1 Abstract

WDB is a software tool-set that simplifies a lot the integration of Sybase databases into the Worldwide Web. WDB lets you provide WWW access to the contents of Sybase databases without writing a single line of code !

In that respect, WDB is the logical evolution of the ESO/ST-ECF/CADC STARCAT database interface towards the graphical user interface world and multi-platform availability. The multi-media aspect of the WWW is also well suited for a migration of STARCAT functionality to the Web (PreView, multi-database server access,...).

The interface between the WWW server and the database only consists of the WDB script written in Perl (augmented with Sybase access facilities) and a set of high-level *form definition files (fdf)*, each describing a different view on the database. WDB first creates HTML query forms from the content of the *fdf* . Then, with user's search constraints, queries the database and finally presents the results back in a new HTML page. WDB comes with a utility to automatically extract information about a table from the database and create a working template *fdf* .

A number of conversions are possible on the data coming from the database before they are shown to the user (e.g., coordinates into hours minutes and seconds; dates, etc.) The unique feature is the possibility to convert data from the database into hypertext links. These hypertext links can address other elements in the database - so providing a simple way of navigating between related information. They could also be links to other documents on the Web - therefore providing easy integration between data in the database and related documents on the Web. Or it could be links to other databases with a WDB or similar interface - hence providing a simple mechanism for cross-database links.

The WDB script and various documentation are available on WWW at `http://arch-http.hq.eso.org/bfrasmus/wdb/wdb.html`

2 Architecture

2.1 A Normal Database Connection.

First lets look at a typical database application layout :

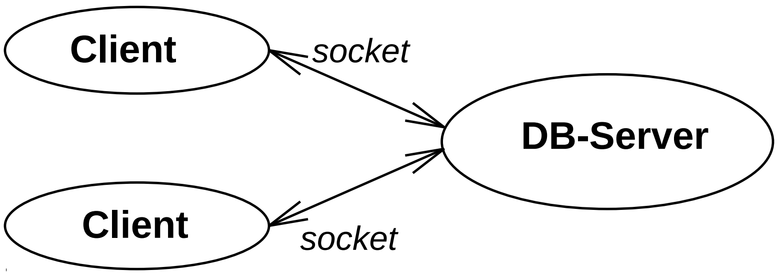

Figure 1: A Normal Database Connection.

Clients connect to a database server, typically using a proprietary library. The client is normally a special purpose application that has to be downloaded and installed at the end-users site. Connections are usually of a long lived nature : The client connects, makes several queries, then disconnects. The server therefore has some notion of what state the client is in (A "current" query, how many rows have been retrieved, which row are the next to be send to the client etc.)

2.2 A Normal Web Connection

Next, we need to understand a "standard" World-Wide Web connection. World Wide Web consists of an http "daemon" which accepts socket connections on a single port, reads a URL query from the client thus connected, outputs the data associated with the URL specified by the user, and closes the socket.

The "standard" World-Wide Web configuration is something like:

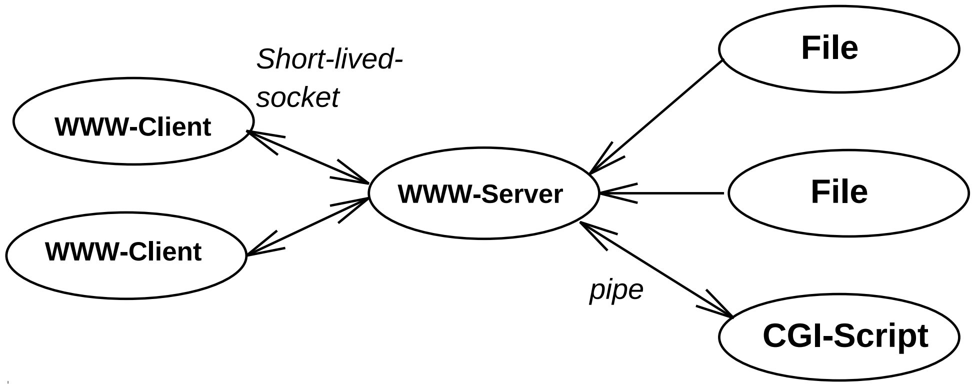

Figure 2: A Normal Web Connection

Here the connection is using a standard (and publicly available) protocol. The clients are freely available, and normally already installed at the end-users site.

Besides static information in files, most web clients and servers now support fill-out-forms. The contents of these can, once filled out by the users, be send to a CGI script (Common Gateway Interface) which can process this data and send the result back to the user. This is the mechanism used by WDB to allow the users to query the databases.

One obvious difference between a database connection and a WWW connection is the length of time the connection lasts – In the case of a database connection, it can be up to several hours; in the case of WWW the connection is often less than a second in duration, or more precisely, the time it takes to process one query.

2.3 A WDB Connection.

WDB is implemented as a CGI script which can generate forms and send them to the user, retrieve the contents of the form, process it and return the result to the user.

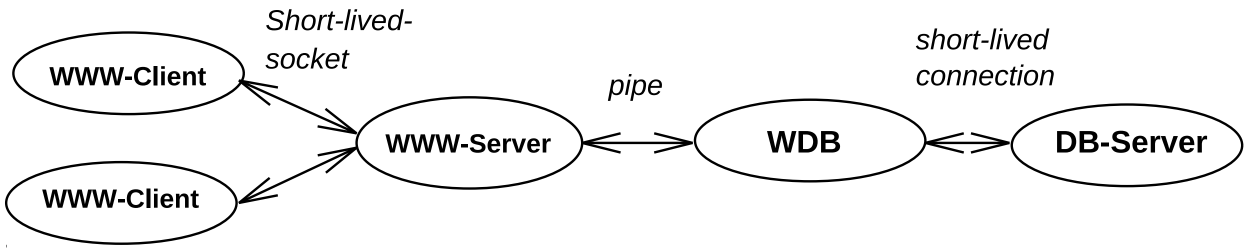

Figure 3: A WDB Connection.

Because of the different nature of a WWW connection the WDB system has to be designed differently from a normal database application. There are no long-lived connections, the server has no notion of what has been going on earlier in a "session". The WDB process is exited immediately after the results are send back to the user. This means that all information needed to perform a given action has to be send from the client each time. Of course the client will have this information from the last results it received from the server.

The way this information is send back and forth between the WWW client and server is via the URL's. URL stands for "Uniform Resource Locator". It is a standard for specifying a piece of information on the Internet. The URL is composed of a protocol name (ftp, http, gopher) the name of the server, and a path on that server pointing to the piece of information need. This can be a file or, on a http server, a CGI script such as WDB. Here the trick is that if there is extra information in the URL after the path, then this is passed to the script as what is called PATHINFO.
(ex. `http://arch-http.hq.eso.org/cgi-bin/wdb/`*`PATHINFO`*)

2.3.1 Flow of Control in WDB.

The actual flow of control is perhaps better illustrated in the following diagram :

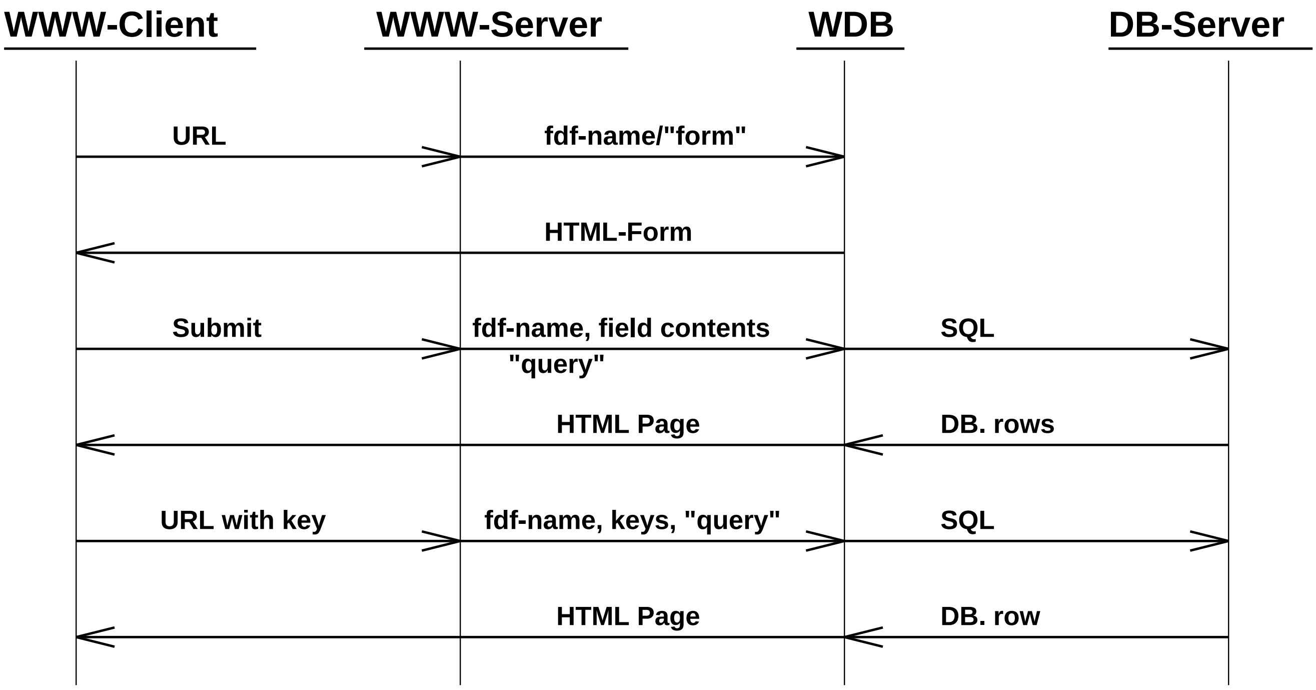

Figure 4: Flow of Control

What happens is the following

- The users "clicks" on a URL leading to the WDB script. The URL contains some extra information in the path (PATHINFO), specifying what information in the database s|he would like to query. This is typically obtained from a normal hypertext page with a menu of available query forms.
- The server receives the URL, starts the WDB script and passes the extra information in PATHINFO, actually the name of the *Form Definition File (fdf)* to use and the word "form".
- WDB reads the *fdf* and generates an HTML form according to the specifications in the *fdf* . The form is returned to the user and WDB exits.

- The user fills out the form with her query constraints and presses the "Submit" button. This sends the WDB URL (with PATHINFO) plus the contents of the form (encoded) to the WWW Server.
- The server receives the URL, starts the WDB script and passes the extra information in PATHINFO plus the form content.
- WDB parses the PATHINFO which in this case is the name of the *fdf* plus the word "query". This tells WDB to construct a SQL query from the information in the *fdf* file and and the user's constraints.
- The SQL query is send to the database server.
- The database server processes the query and returns a number of rows.
- WDB retrieves the rows from the database server and after having converted the data according to the specifications in the *fdf* , constructs an HTML page with the result. If more than one row was returned, a tabular output format is chosen, allowing the user to select a row, via a hypertext link, to get all the details.
- The HTML page is returned to the user and WDB exits.
- The user clicks on a hypertext link for a row. This link contains the WDB URL with the *fdf* name, the word "query" and the key(s) for the current row in the PATHINFO.
- The WWW server receives the URL, starts the WDB script and passes it the extra information in PATHINFO.
- WDB constructs a new database query, this time using the key(s) instead of the form contents.
- The database process the query and returns the result.
- WDB retrieves the row from the database server, converts the data according to the specifications in the *fdf* and this time, as only one row is returned, constructs a HTML page with all the details. The HTML page is returned to the user and WDB exits.
- etc.

3 The Form Definition Files.

One of the goals of WDB was to provide an easy - yet flexible - way to make database information available. In the simple case no coding should be necessary. However if special behavior is needed it should be possible to add it - **without changing the core system**. Also if some special behavior has been defined for one field in a form it should be easy to reuse this in other forms.

As examples of such special behavior can be mentioned :

- Conversion of LaTeX special characters in a text field to the equivalent HTML representation.
- Conversion of coordinates stored in the database as milliarc seconds to hours minutes seconds format.
- Conversion and formatting of dates.
- Computed fields with a value expressed as a function of the values of the other fields in the form.
- Upper-/lower-case conversion of text fields.
- Inclusion of in-lined images.
- Adding hypertext links to fields. Possibly pointing to other WDB objects.

(Some of these conversions can be implemented using database views, however this usually makes it either impossible or very inefficient to search on the corresponding field.)

The way this is obtained in WDB is by using Form Definition Files (*fdf* 's). When a new query form has to be added to the system all that is needed is to create a new *fdf* file specifying what fields should be in the form and how they should be presented to the user.

3.1 Form attributes

An *fdf* is composed of a set of *form attributes*, and a set of *field attributes* for each field in the form. Examples of form attributes include :

`NAME` Name of the form.

`TABLE` List of tables to query.

`DATABASE` Name of the database to use

`TITLE` Text for the <TITLE> tag in the query form and the result page.

`JOIN` Join conditions if more than one table is specified.

3.2 Field attributes

Field attributes describe how to find the field in the database and how to present it to the user. Examples of field attributes include :

`FIELD` Field identifier. Used internally to uniquely identify a field.

`label` Label/title in forms etc.

`column` Database column definition. Defaults to the value of `FIELD`. `column` values can include Sybase computed fields, and table-names.

`type` Database type of field [int | char | datetime | ...].

`length` Length of field including formatting etc.

A whole lot of other attributes can be added to the *fdf* if needed. For example a '`url`' attribute can be added if the field should have a hypertext link. - Or converters could be added to specify how data from the database should be converted to what is to be presented to the user (the 'from_db' attribute), - or how the text entered by the user should be converted to what actually should be searched for in the database (the 'to_db' attribute).

The conversions are expressed using the very powerful Perl language. In fact WDB was coded entirely in Perl, with a few extensions to enable access to Sybase.

3.3 Example of a Form Definition Files.

The following *fdf* file demonstrates several of the features of WDB.

```
NAME       = preprints
TITLE      = ESO Preprints Database
DATABASE   = library
TABLE      = preprint
ORDER      = authors
DOCURL     = /eso/preprint.html
RECTOP     = $val{'title'}
COMMENTS_TO= malbrech@eso.org
PERL       = require "eso_util.pl"; # Include extra perl functions
#------------------------------------
FIELD      = date
label      = Received
type       = int
length     = 10
from_db    = &cvdatein( $val{'date'}, "DD MMM YY" );
to_db      = &cvdateout( $val );
default    = '&gt;01-Jan-'.`date +%y`
no_tab

FIELD      = authors
label      = Authors
type       = char
length     = 255
tablen     = 20
from_db    = &textohtml($val{'authors'})

FIELD      = title
label      = Title
type       = char
length     = 255
from_db    = "<em>".&textohtml($val{'title'})."</em>"
url        = "$WDB/eso/$form{'NAME'}/query/".&cgi_encode($val{'title'})
no_full
key

FIELD      = reference
label      = Reference
from_db    = "$val{'institute'}, <em>$val{'numb'}</em>"
computed
no_tab
no_query

FIELD      = institute
column     = inst
label      = Institute
type       = char
length     = 255
no_tab
no_full

FIELD      = numb
type       = char
hidden
```