Adaptive Reuse of Libre Software Systems for Supporting On-line Collaboration

Paul Adams, Cornelia Boldyreff, David Nutter and Stephen Rank

University of Lincoln
Lincoln, UK
+441522837310

{padams,cboldyreff,dnutter,srank}@hemswell.lincoln.ac.uk

ABSTRACT

In this paper, the adaptive reuse of Plone; an open source content management system is described. In one instance, Plone has been used as the backbone of a collaboration and communication support infrastructure within a large research project. In the other, Plone has been used as the main web-presence of a specialist group of the British Computer Society. This paper analyses the benefits and problems of reusing Plone to support collaboration. Based on this reuse experience, a more systematic approach to supporting Plone reuse is proposed. This approach takes into account the special case of reuse support relevant to open source software developments.

Categories and Subject Descriptors

D.2.13[**Software**]: Software Engineering – *Reusable Software*

H.5.3[**Information Interfaces and Presentation**] Group and Organization Interfaces- *Web Based Interaction*

General Terms

Design, Human Factors

Keywords

CALIBRE, Plone, Collaboration, Reuse

1. INTRODUCTION

Reuse is a well-established issue in software engineering practice. In particular, object-oriented programming has been seen as a harbinger of of reusable artifacts [1,2]. From method and function reuse the OO software engineering community has progressed to class reuse. This, in turn, has led to more generalised software component reuse, such as class libraries. Reuse, however, reaches beyond coding practice; reuse of test cases [3] and system design patterns [4] are also common practice. The libre[1] software paradigm provides software engineers with an extensive library of reusable code, patterns and tests artefacts. Environments such as SourceForge[2], FreshMeat[3] and Savannah[4] act as repositories for such reusable artefacts.

This paper concentrates on a particular form of reuse: adaptive reuse. In this case, an entire software system is reused through adaptation. This may be via customisation, with only a few modifications to its parameters, in order to serve different purposes, or it may involve more extensive change and possibly extensions. The particular libre system reused has been Plone[5]: a content management system. This reuse of Plone, within different environments, has been so effective that it has given rise to the possibility of developing a more focused reusable toolkit for developing support environments for collaborative activities, based on Plone.

Open Source Application Spaces: Fifth Workshop on Open Source Software Engineering (5-WOSSE) May 17, 2005, St Louis, MO, USA.

2. OVERVIEW OF PLONE

Plone is an open source content management system, designed to be run in conjunction with the Zope[6] application server for web-based systems. A number of additional software components have been developed to extend Plone and many have been made available for reuse as open source software. Plone, as a Zope-served application, is based on the Python programming language.

Plone, as with other content management systems, allows for the content of websites to be updated easily. With Plone these updates can be performed through the web itself. The real power of Plone is in its user base. Plug-ins, developed by users of Plone, can transform the system from a server of static content into a rich, web-based groupware tool. Plug-ins and tools for Plone are developed using the Python scripting language. In general, any Python script can be incorporated into a Plone site, either through Zope or directly as script from the host file system.

Unlike some other content management systems, Plone does not run its own server; this is handled by the Zope server. This allows for quick setup of new sites, where one already exists, as one Zope server can handle the requests for many Plone-based websites. Initially, this possibility of such a quick setup influenced the decision to use Plone in the two contexts described below.

---

[1]The term "libre" is used here to refer to both the free software and open source communities

[2]http://sourceforge.net/

[3]http://freshmeat.net/

[4]http://savannah.gnu.org/

[5]http://plone.org/

[6]http://www.zope.org/

3. CONTEXTS OF PLONE REUSE

In this section, the two different contexts in which Plone has been reused are described. Firstly, it has formed the basis for the collaboration and communication infrastructure for the EU FP6 Coordination Action for Libre Software Engineering for Open Development Platforms for Software and Services ("CALIBRE"[7]) project. Secondly, based on this earlier use, it has been adapted as the content management system for the newly-formed British Computer Society Open Source Specialist Group. These two contexts differ greatly in their requirements; however the flexibility of Plone has allowed the same system to be adapted to meet the requirements in both cases.

3.1 Plone in the CALIBRE project

CALIBRE is an EU-funded coordination action involving some of Europe's leading authorities on libre software development. The project incorporates the research of 12 industrial and academic partners with the aims of integrating existing research in the libre software field, developing a road map of current and future research, and establishing a European industry forum for libre software policy.

To support the activities of the CALIBRE project the decision was made to develop an on-line collaborative working environment [5]. This environment needed to be an area in which CALIBRE researchers could collaborate on their joint research, share documents and engage in discussions. It was also important, given the nature of the project, that the system be a showcase for the potential of libre software development.

3.2 Plone in the BCS Open Source Specialist Group

During the development of the CALIBRE Work Environment (CWE)[8], the British Computer Society's Open Source Specialist Group[9] was entering the final stages of approval by the society. In the anticipation of success, it was decided that some form of web presence had to be developed for the group; at the time, no particular requirements were laid out.

Having one Plone-based site in place (the CWE) meant that it was quick and easy to create a new one for a new purpose. At first, Plone was used to support the activities of those involved with establishing the Open Source Specialist Group. As we already had the CWE in use as a collaborative tool, developing a new Plone-based environment seemed the best solution. When the group received approval from the BCS, the role of the site changed from being a collaborative environment to being the main website of the group. It became clear that support for group collaboration (in particular the activities of the group committee) had to be retained whilst also offering services to the general membership, such as forums, mailing lists and event registration. Committee members needed to have a "private" area where they could collaborate, store documents and hold meetings, whilst the remainder of the site needed to be open to all members.

[7]http://calibre.ie/

[8]http:/hemswell.lincoln.ac.uk/calibre/

[9]http://ossg.bcs.org

4. AN EVALUATION OF PLONE REUSE

There are many benefits of reuse in software engineering. Sommerville identified [6] the five key benefits of reuse: increased reliability, reduced process risk, effective use of specialists, compliance and accelerated development. Similarly, there are many potential problems associated with reuse. Sommerville lists the following as being the main difficulties: increased maintenance costs, lack of tool support, "not invented here syndrome" and finding/adapting reusable components. In this section, we evaluate Plone against these benefits and problems, given our experience with the CALIBRE and BCS Open Source Group sites.

4.1 Benefits of Plone Reuse

Increased Reliability Because of the experience gained during the development of the CWE, the researchers developing the BCS website did not need to experiment with the installation and setup, as they did earlier in developing the CWE. This meant that the server had little downtime during the setup.

Reduced Process Risk The most powerful aspect of Plone is its plug-in system. In effect, any Python-based script can be plugged into Plone. During the creation of the CWE many problems occurred in finding plug-ins that were compatible with each other. Often we found incompatible plug-ins were being written by developers who were following poor configuration management practice. Having discovered which tool combination worked, it was possible to apply them to the BCS Open Source Specialist Group website with greater assurance.

Effective Use of Specialists Having learned a lot about Plone/Zope/Python during the creation of the CWE, we have been able to develop features for the BCS website, which can easily be applied to the CWE and any Plone based site.

Standards Compliance As standard, Plone is compliant with XHTML, CSS, US Section 508 and W3C[10] Web Accessibility guidelines. As long as the administrator does not make any radical changes to the Plone templates, these standards will remain for any Plone-based site.

Accelerated Development The development of both the CWE and the OSSG websites were carried out by the same researchers. The experience gained from developing the CWE enabled the development time of the OSSG website to be reduced significantly. Whereas development time of the CWE was four months, the OSSG website was developed within two and only by one developer.

[10]http://www.w3.org/WAI/

4.2 Suggested Problems with Plone Reuse

Increased Maintenance Costs Often, plug-ins for Plone are reliant on the presence of other plug-ins to work successfully. This had led to a situation where certain plug-ins cannot work unless they are installed on a Plone setup that is similar to the development system. In a few cases, this has meant that the code could not simply be reused; it also had to be adapted to make it work. This was time consuming as we did not always posses the knowledge of Python to get tools to work.

Lack of Tool Support This problem does not apply in the context of Plone. Almost all administrative activities for Plone can be performed through any web browser. For all other tasks associated with Plone, standard applications were used, for example, we used the emacs[11] editor for code and in order to edit graphics, the GIMP[12] was used. At no point did we find that there was a lack of tools to support our activities.

"Not Invented Here Syndrome" Plone has a very active support base and many users of Plone make their modifications freely available. Because of this, almost all modifications and plug-ins were available and did not need developing. The only time a plug-in had to be developed from scratch was as a result of the freely available plug-in being incompatible with plug-ins we had already decided to use.

Finding and Adapting Reusable Components As has already been mentioned, Plone has a very active user base. There are many repositories of useful plug-ins for Plone. Many are available through the Plone website. The most notable collection of plug-in projects is known as the "Collective"[13], providing a wide selection of plug-ins for different tasks. Most plug-ins we used have come from the Collective.

5. GENERAL ISSUES IN PLONE REUSE

Python, Zope, Plone and the different Plone plug-ins are all developed separately and are all under constant development. As figure 1 shows, each of these tools is either reliant on or has an effect on the others. This creates an potential environment of multi-dimensional evolution and distributed (possibly uncoordinated) configuration management, in which it can be difficult to work. In one instance, we found that we could not include two different plug-ins within the CWE because they relied on different versions of a third plug-in. All three plug-ins had been developed by the same team. When developing a Plone-based site, it is very important to plan the required functionality before implementing. By doing this, it is possible to determine how successful a particular combination of tools will be.

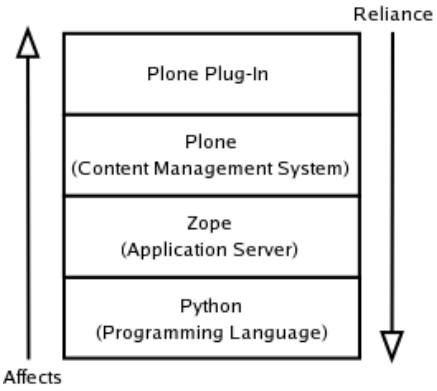

Figure 1: Multi-dimensional affects and reliance in Plone plug-in development

This stack of reliance and affects relations became important during the development of the OSSG website. It was desired that this website should include an awareness tool that allowed users to see which other users were currently logged into the website. A similar tool was first encountered on the Special Interest Group on Open Source Software in Education in Europe (SIGOSEE)[14]. The plug-in providing this feature was downloaded and installed; however, it was not compatible with the existing plug-ins. One of the key benefits of open source software, to developers, is the availability of the source code. As the code was available for the awareness plug-in, it was possible to re-engineer the tool so that it would work within our environment. However, in reuse terms, this solution is hardly ideal.

Despite the configuration management issues involved, the development of new features (not necessarily plug-ins) for Plone is easy, even with little knowledge of Python. It was required that the Open Source Specialist Group website supported event registration; however, no existing tool could be found to support this. In general, if a Python script can be created for a particular task, then it can be accessed through a Plone form as the user interface. We developed a Python script to validate inputs and enter registrants details into a database. Registrants were also automatically sent confirmation emails. Although not a plug-in, this tool can easily be integrated into existing Plone sites as it is. We plan to wrap it as a plug-in in future and contribute it to the Plone project on SourceForge, thus increasing its reuse availability.

6. CONCLUSIONS

The requirement of electronic support for collaborative research has been recognised for some time [7]. Environments, such as the CWE, can allow distributed researchers to communicate and collaborate anywhere in the world; the usefulness of this is self-evident. This kind of environment allows for greater inclusion within the research and makes research projects more scalable as they can easily accommodate new members. This is of particular importance in a project such as CALIBRE; where the libre software mindset demands openness and inclusion. For developers with the CWE experience, it did not take too much time or effort to reuse the Plone-based CWE as a baseline for developing the web-based collaboration and communication

[11] http://www.gnu.org/software/emacs/emacs.html

[12] http://www.gimp.org/

[13] http://sourceforge.net/projects/collective/

[14] http://www.ossite.org/

infrastructure of the BCS Open Source Specialist Group. Distilling these reuse experience into a more generic form is an obvious development direction for this research. One key lesson that has been learned here is that some open source software projects have not given explicit consideration to designing reusable components from the start even though the software produced is intended to be adaptable and provides a collection of plug-in components. In our experience, the compatibility issue remains problematic and it is inefficient if each development team has to forge their own solutions. Although as we have shown that their software in the case of the Plone and Zope projects can be adapted for reuse, with more explicit design for reuse, the wealth of the reusable plug-ins associated with these projects would be more accessible to other developers. A recognition of the need for coordinated distributed configuration management and explicit tracking and control of the multi-dimensional software evolution of various components is a direction for improvement of open source software engineering practice.

References

[1]H. Mili, F. Mili and A. Mili, "Reusing Software: Issues and Research Direction," *IEEE Transactions on Software Engineering*, vol. 21, no. 6, pp. 528-561, December 1995

[2]M. Morisio, M. Ezran and C. Tully, "Success and Failure Factors in Software Reuse," *IEEE Transactions on Software Engineering*, vol. 28, no. 4, pp. 340-357, April 2002

[3]M. Fisher II, D. Jin, G. Rothermel and M. Burnett, "Test Reuse in the Spreadsheet Paradigm," In *Proceedings of the 13th International Symposium on Software Reliability Engineering*, IEEE, 2002

[4]K. Beck and R. E Johnson, "Patterns Generate Architectures," In *Proceedings of the ECOOP*, Springer Verlag, Berlin, 1994

[5]D. Nutter, S. Rank, C. Boldyreff, "An Open Source Collaboration Infrastructure for CALIBRE", In *Proceedings of Cooperative Support for Disributed Software Engineering Processes (CSSE),* 2004

[6]I. Sommerville, "Software Engineering: 6th Edition," Addison-Wesley, 2001, pp 308-309

[7]C. Boldyreff, D.Nutter, S. Rank, "The Experience of OSCAR", In *Proceedings of the International Conference on Software Maintenance*, 2004