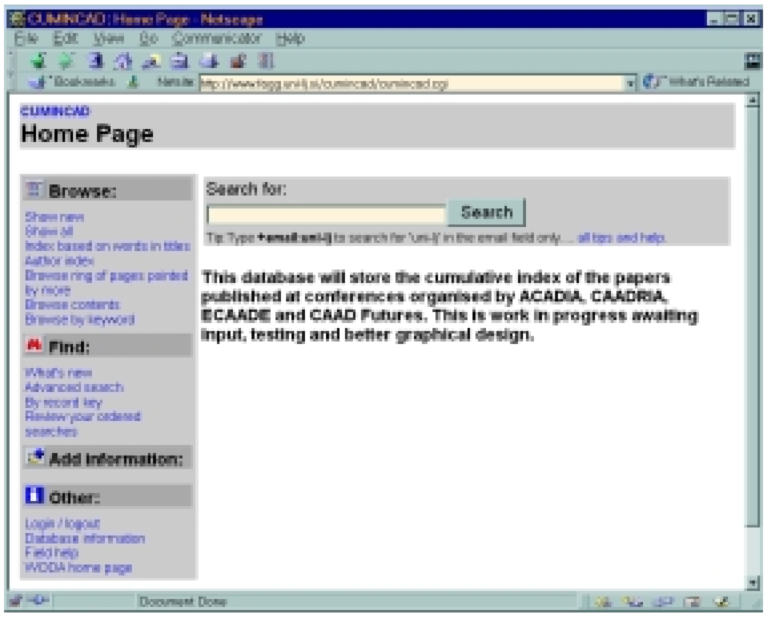
type (field 11)
EMAIL
* Field type

(field 11)

Figure 2: Form based application generator.

3.3 Data definition language

The author had some experiences with database development on PCs and UNIX, using commercial products such as dBASE, FoxBASE and Microsoft Access. What he missed was an option to define as much information about a field as possible in one place and reference that information rather than copy it. On of the main strengths of WODA is a powerful data definition language. Instead of defining a special syntax, the language is Perl itself. Advanced users can use the power of Perl to define the database's structure and other features. To novices the syntax is quite simple. They can also use a form-based interface to define the parameters and generate a ready-to-run web database without coding (Figure 2). This also means that the definition is in transparent text file that can easily be inspected, moved, translated, words can be searched and replaced ... which are all features missing in the visual databases where this information is scattered in millions of forms that must be clicked through.


CUMINCAD : Home Page - Netscape
File Edit View Go Communicator Help
Bookmarks
Netsite: http://www.fagg.uni-lj.si/cumincad/cumincad.cgi
What's Related
Document: Done

CUMINCAD
Home Page

Browse:
Show new
Show all
Index based on words in titles
Author index
Browse ring of pages pointed by mose
Browse contents
Browse by keyword

Find:
What's new
Advanced search
By record key
Review your ordered searches

Add information:

Other:
Login / logout
Database information
Field help
WODA home page

Search for:
Search
Tip: Type **+email:uni-lj** to search for 'uni-lj' in the email field only.... all tips and help.

This database will store the cumulative index of the papers published at conferences organised by ACADIA, CAADRIA, ECAADE and CAAD Futures. This is work in progress awaiting input, testing and better graphical design.

Figure 3: A typical title page of an application generated with WODA's application generator.

In WODA, the owner defines two sets of properties. The first defines features of a table as a whole and its relation to other tables, the second defines the features of individual fields. They are stored in hashes %WBB and %WBF respectively. There are about 90 parameters that define table's features, but only one - the directory to store the database - is mandatory for WODA to work. The only information required for a field is its name. Table 1 lists some of the features related to the table as a whole, Table 2 those related to the field and Table 3 the field types.

Table 1: Some of the table related properties.

property	description
{PRIVATE}**dbTitle**	* database title
about	short descriptive text about the database
recordTitle	title of one record e.g. employee, article, internetResource ...
language	* User interface language. Enter a two letter language code (English=UK, Slovenian=SI ...). You must have the appropriate wbxx.pl libray installed, to support your language.
manager	* name of database's manager (see *managerEmail*)
managerEmail	* *manager*'s email
dataDir	* directory where data of the database is stored (without trailing slash!)
homeURL	* URL which denotes the same directory as the *dataDir*
homePage	Name of a HTML file which is the homepage of the database (in *homeURL*) unless contains a / in which case relative to server root). If empty, the database's home will be the page automatically generated by this script, not an HTML file.
generatedHomePag	WODA can automatically generate a .html file for the database home page.

property	description
e	Define here into which file it should create it (relative to *dataDir*. Can be the same as *homePage* in which case the generated .html will replace what you wrote in it. Mainly used to speed up the delivery of the database's home page.
spiderDir	directory where system generates HTML pages which web robots should find (without trailing slash!). If undefined no such files will be generated. One .htm file per record will be generated plus file toc.htm.
spiderURL	URL which denotes the same directory as the *spiderDir*.
sort;DEFAULT	string to use for DEFAULT sorting of records. If undefined records will be sorted by the first filed. (string expression)
format;DEFAULT	string which formats DEFAULT printout of one record in a listing. If undefined a table will be printed. (string expression)

Table 2: Field attributes.

attribute name	description
{PRIVATE **}type**	* Field type
	* Expression which validates the field value in $_. Keep 1; if anything goes. In BREAK fields text of the break. (logical expression)
cond	Validation expression written in plain English (or another language).
head	Name of the field for table headings. If undefined, field name is used.
p	Short prompt for field
help	Help text on field input. Usually much longer than prompt text.
d	Default value
typePar	Additonal field pramaters.
options	* Allowed options for OPTION typed fields either on a same line separated by | or one per line
into	dataDir or alias of the related database
picture	expression which nicely formats field value in $_ Use &QRY() and &FLD() functions to relate to other databases (string expression)
srt	order of field in forms
sticky	Leave empty (normal) or set to 1 to make the value entered here default next time the same user enters this form. E.g. email addresses could be made sticky. Set to any value including a / to set the path= variable in HTTPD-COOKIE header.
modifies	| separated list of *groups* allowed to modify this field (or empty for all) (regular expression)
sees	| separated list of *groups* allowed to see the value of the field (or empty for all) (regular expression)

Table 3: Field types.

type name	description
{PRIVATE**}BREAK**	A horizontal line to separate fields in forms and tabular output
COMPUTE	A field which is not stored in the database but is computed on the fly when record is read.
DATE	Date type. Internally represented as yyyy/mm/dd.
EMAIL	An email address. By default rendered as mailto: HREF, checked to include '@a'.
FILE	A file somewhere on the server, relative to homeURL. E.g. picture, full text document ... By default rendered as link to that file.
HTMLAREA	A paragraph of text or a whole tagged HTML document. Any formatting should be done in HTML. Adjust size with ROWS=n COLS=m
IMAGE	Like USERFILE but will be rendered on screen as an IMG.
INPUT	The default field type. A line of text.
LINKOPTION	An option list where the options are keys from another table. Define table's alias or dataDir in the 'options' attribute
LIST	A list of values. Values should be entered one per row. Formatting and verification applies one per each item
OPTION	An option list. Define options in the 'options' attribute
TEXTAREA	Several rows of preformatted text. Adjust the size with ROWS=n COLS=m. Will be tagged as <PRE> unless P or BR tags are present
URL	A URL. By default rendered as A HREF
USERFILE	Like FILE but database users can upload it to the server using the Netscape 2+ or Explorer 4+.

4. Implementation issues

4.1 Packaging, availability and installation

Over the years WODA has grown to about 200 kilobytes of Perl code. Although several database and web related Perl packages are freely available, WODA does not rely on any of those. They could save the author some work but make WODA more difficult to install, bigger and dependent on other people's work. Besides the core WODA functions were written before similar were available from CPAN. WODA source code is freely available under GNU and Artistic licences.

There are several ways in which WODA can be installed but regardless of the installation the applications can stay the same. The main problem with the Perl CGI programs is the relatively large overhead caused by (1) starting Perl, and (2) compiling WODA and the application into Perl's intermediate code. In all tasks except in searches these overheads take most of the time to service a request. The actual processing of the request is rather quick. The problem of this overhead can be tackled in several ways:

- Use of Perl 4 instead of Perl 5. Perl 5 may run programs faster, but takes much longer to start and to compile.
- Use of compiled Perl programs. Perl compiler available in Perl 5.005 is not stable enough to compile WODA, but on systems that support the undump utility, WODA can undump itself to a binary program.
- Compile times can be dramatically reduced if not the entire WODA code is used (and compiled) in each request but only the functions actually needed. WODA is therefore distributed in two ways - in one big file (the 'big' WODA) and in several smaller ones (the 'spit' WODA). Each of the small files only includes the code needed to process one type of request for example a search or an edit.
- Use of ModPerl. ModPerl is a module for Apache Web servers that lets users add Perl programs directly to httpd server. In this case WODA becomes part of the httpd server; it is always there, precompiled, ready for applications to call its functions. This way has been found to be the most efficient way to deploy heavily used WODA applications.

Table 4 summarises the relative speed differences of the various installations of WODA relative to the author's typical setup. The 'big' WODA is trivial to install - only one file must be copied to a Web server. In the case of 'split' WODA many files must be coped to the right directory.

Table 4: Times to generate a simple page using WODA (relative, smaller means faster).
An .html file has an index of 0.08.

{PRIVATE**}**WODA distribution / Perl	Perl 4.036	Perl 5.005
big WODA	2	3.2
split WODA	1	1.3
undumped big WODA	0.22	-
MOD_PERL and big WODA	-	0.12

4.2 Customisation

Contrary to some other databases for the Web, the philosophy of WODA is that it generates all the pages an application needs. Owners of the database can modify how the pages look and what they include by changing the %WBB parameters. They can't insert calls to WODA in the middle of a .html file. If they don't like what WODA does they can replace WODA's handlers by their own. There are several ways to customise WODA and the applications:

- By configuring the few parameters in the source code of the WODA engine. These parameters *must* be configured once per web server.
- By setting values to many attributes that define the database and the database fields. This can be done through the form-based interface in the Administration menu of any WODA application.
- By overloading cgi* functions defined in the WODA engine with custom functions. The mechanism for doing so does not require changes to the WODA source code. Pages generated by a database applications will have a URLs like `http://some.server.com/cgi-bin/yourscript/Search`, where "script" is the name of the cgi script and "Search" is the command the script should perform. The command Search is performed by a subroutine `cgiSearch`. If owners don't like what the `cgiSearch`

does, they copy it into file `yourscript`, rename it into `mySearch`, change whatever they need and the my* routine will always be used instead the cgi* routine.

- By translating all system messages printed by WODA. This is mainly used to translate WODA engine into other languages. So far WODA has been translated into Slovenian, German, English, Italian, Swedish, French, Russian ... A special program extracts all parts of the source code that are language specific into a special file. The lines are then translated and another program merges the translation with the English version.

4.3 Relational features - linking tables

Relational features are at present limited to:

- Setting items of selection fields to keys of another WODA table. Specify the field as *LINKOPTION* type and in the *INTO* field define database alias or path to its *dataDir*.
- Specifying a query into another WODA database as a picture of a field. Function &*QRY ($table, $search, $then, $sort, $format, $first, $max)* can be used in all data definition fields where string expressions are allowed.
- Getting a value of a field from another WODA database. Function *&FLD ($table, $id, $fieldName)* can be used in all data definition fields where string expressions are allowed.
- Getting raw data from another table in TAB format. Function *&ROWS($table,$flds,$search,$then,$sort,$format,$first,$max)* returs rows of data from another table. Only the fields listed in *flds* attribute (comma separated) will be returned.

These features, however sufficed to implement a Web site for our faculty in Woda. The schema is shown in Figure 4.

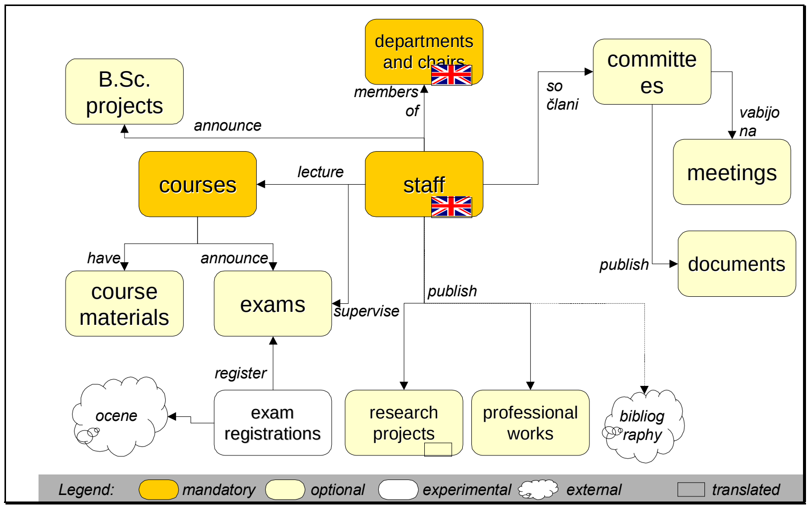


Figure 4: Faculty's web site implemented as a set of related WODA tables.

4.4 Security and Privileges

WODA distinguishes between users and groups of users. Access to individual actions within WODA is based on a group into which a user belongs. Three levels of security can be assigned to a WODA table:

1. Minimal; groups only. No groups are defined; groups *admin* and *guest* are created by default and are used. Admin can do anything. Guest can do anything except access the administration menu or order agent searches.
2. Custom; groups only. Other groups are defined and access rights of the groups are defined. These access rights allow or deny user actions with WODA based on the action's URL.
3. Custom; groups and users. A separate WODA table for user information is set up.

It is also possible to prevent groups of users to see or modify some fields. Users are authenticated using Web server's login mechanisms, by IP number, IP address or using WODA's internal login mechanism.

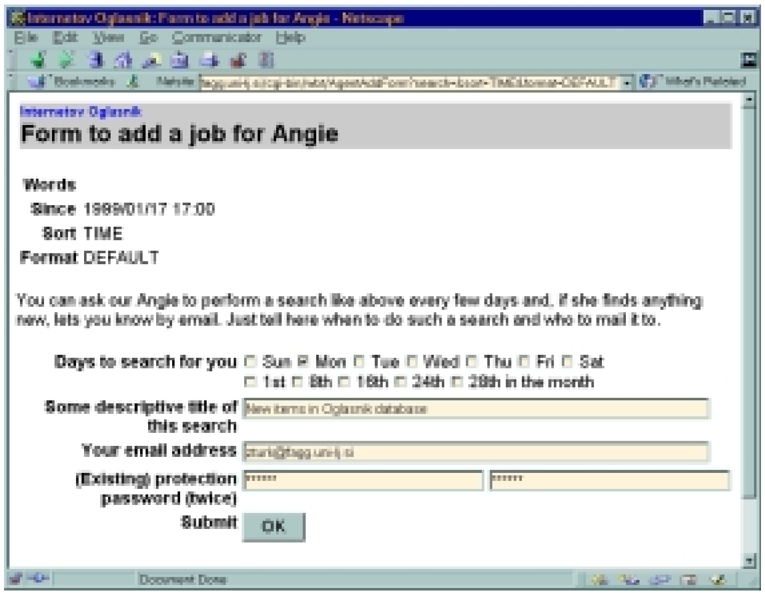
4.5 Browser compatibility

The author has been rather conservative in using the latest HTML extensions and script languages, however, some of these are really useful. Cookies are used to determine, when it was the last time someone used the database so that news can be displayed. Cookies are also used to store the values of sticky fields. For example user's email address is often a required field in databases - and it is stored as a cookie so that it is suggested as default on subsequent uses. JavaScript is used to validate the fields on the client side. Unfortunately Perl's and JavaScript regular expressions are not 100% compatible. Recently style-sheets are used to determine the look of the generated tables. Unfortunately, the support in Netscape 4.5 is sluggish.

4.6 Some other features

- 'Best-match' searching does a full text search of the database and provides relevancy ratings. Search syntax is similar to AltaVista's.
- A-Z indexes, rings of Web pages, keyword browsing, pages to write serial emails can be easily enabled.
- Complex searches can be assembled using friendly interface similar to other query building applications (e.g. MS Query).
- Each record can be individually password protected by the person who entered it.
- Database administration is web based - the administrator may delete outdated records, generate static .html pages and even change the database definition ... all from within a browser.
- WODA is smart. It remembers user's options, when he last searched for something (so that we can show only changes since) etc. WODA includes an agent that can do searches for the user and remind him of changes periodically.

- WODA databases are open and can export/import data from Windows applications such as Microsoft Excel or Access.
- WODA includes an agent that can be ordered to do searches on behalf of the user periodically and forward the results in text and HTML formats over the email.
- WODA can nearly-automatically generate a snapshot of a database that can be used from a CD-ROM. To do so it generates several static HTML files and replaces WODA's search routine with a JavaScript subset. This allows owners of a database to offer it on a CD ROM for which no installation or special software is required, only a Web browser.


Internetov Oglasnik: Form to add a job for Angie - Netscape
File Edit View Go Communicator Help
Bookmarks
Netsite:
What's Related
Document Done

Internetov Oglasnik

Form to add a job for Angie

Words
Since 1999/01/17 17:00
Sort TIME
Format DEFAULT

You can ask our Angie to perform a search like above every few days and, if she finds anything new, lets you know by email. Just tell here when to do such a search and who to mail it to.

Days to search for you	☐ Sun [✓ Mon] ☐ Tue ☐ Wed ☐ Thu ☐ Fri ☐ Sat ☐ 1st ☐ 8th ☐ 16th ☐ 24th ☐ 28th in the month
Some descriptive title of this search	New items in Oglasnik database
Your email address	pturk@fagg.uni-lj.si
(Existing) protection password (twice)	****** ******
Submit	OK

Figure 5: Built-in agent. At the end of each search screen a link to a form like this can be enabled**.**

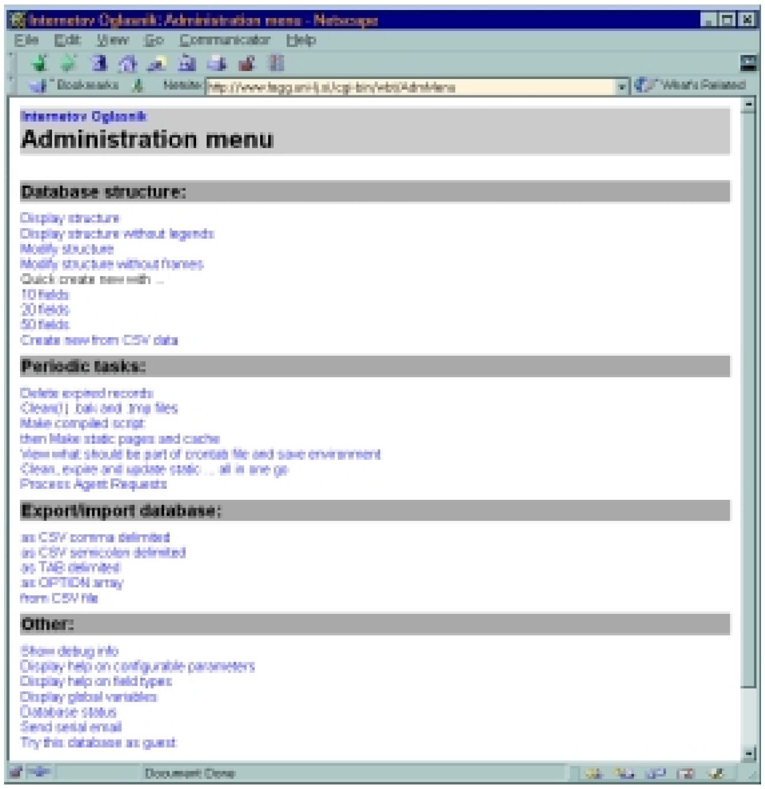
4.7 Web orientation

Several commercial and free database systems outperform and out-feature WODA in traditional database characteristics. But WODA has been developed, from scratch, as a Web-oriented system. A Web oriented database system is a database system that was designed to allow the definition, application generation, maintenance and use of the database over the Web. Resulting database application adheres to Web's conventions and standards. It looks and feels like other Web applications and it requires a little more than a Web browser to be maintained and used. WODA's Web oriented features include:

- 99% of the creation and maintenance work can be done through the browser. The remaining 1% is not done through the Web for security reasons.
- It supports Web data types such as URLs and email addresses. It can send serial emails to addresses in a table.
- It supports typical Web applications such as rings of pages, mailing list archives, on-line

discussions, guestbooks ...

- It support's Web style of searching. Users simply type in words. No syntax to learn.
- It support's the creation of collaborative Web databases without putting users through the choirs of signing up, registering etc. But still lets them keep a lock on their data.
- WODA's agent (Figure 5) can be customised to push data and deliver channels onto user's desktop.
- WODA can be configured to generate static pages of information in a database so that Web crawlers can find and index them.


Internetov Oglasnik: Administration menu - Netscape
File Edit View Go Communicator Help
Bookmarks
Netsite:
What's Related
Document: Done

Internetov Oglasnik

Administration menu

Database structure:

Display structure
Display structure without legends
Modify structure
Modify structure without frames
Quick create new with ...
10 fields
20 fields
50 fields
Create new from CSV data

Periodic tasks:

Delete expired records
Clean(!) .bak and .tmp files
Make compiled script
then Make static pages and cache
View what should be part of crontab file and save environment
Clean, expire and update static ... all in one go
Process Agent Requests

Export/import database:

as CSV comma delimited
as CSV semicolon delimited
as TAB delimited
as OPTION array
from CSV file

Other:

Show debug info
Display help on configurable parameters
Display help on field types
Display global variables
Database status
Send serial email
Try this database as guest

Figure 6: The administration menu.

5. Conclusions

WODA is a practical, free, Web oriented database system and application generator. It allows relatively inexperienced users to quickly set up a flat database on the Web. If they learn a little bit of Perl, relational databases are possible as well. WODA is not as fast as the mammoth databases and lacks a whole lot of database features, but is fast enough for most non-professional Web applications. To users fluent in Perl, it is extremely flexible; if used under ModPerl and if cacheing of searches is enabled, it becomes very fast. WODA cannot compete with the professional databases in mission critical applications like large internet stores, payroll or inventory systems, on-line banking etc. On the other hand it is much more practical for smaller tasks that are needed to make Web sites easier to maintain and use.

WODA evolves according to the requirements of its user community. It is used internally at several universities and academies, the NASA, Siemens, Amazon.com's subsidiary ... The following extensions are being considered:

- Optimise searches in ModPerl installation by pre-loading indexes and portions of the database.
- Enable the use of an SQL database as the underlying database engine.
- Allow users to do add code that would do something to the listed records.

WODA and more information about it are available at http://www.fagg.uni-lj.si/woda/.

6. References

GLIMPSE (1999). http://glimpse.cs.arizona.edu/index.html

SWISH. (1999). http://www.best.com/~pjl/software/swish/

Turk, Z. (1995). Virtual Shareware Library - A WWW based system for cataloguing software on theInternet, R. Holzapfel (editor), Poster proceedings, Third International WWW Conference Darmstadt'95, Frauenhofer Institute for Computer Graphics, Darmstadt, Germany, 13-16.

Turk, Z. (1997). Publishing Construction Information on the Web, Structural Engineering International, ISSN 1016-8664, Vol. 7., No. 4, 292-296.

WAIS (1999). http://vinca.cnidr.org/software/Isite/Isite.html

Wall, L., Christiansen, T., Schwartz, R.L. (1996). Programming Perl (2nd Edition) , O'Reilly & Associates; ISBN: 1565921496.