

Plone Content Management System

Katherine Irvine
September 20, 2004
Office of Learning Technology, UBC

September 20, 2004

Katherine Irvine
Web Application Developer, Office of Learning Technology
LSK 209
University of British Columbia

Gwen Litchfield
Computer Science & Alternate Routes to Computing Coordinator
CISR 253
University of British Columbia

Dear Gwen,

Please find attached the work term report for my second Co-op term, during which I worked at the Office of Learning Technology at UBC. This report was prepared for the office, as a guide for using the content management system Plone.

Sincerely,

Katherine Irvine

Plone Content Management System

For:

Gwen Litchfield

Department of Computer Science, Faculty of Science

By:

Katherine Irvine, 38933024

September 20, 2004

Office of Learning Technology, UBC

Summary

Plone is a very powerful and flexible content management system. It can be used both as intranet and public website, and is primarily managed through a web browser. Plone successfully separates content from presentation; it requires little technical knowledge on the part of those adding and managing the content, who can choose from a variety of formats in which to present the pages. This report will give a brief introduction to Plone and give several reasons for choosing Plone. It will then cover some of the issues involved in installing, administering and managing Plone. Tips are given for working with Apache web server, backup procedures, scalability issues, and customizing the appearance of a Plone site. A list of useful products is also provided.

Table of Contents

Introduction.....	1
Reasons for Choosing Plone.....	1
Plone CMS.....	1
Overview of Plone.....	1
Plone Components.....	1
Back-End.....	2
Installation.....	2
Installing from Source.....	2
Installing from an Installer or RPM.....	2
Installing extra Products.....	2
Aliasing, IP's an Port Numbers.....	3
Make Zope Listen on a specific IP and Port Number.....	3
Make Plone listen on a specific Domain Name.....	3
Running Zope behind Apache.....	4
Backup Procedures.....	4
Scalability and Performance.....	5
Mounting Multiple ZODB Databases.....	5
ZEO.....	6
Debug mode.....	6
Security.....	6
Front End.....	7
Customizing the Look and Feel of Plone.....	7
Changing the Header.....	7
Customizing the Portlets.....	7
enabling RSS feeds.....	7
Other Changes.....	8
Products.....	8
A List of Useful Products.....	8
Appendix 1.....	10
Appendix 2.....	11

Table Of Figures

Figure 1.....	3
Figure 2.....	4
Figure 3.....	5
Figure 4.....	5

Introduction

This report covers installation, customization, and maintenance of the Plone content management system on a Linux server. Most of the information in this report can easily be adapted for use on other platforms. Where appropriate, differences between platforms will be mentioned. Plone is designed to be cross-platform and, after initial set-up, can be managed almost exclusively from a web browser.

Reasons for Choosing Plone

The Office of Learning Technology at UBC has chosen to use Plone because we wanted a product (CMS) that would serve as both intranet and public website. We wanted something that was easy for people with no technical knowledge to add content to, and which had a broad spectrum of functionality, as we are coordinating several groups, with diverse needs. We also needed something that would provide a workspace and discussion space for the e-learning community on campus.

Plone CMS

Overview of Plone

Plone is an open source content management system that runs on top of the application server Zope. Zope is an open source application server written in Python. Content management systems are useful for large organizations because they effectively separate the technical side of a website from the content.

Plone requires someone with technical skills to set up, configure, and customize. Plone is highly customizable, there are many Products available online to expand Plone, and if the back-end administrator knows Python, he or she can even create new Products.

However, those creating and maintaining content in Plone need only enough technical knowledge to fill out forms on a web browser. They do not even need to know html, or anything about creating webpages. When Plone has been highly customized, the level of technical knowledge required by those adding content to Plone sites remains low, even when they use the extra Products.

Plone Components

Plone is administered through a web browser. The Zope Management Interface (ZMI) is found by adding /manage to either the Zope server root, or a Plone site root (eg. <http://localhost:8080/manage> or <http://localhost:8080/Plone/manage>). All the content in a Zope site is stored in a database, generally either ZODB(Zope Object DabaBase) or ZEO (Zope Enterprise Objects). Plone is customized and extended through the use of Products, each of which is a collection of templates and methods written in Python. The files are on the server, not in the database, one folder for each product.

Back-End

Installation

There are two ways of installing Plone on your server. It can be installed from source, or from an installer or RPM. The installation process is similar for all operating systems, and detailed instructions for each can be found on the Plone website.

From a technical standpoint, neither method is recommended over the other. However, if you have more than one server, for example, a production server and a testing server, it is good practice to have the same type of installation and the same version on both servers. If you are running ZEO (see section on Scalability) you are required have identical installations on each client server, including the same products and versions.

Installing from Source

Installing from source is the more complicated way to install Plone, but is also the most flexible. This is the recommended route for those wanting a custom installation, those already running Zope, those planning on becoming a Zope or Plone developer (since they need access to the source code), or if the RPM or installer fails to install properly on a system. Installing from source is done from the command prompt; detailed instructions for installing from source on Linux are in Appendix I.

Installing from an Installer or RPM

This is definitely the easiest way to install Plone. In Windows, simply double-click the installer and follow the instructions. In Linux, type `rpm -i /path/to/the/Ploneinstaller.rpm`, and follow the prompts. The installer will set up the program folder for Zope, and create a default Zope instance which already contains a Plone site. The basic Products necessary to run Plone will also be installed.

Installing extra Products

1. Unzip the product's folder into your Products folder in the Zope instance. The path of the Zope instance can be found on the Zope control panel page.
2. Restart Zope.
3. Log in to Plone as an administrator.
4. Go to Plone Setup -> Add/Remove Products.
5. Select desired products and click "install."

There are occasionally problems installing new products. Products can break at either the Zope level or the Plone level. Find the broken product in either Plone setup -> add/remove products or Zope -> Products; there should be an error message provided. The two most common errors are incorrect file permissions and missing pre-requisite

products. A list of required products can generally be found in the readme file that comes with the product.

Aliasing, IP's an Port Numbers

Zope runs by default on port 8080 (8282 on a Mac); if you installed Plone using an installer, the default Plone site may be available at port 80. However, this is also the default port for Apache and other web servers, so there may be a conflict if you are also serving other web content. However, you can modify your Plone site to use whichever IP Number and Port you specify, and there are several ways of making Plone “play nice” with Apache.

Make Zope Listen on a specific IP and Port Number

To specify the IP number on which Zope listens, you need to edit `zope.conf`, located in `$INSTANCE/etc/`, and set the `ip-address` directive. By default, Zope will listen on every IP address available on your computer.

To change the port number on which Zope listens, you need to edit `zope.conf`. By default, this is in the `etc` folder in the Zope instance, although some installers may put it in a different location. If different, the location will be documented in the installer's readme file. Changing the `port-base` directive will modify all the ports that Zope listens on (Figure 1, Example 1). For example, changing the `port-base` to 1000 will make the `http-server` listen on 9080 instead of 8080, the `ftp server` listen on 9021 instead of 8021, and so on. The other option is to change the server settings and specify a `port-number` for each server type (Figure 1, Example 2). For example, you could have the `address` directive of the `http-server` set to 80, and still have the `ftp-server` listen on port 8021. If Plone has been installed using the Mac OSX installer, the port numbers for the various servers can be changed in `Plone.conf`.

```
Example 1
Port-base 1000

Example 2
<http-server>
  address 80
</http-server>
```

Figure 1

Make Plone listen on a specific Domain Name

To make a specific directory in Zope (for example, a Plone site) served to a domain name, you can use a Virtual Host Monster. In the root folder of your Zope site, add a Virtual Host Monster from the drop down menu. The Host Monster does not need to be given any particular ID. In the mappings tab of the Host Monster item, add a line for each site, in the form `host.com/directory` or `*.host.com/directory`.

Another available method for making your Plone site (or any other directory) appear to be located at a different URL is the SiteRoot product. A SiteRoot product should be added to the root of the Plone site, and a URL base and path can be specified, similar to

the VirtualHostRoot and VirtualHostPath which can be specified with an Apache rewrite-rule, as detailed below.

Running Zope behind Apache

In order to use Zope in conjunction with Apache, you must use the mod_proxy and rewrite_rule directives in Apache. It also requires that the Zope instance have a Virtual Host Monster item in the root folder (see above).

Apache can be made to serve content to by proxy from a specific Plone site. This is done by including a rewrite-rule in Apache's httpd.conf file, either in the default server or in a virtual host (See Figure 2, Example 1). What this means is that the public can go to http://www.mysite.com and see the content you would find at http://localhost:8080/Plone. Apache can also do inside-out hosting with Plone, using a slightly different rewrite-rule. The rewrite-rule shown in Figure 2, Example 2 would make that same Plone site "Plone" appear to be the folder www.mysite.com/plone_stuff.

Example 1 (single line)

```
RewriteRule ^/(.*) http://localhost:8080/VirtualHostBase/http/www.mysite.com:80/  
Plone/VirtualHostRoot/$1 [L,P]
```

Example 2 (single line)

```
RewriteRule ^/plone_stuff/(.*) http://localhost:8080/VirtualHostBase/http/  
www.mysite.com:80/Plone/VirtualHostRoot/_vh_plone_stuff/$1 [L,P]
```

Figure 2

Backup Procedures

Backing up data is always important and, for a content management system that contains all the data for several intranets and multiple websites, it is essential. Zope (and therefore Plone) stores your data in a database; by default, all your data is in a single database file. The good thing about this is that you only have one file to back up. The bad thing is that that file can quickly grow to be several gigabytes in size. This means that complete backups, especially if done nightly, can quickly fill up your storage space, while incremental backups are complicated. There are, however, several solutions.

If you have a smaller database, you can simply use a cron job (which is a standard method of scheduling tasks on Linux) to do complete backups. An example of the code required is provided in Figure 3. To have it also create backups of other Zope instances, only the backup.sh script needs to be edited.

```
the cron tab (single line):
0 0 * * * /home/administrator/backups/backup.sh
    >>/home/administrator/backups/logs/script.log
    2>>/home/administrator/backups/logs/script.err

backup.sh (three lines):
cp /var/www/html/Zope/var/Data/fs /home/administrator/backups/Data.fs
tar czf /home/administrator/backups/`date+%Y%m%d`Zopemain.tar.gz
    /home/administrator/backups/Data.fs
rm /home/administrator/backups/Data.fs
```

Figure 3

If you need to do incremental backups, there are several scripts available online. One full & incremental backup & data restores script is available at <http://cvs.Zope.org/ZODB3/Tools/repozo.py>

Finally, it is often possible to reduce the size of your database. Because Zope creates a backup copy of an item each time it is modified (for the undo utility) a database can quickly become large if items are modified repeatedly, even if there is not a large amount of content in the site. When this happens, it is possible to pack the database, to discard the old copies. This is done through the ZMI. Go to “Control Panel/Databases” and click on the database you want to pack. Enter how many days’ backup you wish to keep, and click “pack”.

Scalability and Performance

Mounting Multiple ZODB Databases

To mount a second database, edit Zope.conf and add another database entry with a new mount point (Figure 4). Restart Zope. Then, in the root directory of the ZMI, choose ZODB Mount Point from the Add New... drop down menu. Make sure the new database /second-db is checked, and click “ADD”.

If DataTwo.fs did not exist or was empty, the file DataTwo.fs will have been created on your hard drive, and the folder /second-db will now show up in the ZMI. Fill this folder with whatever content you like. Unfortunately, you cannot have a Plone site or other

```
<zodb_db main>
  # Main DirectoryStorage database
  <directorystorage>
    path $INSTANCE/var/Data.fs
  </directorystorage>
  mount-point /
</zodb_db>

<zodb_db second>
  # Secondary Storage database
  <filestorage>
    path $INSTANCE/var/DataTwo.fs
  </filestorage>
  mount-point /second-db
</zodb_db>
```

Figure 4

product as a mount point, you have to put it inside the folder. You can, however, remove /second-db/ from the apparent URL of your plone site; see the section on Aliasing, IPs, and Port Numbers.

If DataTwo.fs previously contained data, the contents of the database will now appear under /second-db/.

ZEO

For sites which are slowed down by the number of hits they receive, it may be advisable to split the load between several client servers. This requires switching to the ZEO database. This has not been done at the Office of Learning Technology; instructions can be found on the plone.org site.

Debug mode

If a Plone site runs quite slowly, it may be running in debug mode. When installing Zope from source, it may ship with debug mode on. When using an installer or RPM, it generally does not. Debug mode causes Zope to run more slowly because it constantly checks every item served from the cache against the copy in the database.

To turn debug mode off, you must edit Zope.conf, and add or uncomment the line “debug-mode off”. If there exists a line “debug-mode on” debug mode will be turned on. Also, if debug-mode is not specified, Zope will run in debug mode.

Security

Plone assigns permissions to users based on roles – Member, Reviewer, Manager, and Owner. Users are automatically assigned the Member role when they create their account, and they are automatically the owner of any pages they create. Managers and owners can give users further roles. Roles can be assigned at the site or page level.

To prevent the public from creating their own accounts, go to the security tab of the Plone site in the ZMI, and uncheck “acquire?” for “add portal member”. Then check “Manager”.

To use SSL encryption for the login form, you need to be running your Plone site behind Apache, or another web server that can handle SSL (see Aliasing, IPs and Port Numbers). You will need to edit your Apache configuration, and to create a Python method. Complete instructions can be found at darkandbrooding.com/howto/plone/20040201.

Front End

Customizing the Look and Feel of Plone

Much of the customization of a Plone site can be done through CSS. Stylesheets are used extensively throughout Plone, and are a very powerful and flexible tool for changing the appearance of a Plone site. However, some changes cannot be made solely through CSS. Listed below are some of the changes that require modifying the template files. More detailed information and instructions for the changes mentioned below (including manipulating stylesheets) can be found in Appendix II.

Changing the Header

There are two ways of doing this. One modifies the `main_template`, the other modifies `global_logo`.

Customizing the Portlets

Changing the visibility and order of the portlets can be done in the properties tab of the Plone site's root folder. You can also specify that specific portlets only appear for logged-in users, or only for members of certain groups, but this requires editing the templates for that portlet.

There are two things specific to the navigation portlet which it is useful to know how to customize. These are removing the images and changing the indentation. Both these changes require customizing the `portlet_navigation` template, which can be found in `plone-skins/portlets`.

To remove the images, delete the line `<tal:block replace="structure python:path('here/%s' % sibling.getIcon(1))" on-error="structure here/site_icon.gif" /> ` which appears twice.

To change the indentation, modify the line `str(te['indent']/2.0)`. For example, to double the indentation, change it to `str(te['indent'])`, and to remove the indentation, delete `str(te['indent']/2.0) +`.

enabling RSS feeds

1. Log in to Zope as admin through web browser.
2. Go to PloneSite1 -> `portal_syndication`.
3. Under the properties tab, click Enable Syndication.
4. Set update period to hourly and update frequency to 2. click save.
5. Under the actions tab, ensure the visible? box is checked for syndication, and click save.

Other Changes

You can change all the images and icons in the Plone site by finding the appropriate image in plone-skins/images, customizing them, and uploading a new image. Changing the small plone logo (the one that appears at the root of the site in the navigation, and as the “bookmark this page” links) is more complicated, as you may not want these to locations to share an icon.

Finally, you may not wish visitors to your website to be able to join and become members, or even realize that your site has log-in functionality. There are three things to customize to obtain this goal: hide the “you are not logged in >log in >join” bar for anonymous users, hide the “log in to add comments” button from anonymous users, and dis-allow anonymous users from registering themselves.

Products

There are many Products for Plone available online. Most can be found on the plone.org or zope.org sites, but not all. Searching the web for “Plone” and a description of the functionality you are looking for can be very helpful.

A List of Useful Products

Archetypes

This product provides the basic framework of a Plone Product for those wishing to create Products but who don't have the Python skills to create one from scratch. Archetypes must also be installed to use other Products which have been created using Archetypes.

CMFBoard

This is a bulletin-board product, allowing discussion between users of the website. For each bulletin-board created, you can specify whether users have to be logged in to use it, along with many other settings.

CMFCalendar

This is a more robust calendar than the one which is provided with Plone, but it still works with the same Event content.

CMFContentPanels

This Product creates a page which pulls content from one or more other pages in the Plone site, with a highly flexible layout. It also allows you to edit the left and right portlet columns as you would any page, rather than through the ZMI.

CMFPhoto

This Product allows for more robust handling of images than the default Image type, including allowing the user to rotate and zoom images.

CMFPhotoAlbum

CMFPhotoAlbum works with CMFPhoto, and provides a default folder listing with thumbnails.

Epoz

This allows WYSIWIG editing of html pages in Plone.

PloneArticle

A PloneArticle is a document with attached images and files. There are a number of layouts to choose from.

PloneCollectorNG

This is a very flexible and useful bug collector. It can be used to collect bug reports about the website, or it can be used to document bugs and workflow for any other project.

PlonePopoll

This allows you to poll your website audience.

PloneSearchBox

This provides a more robust search tool than is provided by default by Plone. It works with a variety of search engines.

PloneSiteMap

This tool automatically creates an up-to-date site map of the entire Plone site.

SimpleBlog

SimpleBlog allows users to create blogs within Plone.

Workgroup

Workgroup allows groups to be created on the fly, with group-owned folders anywhere in the site. It also allows groups to be managed by those who do not have the manager role.

ZWiki

ZWiki is a Wiki, Plone style. This is different in a couple of ways from more traditional Wikis, but is useful for those who wish a Wiki-style workflow, especially the changes history, but with more security.

Appendix 1

Installing Plone from source on Linux

Install Python

1. Unzip the package.
2. Navigate to the package directory.
3. Run `./configure` then `make` then `su root` and then `make install`.
4. Exit from root user.

Install Zope:

1. Unzip the package.
2. Navigate to the package directory.
3. Run `./configure` to install in the default location, or `./configure --prefix=/where/to/install/zope`.
4. Run `make` then `make install`.
5. Navigate to the install directory.
6. Run `./bin/mkzopeinstance.py`.
7. Follow the prompts to specify username, etc.
8. Start Zope with `/location/of/zope/instance/bin/runzope` – this will run Zope in the foreground and will output debugging info. If it starts correctly, open your web browser to `http://localhost:8080/manage`, and provide the username and password from step 7. You should see the Zope Management Interface (ZMI).
9. If Plone is running correctly, exit from `runzope` (^C) and run `location/of/zope/instance/bin/zopectl start`. This will start Zope as a demon (which will run in the background).

Install Plone:

1. Unzip the package.
2. Move the contents of the package to the Products folder of the Zope instance.
2. Restart Zope.
3. Log in to Zope as admin through web browser.
4. In the Root folder, add a Plone Site from the drop-down menu.

Appendix 2

Customizing the Look and Feel of Plone

All customizations detailed here are done through the ZMI. The “custom” folder referred to is at `plone_skins/custom`. All URL’s are from the root of the Plone site being customized.

Changing the Header

There are two ways of doing this. one modifies `main_template`, the other modifies `global_logo`.

The first method is best when adding a banner above the logo.

In the “custom” folder, create a new Page Template (from the drop-down menu) called `my_custom_header` with the following content:

```
<div id="my_custom_header" metal:define-macro="my_custom_header">
    ***code for the banner or header***
</div>
```

If you have not yet done so, customize the main template file (`portal_skins/templates/main_template`), and insert

```
<div metal:use-macro=" here/my_custom_header/macros/my_custom_header ">
    My custom header
</div>
```

mediately before

```
<a metal:use-macro="here/global_logo/macros/portal_logo">
    The portal logo, linked to the portal root
</a>
```

Customizing the global logo template (`portal_skins/templates/global_logo`) is recommended when replacing the existing logo with a banner or when the custom header will have code before and after the logo.

Customize `global_logo`, and add the code for your header inside the div tag:

```
<div metal:define-macro="portal_logo">
```

As well, you can create a custom template, as in the first method, and call it from within the `portal_logo` file.

Customizing the Navigation

Create a custom copy of the navigation portlet template (`portal_skins/portlets/portlet_navigation`).

1. to remove the images:

```
remove: <tal:block replace="structure python:path('here/%s' % sibling.getIcon(1))"
        on-error="structure here/site_icon.gif"
        />&nbsp;
```

(it appears twice, remove both)

2. to increase the indentation:

```
change: str(te['indent']/2.0)
to:     str(te['indent'])
```

3. to remove the indentation:

```
remove: str(te['indent']/2.0) +
```

4. to make the navigation only visible to logged-in users

```
to: <div metal:define-macro="portlet"
    i18n:domain="Plone"
    tal:omit-tag=""
add: tal:condition="not: isAnon"
```

enabling RSS feeds

1. Log in to Zope as admin through web browser.
2. Go to PloneSite1 -> `portal_syndication`.
3. Under the properties tab, click Enable Syndication.
4. Set update period to hourly and update frequency to 2. click save.
5. Under the actions tab, ensure the visible? box is checked for syndication, and click save.

Other Changes

to change the logo in the navigation panel

~upload new icon to custom folder, give unique name

~customize `portlet_navigation`, change both `"site_icon.gif"` to new icon's name

to change to bookmark icon for breadcrumbs and document buttons
~customize both site_icon.gif and addFavorite.gif

to change portlet order/existence
~Plone site root folder -> properties tab

to provide a different stylesheet for different users
~customize base properties (->properties tab)
~add properties, eg. anonymousFontColor, adminGlobalBorderColor
~customize PloneCustom.css a bunch of times, give them names like anonymous.css
~customize header
~change where PloneCustom.css is to a choice of which custom stylesheet to use,
surrounding it with the following code:s

```
for anonymous only:  
  <div tal:condition="IsAnon">  
for logged in only:  
  <div tal:condition="not:IsAnon">
```

to hide "you are not logged in >log in >join" bar for anonymous users
~customize main_template
~change:

```
<div metal:use-macro="here/global_personalbar/macros/personal_bar">  
  The personal bar. (log in, logout etc...)  
</div>
```

to:

```
<div metal:use-macro="here/global_personalbar/macros/personal_bar"  
  tal:condition="not:here/portal_membership/isAnonymousUser">  
  The personal bar. (log in, logout etc...)  
</div>
```

to hide "log in to add comments" button from anonymous users:

~customize viewthreadsatbottom
~comment out or delete the block of code that begins
 <form tal:condition="python: userIsAnonymous and not userHasReplyPermission"

removing navigation for anonymous users

~if you add a tal:condition to portlet_navigation, you get a blank slot on the left unless
you have something else in that column

~if you don't have anything else in that column, customize main_template, add the
tal:condition to

a div surrounding <td id="portal-column-one" </td>