



Technical Evaluation

of selected

Portal Systems



Table of Contents

1	INTRODUCTION.....	3
1.1	Introduction.....	3
1.2	Evaluation methodology.....	3
2	THE EVALUATION – STAGE ONE.....	4
2.1	Terms of Reference.....	4
3	PORTAL EVALUATION – STAGE ONE	6
3.1	The 5 Portals	6
4	STAGE ONE - FEATURE COMPARISON	8
4.1	Comparison Chart.....	8
4.2	Stage One Conclusion	9
5	PORTAL EVALUATION - STAGE TWO.....	10
5.1	Introduction.....	10
5.2	MetaDot.....	10
5.3	TikiWiki	11
5.4	Typo3.....	12
5.5	Final Comments	12
5.6	Stage Two Conclusions.....	13
6	PORTAL DECISION AND REASONING.....	14
6.1	Decision.....	14
6.2	Final Comment	14

1 Introduction

1.1 Introduction

This document is part of the *Evaluation of Learning Management Software* undertaken as part of the *Open Source e-Learning Environment & Community Platform* project. One of the goals of the project is to select a best-of-breed Open Source Learning Management System for development and large-scale deployment.

The criteria for this review expand on a subset of those criteria set forth in the *Evaluation of Portal Framework Software*, focusing on the technical aspects of the systems.

In an effort to make this review relevant and accurate, a significant effort has been made in gaining *hands on* experience with each software package rated, and their related communities and documentation.

1.2 Evaluation methodology

The Terms of Reference for the first stage of the portal evaluation are derived from the Open Source e-Learning Platform Project requirements document, and from the feature-sets provided in addition by the portals studied.

uPortal was eliminated prior to first stage analysis because this initiative did not meet the criteria of open source community development that is core to the NZOSVLE project. uPortal has a JA-SIG Collaborative license as opposed to a standard open source license. The community can be described as a collaborative rather than an open source community structure. The code is primarily java which is less attractive than the php scripting language for web applications.

A total of 5 portals were chosen for the first stage analysis, from a choice that covered the major open source portal and content management options available at the time.

Based on our findings from this first stage, three portals were chosen for more in-depth analysis.

Finally, the surviving 2 portals went through a further evaluation stage where their development communities and philosophies were scrutinised to assess which best met the needs of the NZVLE project.

2 The Evaluation – Stage One

2.1 Terms of Reference

The Terms of Reference for the first stage of the portal evaluation are derived from the Open Source e-Learning Platform Project requirements document, and from the feature-sets provided in addition by the portals studied. A total of 5 portals were chosen for the first stage analysis, from a choice that covered the major open source portal and content management options available at the time.

In essence the first stage consisted of a feature-by-feature comparison for the following features.

- **User/Group management**

The ability to provide reasonable features for managing users and groups, and access control for areas of the site and site functionality.

- **Version Control**

The ability to view and restore previous versions of site content.

- **Audit Trails**

The ability to determine when modifications occurred, and who performed them.

- **Platform Compatible with LMS**

The ability to operate on the same platform as the LMS, preferably capable of utilising the same RDBMS and web server.

- **Multi-language**

Capable of supporting pages in multiple languages properly.

- **Workflow**

Capable of supporting simple workflow for content editing and creation. While the requirements document suggested that workflow should be compatible with the LMS, no products found were capable of externalising their workflow system.

- **Logging**

The ability to log pertinent details of the portal operation for later analysis or security review. In general this is considered met if the portal operates under an external web server application such as Apache that supports logging.

- **LDAP support for user authentication**

While LDAP was not yet ascertained as a direct requirement, the ability to perform LDAP authentication was seen as a good marker for a flexible authentication system which could be easily adapted to meet whatever the final requirements might be.

- **Calendar**

A core or plug-in calendar feature for providing information indexed by date.

- **Forum**

A core or plug-in forum feature providing forum features which could be deployed appropriately within the site.

- **RSS Feeds**

The ability to consume RSS feeds from external systems and display them.

- **Chat**

Near real-time chat provided through either web pages with refresh, or alternate mechanisms such as Java or Flash.

- **Notification**

Notification via email or other forms regarding changes to content. This feature was poorly supported in the available portals, but is considered to be easily implementable and thus not a showstopper.

- **WYSIWYG editor**

A core or plug-in feature providing an easy-to-use editor for content including a toolbar for modifying text types, tables and creating links.

- **File management**

A core or plug-in feature to allow the management of downloadable files on the site.

- **Search**

Simple keyword searching of the portal site.

- **Questionnaire**

A core or plug-in feature that provides question and answer style survey capabilities that can be deployed and managed from the portal interface.

- **Poll**

A simple multi-select style poll feature.

- **News**

A core or plug-in feature to manage date-indexed articles on a page.

- **Image management**

A core or plug-in feature allowing images on the site to be managed.

- **User publishing**

A core or plug-in feature to permit registered users to create content.

- **User customisation**

A core or plug-in feature to permit registered users to customise their view of the site. Of primary interest was the ability to define blocks of customised content that would be consistently visible on a sidebar or similar.

Note:

Of features that were left out, only one is of particular note. Automatic provisioning via Web Services was considered too complex an issue to be addressed by a portal framework. None of the available options provided explicit support for web services and the variety of web services available led to the conclusion that such support was unlikely to be of interest in any case.

3 Portal Evaluation – Stage One

3.1 The 5 Portals

Plone

<http://plone.org/>

Plone has gained considerable fame for its focus on making the powerful but somewhat esoteric Zope framework into a flexible, attractive content management solution. Plone emphasises out-of-the-box usability, and a large number of high-profile site adoptions.

Plone, as with Zope, works best in conjunction with its internal database and web server. All components integrate heavily, and the Zope framework allows the system to be heavily customised.

Metadot

<http://metadot.com/>

The Metadot portal server takes a slightly different focus to that of Plone, emphasising portal functionality but in the end it has evolved to present a similar feature set. Metadot relies on the more common Perl/MySQL combination for its platform and benefits in terms of a large support community and a dazzling array of plug-ins.

Typo3

<http://typo3.com/>

Typo3 competes almost directly with Metadot, and has found significant favour from webmasters. Its European based development has given it strong multi-language capabilities, and like Plone, Typo3 presents itself as a polished product.

Typo3's obsession with good presentation has provided it with some powerful tools for properly manipulating images, and integrating new content within the site styles.

TikiWiki

<http://tikiwiki.org/>

TikiWiki is a popular web publishing system evolved from the WikiWiki concept. With a strong focus on the user it, along with Everything2, is the only system supporting user-defined sidebars out of the box. TikiWiki makes excellent use of stable external projects for its templating and authentication requirements, and boasts excellent support for RSS.

Everything

<http://everydevel.com/>

The Everything engine is definitely the black sheep of the selection, focusing more on external users than the other products. Everything is geared towards the development of communities rather than sites, powering user-driven websites such as [Everything2](#) and [Perl Monks](#).

Everything has the distinction of being the only system to support real-time chat effectively.

Unfortunately the power of the engine itself is let down by marginal documentation and a lack of initial polish. Experts have developed uniquely powerful sites with the engine, but the learning curve and heavy customisation required have proven too steep for many.

4 Stage One - Feature Comparison

4.1 Comparison Chart

The following comparison table permits a quick overview of the feature-set of each project.

<i>Feature</i>	<i>Plone</i>	<i>MetaDot</i>	<i>Typo3</i>	<i>Everything</i>	<i>TikiWiki</i>
User/Group Management	Yes	Yes	Yes	Yes	Yes
Version Control	Yes	Limited	Yes	Yes	Yes
Audit trails	Yes	Limited	Yes	Yes	Yes
Platform compatible with LMS	Yes	Yes	Yes	Yes	Yes
Multi-language	Yes	No	Yes	No	Yes
Workflow	Yes	Limited	Limited	Limited	Yes
Logging	Yes	Yes	Yes	Yes	Yes
LDAP Support	Yes	Yes	Yes	Limited	Yes
Calendar	Plug-in	Yes	Plug-in	Plug-in	Yes
Forum	Plug-in	Yes	Plug-in	Yes	Yes
RSS Feeds	Yes	Yes	Limited	Plug-in	Yes
Chat	No	No	Plug-in	Plug-in	No
Notification	No	Limited	No	No	No
WYSIWYG editor	Plug-in	Yes	Yes	No	Yes
File management	Yes	Yes	Yes	No	Yes
Search	Yes	Yes	Yes	Yes	Yes
Questionnaire	Plug-in	Limited	No	Plug-in	Yes
Poll	Plug-in	Yes	Yes	Plug-in	Yes
News	Plug-in	Yes	Plug-in	Yes	Yes
Image management	Limited	Yes	Yes	No	Yes
User publishing	Yes	Yes	Yes	Yes	Yes
User customisation	Limited	Limited	Limited	Yes	Yes

4.2 Stage One Conclusion

With the completion of the first stage evaluation, two portals were removed from consideration. Everything2 simply did not provide enough to build upon. Weak documentation and lack of application polish made it a poor contender.

Plone was also removed. While its showing overall was impressive, it did not provide compelling advantages over the other options. Its use of the Zope platform imposed problems in terms of local knowledge, and sharing a system properly with the LMS which is PHP/PostgreSQL/Apache based. Presented with other viable alternatives, there was little reason to follow this choice.

The remaining three, MetaDot, TikiWiki and Typo3 were subjected to more in-depth analysis in the second stage evaluation.

5 Portal Evaluation - Stage Two

5.1 Introduction

For the second stage of the portal evaluation, each portal was analysed in four key areas, with a view to how well it could be integrated and extended to meet the needs of the project.

The first key area was architecture. In the process of a general code review, an overall view of the architecture of the software was obtained and outlined. The implications of the architecture in view of the expected use scenarios are outlined.

The second key area was the quality of the code itself. While part of the objective for the portal development is to make as much use of the current code base as possible, no evaluated portal fulfilled all the desired features. As a consequence it was understood that whatever the choice, some degree of additional development would be required.

The quality of the code has a direct effect on this, helping or hindering the development and maintenance of code changes.

The third key area was documentation. With limited time and budget available, additional documentation above and beyond that intrinsic in the code itself helps bring developers up to speed faster, and with greater confidence. The provision of documentation both at the code level and as templates for portal user guides act not only as a key indicator of how focused the developers are on providing a complete product, but potentially save time and money that can be put to use providing more requested features.

The last key area was interface. While the selected portal will undoubtedly undergo changes to accommodate the interface requirements of the overall project, a general interface philosophy that is both intuitive and naturally aligned with that of the project as a whole is preferable.

5.2 MetaDot

Architecture

MetaDot provides a two tier architecture at the server side, interfacing with an RDBMS directly. The primary functionality is hard-coded in the core, with additional plug-ins available using a system called a Gizmo. The main consequence of the architecture as it stands is that html and strings are contained directly within the core, making the addition of other languages impractical.

Code quality

MetaDot's code base was disappointing. In keeping perhaps with an application that has evolved heavily over its lifetime, much of the application was uncommented, with poorly named functions, improper abstractions and large, rambling sections of code that contained obvious redundancies.

Of more immediate concern was the identification, within minutes of beginning the review, of SQL-injection class security issues. While it is believed the holes identified are unexploitable when MetaDot is used in tandem with current versions of the MySQL database, their existence suggests an extremely lax

attitude to handling of user-supplied data.

Documentation

Metadot's documentation was reasonably strong, covering all the bases. Not all the documentation was up to date, the latest developer documentation was for the previous version of Metadot for instance, but the primary documents were clearly maintained.

Interface

Metadot's interface was the most obvious of the offerings, providing a reasonably straight-forward set of operations for adding content to pages and managing users. Even so, some additional development would be necessary to bring the administration interfaces to the level where non-technical users would be easily capable of using them.

5.3 TikiWiki

Architecture

TikiWiki provides two tier architecture at the server side, interfacing directly with an RDBMS. A heavily modular architecture, it benefits from low coupling between components. Unfortunately no provision has been made for the performance impact of this architecture.

It is certain that much of the load can be split between multiple servers without major problems, suggesting that appropriate investment would permit TikiWiki to support the necessary levels of traffic.

It is certain that a modest amount of programming effort could result in significant improvements in performance for the specific needs of the project.

Comparative benchmarking suggests that the practical impact is not overly significant.

Code quality

TikiWiki presented a reasonable code base. While a number of areas of aesthetic concern were identified, including several which could have mild performance effects, overall it was workable, helped enormously by its consistent use of companion projects ADODB, Smarty and PEAR appropriately.

Documentation

TikiWiki fails to provide effective complete documentation. While there is a major effort underway to develop it, at the time of writing the documentation wiki was missing numerous entries, and updates to the documentation appeared rare (The 5 most recent changes spanned almost 2 months)

Interface

TikiWiki's interface had the surprising qualities of being reasonably easy to use, but only if the relevant control panel could be found. A number of links within the main panels appeared to be missing. Some degree of additional development would be required to resolve this issue.

5.4 Typo3

Architecture

Typo3 provides two-tier architecture at the server side, interfacing directly with an RDBMS. It provides a reasonable level of abstraction, splitting the code effectively into related files and providing a caching mechanism that significantly improves its responsiveness on average.

Code quality

The Typo3 code base was clean and well maintained. Comments were regular and to the point, and in general the code read well with no surprises.

Documentation

Typo3 provided by far the most, and highest quality documentation. The entire application core and APIs had complete and up-to-date documentation available, as well as a vast array of documentation for all plug-ins, various use scenarios and tutorials.

Interface

While Typo3's interface was very complete, providing access to the vast majority of the functionality offered by the framework, it could not be called intuitive. Some aspects were fairly obvious but without a reasonable amount of time spent reading the relevant documentation and understanding the terminology used, even some relatively trivial tasks appeared difficult.

In addition, it was discovered that browser support in the backend interface is limited. All major browsers operate, however Netscape v4 and Opera v6 do not. In light of the desired use of embedded HTML editors with limited platform support this issue is only minor, but worthy of note.

For use by end-users in the overall project, some development effort in producing an interface that works more intuitively will be necessary.

5.5 Final Comments

Performance

A major effort at measuring the performance of each contender was not performed. It is accepted that some performance development will be necessary, however each of the contenders displayed characteristics that permit them to be scaled up to cope with the heavy usage levels anticipated for the project. While Typo3 had the most support natively for performance, rough benchmarking indicated that this did not improve performance overly in comparison to the other contenders.

Security

Security was assessed at two levels. The first was the security features of the application itself, these were addressed in Stage 1, User and Group Management as

well as the associated aspects such as Audit trails. The second level concerned the security practices inherent in the code. Both TikiWiki and Typo3 displayed a healthy degree of paranoia regarding user-supplied data, using components that were designed with untrusted data in mind.

Metadot displayed a surprising disregard for the danger of using unreliable data, cheerfully assembling direct database commands with user data embedded directly with no form of checking whatsoever. While native PHP and MySQL features mitigated the end result of the breaches discovered, the use of our preferred PostgreSQL database and the use of certain PHP options could render MetaDot a severe security liability.

User Personalisation

TikiWiki provided the most comprehensive customisation tools, permitting not only the user-publishable areas supported by Typo3 and Metadot, but also personal customisation of sidebars and various other aspects.

In general all three supported the necessary elements, with additional work required to bring Metadot or Typo3 up to the level TikiWiki provides by default.

Not Covered

A number of aspects were not covered in the investigation of the portal options. The purpose of the document is the selection of the best available option. As a consequence features that were provided by all options, are trivially implementable on all options, or would require overly extensive analysis without any appreciable difference to the outcome were not considered. These include:

- Seamless integration between Portal and LMS. This requirement is the bulk of the development task and is equivalent across all options.
- Operate without a need for plug-ins. None of the options assessed required plug-ins to operate.
- Support advanced, robust searching. All of the options supported a reasonable level of search. More advanced searching would require the integration of an external search component, a task equivalent across the options, and reasonably straightforward.
- Support XSLT or CSS. All options supported CSS; none provided XML/XSLT output.

5.6 Stage Two Conclusions

At the end of stage 2, MetaDot was removed as a contender. The effort required to resolve both seemingly endemic security issues, and the lack of multi-language support was considered unacceptably high with no offsetting benefit in terms of features.

Typo3 and TikiWiki presented us with a harder choice. In the end, a final analysis focusing on the overall applications, their communities, and philosophies was performed to provide an idea of the environment in which the development team would be operating.

It is important to note that either TikiWiki or Typo3 would fulfill the role of portal framework adequately. There was no question that both supported the features, performance and structure to meet all expected requirements with an acceptable (and roughly equivalent) development effort.

6 Portal Decision and Reasoning

6.1 Decision

In the final analysis, TikiWiki stood out as the platform of choice for this project.

This determination was primarily one of communities and philosophy rather than feature wins one or another. TikiWiki is the product of a diverse group cooperating together, with a variety of divergent needs that are commonly addressed and integrated as appropriate. Typo3 on the other hand has a strong, focused leadership that has a very clear idea of its direction. This offers a benefit for Typo3 as a CMS, but not for our vision of Typo3 as an education portal.

The production of project documentation is also affected by this difference in community. While the Typo3 documentation is definitely the most comprehensive and complete, TikiWiki wins out because its documentation is provided in a Wiki that the development team can modify and work on in collaboration with other members of the community, providing an easy, centralised location for much of the writing effort.

While Typo3 and TikiWiki share similar levels of functionality in terms of theming, customisation, database and authentication abstraction, there is a clear preference for the TikiWiki approach of using standard, independent module (primarily from the PEAR project) for performing this work. The development team is already familiar with many of the PEAR libraries used which provides an immediate knowledge base, and the components are more likely to meet the future needs of the project since they are used in many other, unrelated applications and thus offer a more stable, generalised interface.

6.2 Final Comment

TikiWiki's development community's philosophy supports our desire to be involved heavily with the project and integrate much of our work, its design is based on standard modules with which we are already familiar and its community is already diverse enough that the addition of another group with a particular focus on education can be well accommodated.

Further work will need to be done to bring TikiWiki to the level that is required. Weaknesses in interface and architecture will need to be addressed, and various features tweaked and tuned for our requirements. Fortunately the vast majority of the required feature set is already supported by default, and the changes that need to be made are straightforward.

The resulting portal application will effectively meet the needs of educational institutions as a learner-centric aide for accessing information, publishing and integrating with the LMS.