

Databases & Content Management



10 July 2007

Helen Varley Sargan

For many years now the use of databases for handling and maintenance of information on web servers has been held up as a possible ideal solution. There are many advantages to taking this strategy - use of databases allows incorporation of dynamic content and control of templating that is not possible via any other route. The strategy is not perfect for all web pages though, and for diverse content the results may not be ideal and maintenance of several templates onerous, so projects need to be assessed carefully before committing to a database-driven strategy.

Databases

The use of databases on the web is usually referred to as being at the 'back-end', because they hold the information for the page in a raw form that will be finally presented possibly in a number of different ways.

There are two basic ways that databases can be used on the web:

- as simple containers of information that is then put into a template to form the pages that make up a website - the principal of a content management system
- as a repository of information that can be actively queried and from some or all of which a page is created in response. In addition a database can be set up to acquire information from a user.

Databases will **only** be effective with suitably structured data.

Simple templating and automating

Pages created from a database into a template framework are no different to standard web pages - the pages are assembled using a script that takes the data and inserts it by some means into a template, which is then stored in a static form until a further update of the content takes place. It is virtually impossible to detect pages that are generated in this way as their final appearance is of a static page.

Virtually any database can be used in a simple way to generate pages by inserting database fields into a template composed of the html. In many cases products (such as InDesign and Word) can output XML (by mapping fields to XML tags), which can then be used with appropriate transformation to produce web pages, either static or dynamically. XML can be output from many databases, but the structure of the data is essential here - poorly structured XML will not be useful for much.

Products such as WebMerge (<http://www.fourthworld.com/products/webmerge/>) and Mergemill (<http://www.crossculture.com.hk/>) will produce static pages from various databases into templates. The processes can be scripted.

Automating these processes without using a commercial product may take somewhat more effort, and in doing so the system is becoming 'content management' by the back door. Common routes are listed below:

Database	Type of product	Scripting or interfacing tool used
DBM file	Open source	Perl
MySQL	Open source	php or Perl
PostgreSQL	Open source	php or Perl

FileMaker Pro	Commercial	In-built, other commercial product (Lasso) or php
Access	Commercial	In-built
Oracle	Commercial	In-built

(in addition, the databases in the above list can use ODBC connections to connect to a scripting tool on another platform)

This system can be used to create pages dynamically but (if not handled carefully) the process can use much server power. For quickly changing content, such as news stories and schedules, this method is ideal.

For queries

Use of an interactive database on the web is essentially a three-component system:

- the web client 'front-end'
- the interface of web server and scripting or interface intermediary
- the database at the 'back-end'

Things to consider when starting a web database project:

- Be clear what the aims are - the critical part of designing such a project is organizing the database so the user can ask the most relevant questions and get appropriate answers.
- where the database will be hosted (and check that this can take place) and use software suitable for the platform.
- how the database will be maintained so that the information manager can have suitable access to update it, and a sound back-up strategy. If the database fails while in use it may be corrupted.
- who will monitor it, to ensure it is still running and check logs to find out usage.
- keep the software up-to-date if there are software upgrades and security patches.

Examples using MySQL or Filemaker

An example of an interactive database is at <http://mediaguide.admin.cam.ac.uk/>. In this example the database contains information about people that can be queried in a number of different ways. Intermediary query language on the web pages allows the database to be searched and the results displayed using a template. For each query available there is a different template arranging the results in a particular way.

Another, more complex, example is at <http://server.vet.cam.ac.uk/>. This is a relational database, pulling references in from a reference database and building a list of short and full references and links to online information for display on the web.

Other set-ups

- In one of the examples the database is FileMaker Pro, which has its own web interface language (or can now use php instead, see above).
- Access can be web-enabled via ASP to an IIS server, as well as being web enabled from a non-Windows web platform by using a tool such as Sun Java System Active Server Pages 4.0 (was Chilisoft-see <http://www.sun.com/software/chilisoft/>) with Apache (but not all the ASP settings may be functional), or using the Microsoft ODBC drivers to give access to the database.
- There is also the option of using MySQL (or another open-source database) and have an Access front end that links through to MySQL using ODBC (with MyODBC for connectivity). This approach can be used with other databases that can be enabled to use ODBC. (See <http://www.phpbuilder.com/columns/timuckun20001207.php3?page=1> for useful information about how to tackle php-Access connections across platforms.)

Acquiring information

In the Mediaguide, the database was initially used to collect online the information from the subjects in the database. These records were then edited and transferred to the final version of the database for use

as a resource. It is usually quite easy to enable collection of data from users but it **must** be kept separate so that edited and unedited materials don't get mixed up.

Information management

There are a group of products that put in place a framework for managing information, on top of which can sit a more specialised module for content management. They are often, but not solely, open source, and a number are listed here:

Product	Type of product	CMS add-ons or other info
Apache Cocoon	Open source (Java)	All undergoing active development Daisy - http://cocoondev.org/daisy/ Hippo - http://www.hippocms.org/ Lenya - http://lenya.apache.org/
AxKit	Open source	Is an XML application server using built-in Perl interpreter - http://axkit.org/
HTML::Mason	Open source (Perl)	The Mason framework itself can be developed for use (see http://www.masonhq.com/). Bricolage is a mason cms - http://www.bricolage.cc/ (undergoing active development)
PHP plus database	Open source	phpWebSite (http://sourceforge.net/projects/phpwebsite/), Postnuke, Mambo or Drupal (and many others)
Joomla (derived from Mambo, also php based)	Open source	Integrated - http://www.joomla.org/
Polopoly	Commercial (Java), built on open source	Polopoly content manager - http://www.polopoly.com/
Serena Collage	Commercial (Java)	Integrated - http://www.serenainternational.com/UK/Products/collage/home.asp
Straker Shado	Commercial	Integrated - http://www.strakerinteractive.co.uk/
Terminalfour Site Manager	Commercial	Integrated - http://www.terminalfour.com/
Rhythmyx	Commercial	Integrated - http://www.percussion.com/
Typo3	Open source	Integrated (needs php) - http://www.typo3.com/
Zope	Open source (Python)	Plone - http://plone.org/

In a November 2006 award (<http://www.packtpub.com/article/open-source-content-management-system-award-winner-announced>) Joomla came first, followed by Drupal and Plone (which is much more difficult to learn but immensely powerful and extensible).

See <http://www.oscom.org/matrix/index.html>, <http://www.cmsinfo.org/>.

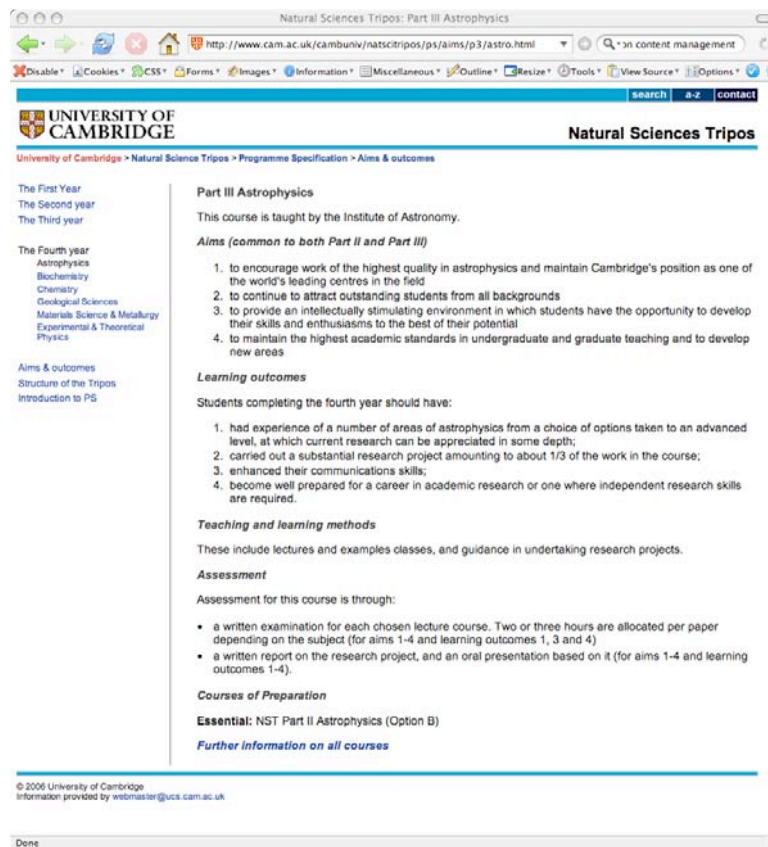
<http://www.opensourcecms.com/> and <http://www.cmswatch.com/> give details of new developments across a range of open source and commercial software and also produces a commercial report comparing products.

The Open Source products may require Apache with particular modules running, such as Jakarta (for Java), Mod-Perl, WebDAV.

A web liaison group meeting in December 2004 covered presentations from a number of people in the University using content management which covered some of these products - see <http://web-support.csx.cam.ac.uk/webliasion/wlg10.html> for links to most of the talks.

Example using HTML::Mason

The information for the programme specifications (for example <http://www.cam.ac.uk/cambuniv/undergrad/natscitripos/ps/aims/p3/astro.html>) is a good example of a standard page structure that is repeated, so we have used Mason to manage the information more easily.



The screenshot shows a web browser window displaying the 'Part III Astrophysics' page on the University of Cambridge website. The page features a navigation menu on the left with links for 'The First Year', 'The Second year', 'The Third year', and 'The Fourth year'. The main content area is titled 'Part III Astrophysics' and includes sections for 'Aims (common to both Part II and Part III)', 'Learning outcomes', 'Teaching and learning methods', 'Assessment', and 'Courses of Preparation'. The footer contains copyright information for the University of Cambridge.

The courses for each year sit in a directory in which the content for each course sits in a file. Also in the directory is a file for the breadcrumb navigation and another for the left-hand navigation:

```
-rw-rw-r-- 1 webadm web          1907 Feb 28  2003 astro.html
-rw-rw-r-- 1 webadm web          2447 Feb 28  2003 biochem.html
-rw-rw-r-- 1 webadm web          1889 Feb 28  2003 chem.html
-rw-rw-r-- 1 webadm web           238 Feb 28  2003 crumbs.mason
-rw-rw-r-- 1 webadm web          2807 Feb 28  2003 etp.html
-rw-rw-r-- 1 webadm web          2182 Feb 28  2003 geosci.html
-rw-rw-r-- 1 webadm web           359 Feb 28  2003 index.html
-rw-rw-r-- 1 webadm web           656 Feb 27  2003 lhnnav.mason
-rw-rw-r-- 1 webadm web          2635 Feb 28  2003 msm.html
```

The top part of the course info file looks like this:

```
<%method title>
Natural Sciences Tripos: Part III Astrophysics
</%method>

<%method heading>
Natural Sciences Tripos
</%method>

<h1>Part III Astrophysics</h1>
<p>This course is taught by the Institute of Astronomy.</p>

<h3>Aims (common to both Part II and Part III)</h3>
```


- to encourage work of the highest quality in astrophysics and maintain Cambridge's position as \$

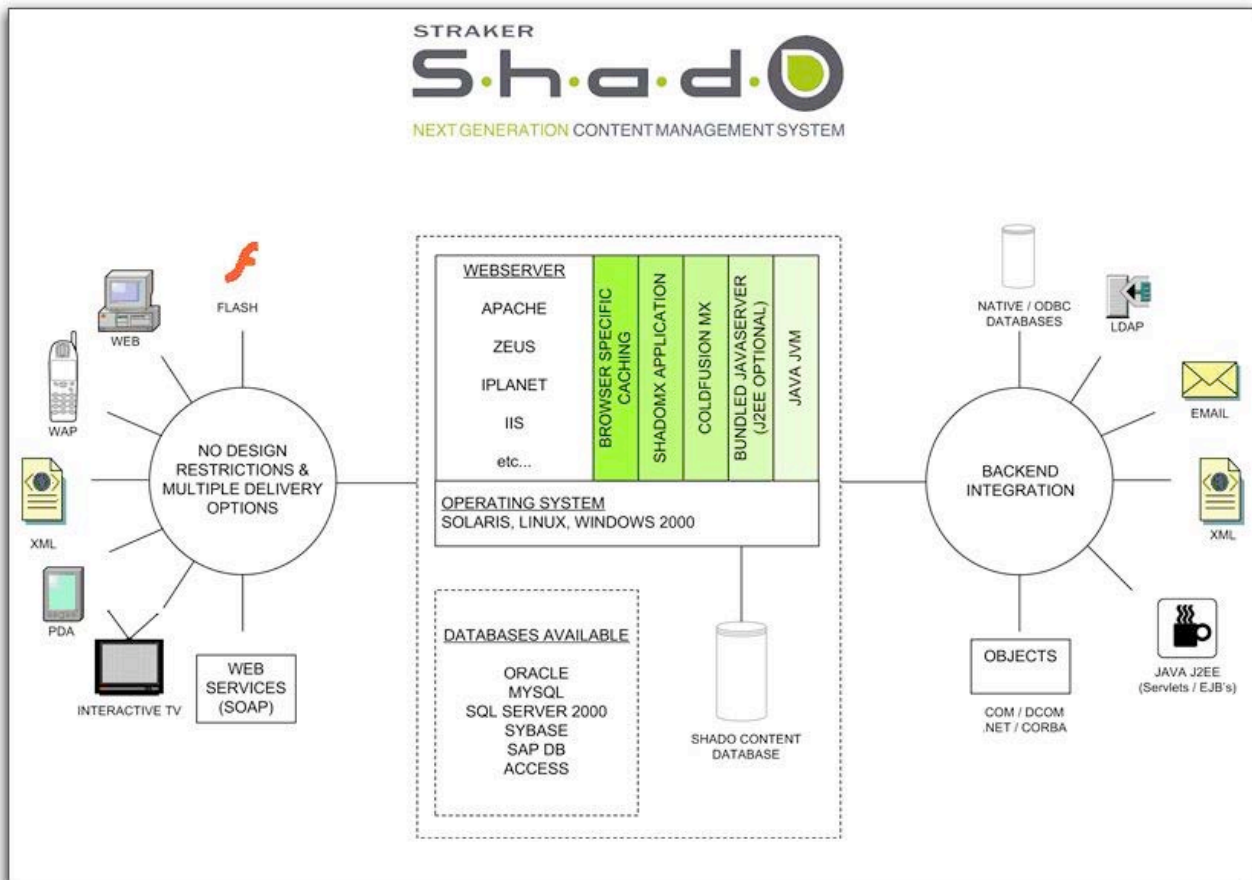
and the breadcrumb info looks like this:

```
<& /hs/breadcrumbs.mason,  
'University of Cambridge' => '/',  
'Natural Science Tripos' => '/cambuniv/undergrad/natscitripos/',  
'Programme Specification' => '../../',  
'Aims & outcomes' => '../',  
&>
```

A file called autohandler.mason configures what happens to the components (if there is no file of this name in the same directory as the file being requested, mason will look in the parent directory for one, so a whole tree of information can be configured by the same file). In this case the title and heading from the course file will be taken and placed with the template, and the breadcrumb navigation fragment and left-hand navigation will be dropped into the template also, so the pages can be built up without having to worry about adjusting all the elements manually (the title will also be inserted as part of the metadata). When the page is served to the user, only the line `<meta name="generator" content="HTML::Mason" />` shows this wasn't generated by hand.

Content management systems

As mentioned above, broadly speaking content server management is done through using a database for taking page content and inserting that content into one of a range of templates, usually through the following process:



Many content management systems (cms) are bespoke, intended for commercial organizations, and cost very large amounts of money. The most important aspects of a cms are:

- that they suit your requirements, the way your team works (both for building and maintenance of the system and for maintaining the content) and their proficiencies - this is the most usual failure point for cms installations
- cost - not just for buying the product but for training, keeping content up to date and changing templates or instituting page design changes. Make sure you know what you are getting
- lock in - when you want to change product or the company goes belly-up, can you get your content out again?

In an institutional setting, since they are all open source, the ones to look at first are probably Plone (<http://www.plone.org/>), one of the php-based tools (if you are proficient, try Joomla [<http://www.joomla.org/>], which has recently won an award) and one of the Mason tools - (see Mason <http://www.masonhq.com/>) since they are open source. Daisy and Bricolage are also worth evaluating - Daisy is being actively developed and is becoming a respected choice. Of commercial solutions, several universities in the UK are using each of:

- Polopoly (<http://www.polopoly.com/>),
- Shado from Straker (<http://www.strakerinteractive.com/>) which is a mid-price solution built on Cold Fusion,
- RedDot (<http://www.reddot.co.uk/>) which is a mid-price solution built on IIS
- Terminal Four Site Manager (<http://www.terminalfour.com/>)
- Collage (<http://www.serenainternational.com/UK/Products/collage/home.asp>).

Problem areas

Major difficulties arise from several places.

- Development time and effort can be extensive and involve highly skilled staff and/or lots of money.
- Usability and complexity - the system must be usable by those contributing information, else more skilled staff will be used for this process (when the system was meant to help them out).
- Technically, thought must be given to making friendly URLs from dynamic database pages, so that the pages can be referred to, indexed and bookmarked (see reference below for how to do this).
- Training - contributors will need training even when systems are simple to use. The need will be ongoing.
- Sameness - many cms driven sites look the same.
- A cms can only solve so many problems. Getting content will still be as difficult as it ever was.

Further help

- **LAMP** - Linux, Apache, MySQL, php website is at <http://www.onlamp.com/> - see http://www.apple.com/downloads/macosx/unix_open_source/mamp.html for Macintosh equivalent, and <http://www.wampserver.com/en/> for a Windows version. Package updates give bundled updates of all pieces of software.
- **Selecting a CMS (October 2006)** - <http://contenthere.blogspot.com/2006/10/selecting-cms.html> - there are links to further information here as well
- **How to evaluate a content management system** - http://www.steptwo.com.au/papers/kmc_evaluate/index.html - there are other papers on this site well worth reading too
- Web enabling databases for scientists: see <http://www.ictp.trieste.it/~its/>
- Web Database Applications with PHP and MySQL: O'Reilly
<http://www.oreilly.com/catalog/webdbapps2/> PHP Center can be found at <http://www.onlamp.com/php/>
- Embedding Perl in HTML with Mason: O'Reilly - <http://www.masonbook.com/book/>
- Making dynamic pages searchable: <http://www.searchtools.com/robots/goodurls.html> (with refs)
- <http://www.phpbuilder.com/> including Building Dynamic Pages With Search Engines in Mind: <http://www.phpbuilder.com/columns/tim19990117.php3>