



LAWRENCE  
LIVERMORE  
NATIONAL  
LABORATORY

# **Integrated Design and Production Reference Integration with ArchGenXML V1.00**

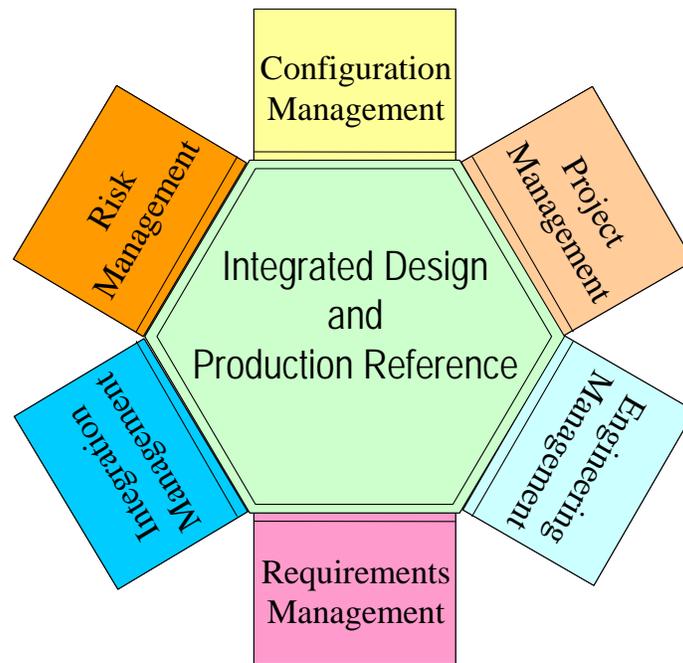
*R. H. Barter*

**July 29, 2004**

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

This work was performed under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

# Integrated Design and Production Reference Integration with ArchGenXML



Version 1.00

R. H. Barter

July 29, 2004

# Table of Contents

Overview.....	1
Configuration Requirements.....	1
Cosmetic .....	1
Structural.....	1
Presentation.....	2
Behavioral.....	3
Design Approach .....	4
argoUML.....	5
Poseidon.....	6
Plone Products Folder.....	10

## Overview

ArchGenXML is a tool that allows easy creation of Zope products through the use of Archetypes. The Integrated Design and Production Reference (IDPR) should be highly configurable in order to meet the needs of a diverse engineering community. Ease of configuration is key to the success of IDPR.

The purpose of this paper is to describe a method of using a UML diagram editor to configure IDPR through ArchGenXML and Archetypes.

## Configuration Requirements

The production version of the IDPR has a requirement that a person in the role of “Configurator” be able to configure IDPR to match a particular engineering process. Configuration has four major aspects:

1. Configuring the look-and-feel (**Cosmetic**),
2. Configuring the data structures (**Structural**),
3. Configuring how information will be presented (**Presentation**), and
4. Configuring how information will be controlled (**Behavioral**).

The software requirements for “configuration”<sup>1</sup> are listed below for each of the four aspects.

### **Cosmetic**

The following requirements can be implemented with dedicated configuration pages or as attributes in a UML diagram.

- The splash page shall include a link called “Recent Changes” in the Utilities section that will go to a page displaying the most recent N changes made to the system, where N is a number configurable by the Configurator.
- Document Storage Areas shall have a configurable title in the upper left corner.
- Document Storage Areas shall have a configurable classification level in the upper right corner

### **Structural**

The structural configuration requirements are best implemented through the use of a UML editor.

- IDPR shall have the ability to be configured with one or more hierarchies.  
*UML classes can be used to define the IDPR hierarchies*

---

<sup>1</sup> *Software Requirements Specification for the Integrated Design and Production Reference*, UCRL-MI-204395, May 28, 2004, R. H. Barter, E. A. Quinnan, A. Vinzant

## IDPR ArchGenXML Integration

- The properties of each hierarchical element shall be configurable through the Zope Management Interface (ZMI).  
*UML class attributes can be used to define the properties.*
- IDPR shall have the ability to be configured for one or more information hierarchy templates.  
*The need for hierarchy templates goes away if UML classes define the information structure*
- The number of document storage areas shall be configurable  
*If the document storage area is defined as a UML class, then the Plone interface can be used to instantiate each document storage area.*
- The title of each document storage area shall be configurable.  
*The title is an attribute defined in the UML class*
- The Configurator shall be able to define specific child objects that can be added by the user  
*The child objects can be defined as UML classes and class attributes can be used to control whether or not a child can be added in a particular situation.*
- The Configurator shall choose which child object is available for a user to add to any given Information Category Object.  
*See comment above*

### **Presentation**

- Three independently scrollable panels shall be simultaneously displayed  
*As shown below; it should be possible to define a set of abstract Display classes. A configurable UML class would then inherit from one of the abstract Display classes to implement a specific presentation.*
- The first panel shall include physical hierarchy, functional hierarchy, and zero or more user-defined hierarchies  
*See abstract Display class above*
- A configurable title shall be shown at top left corner  
*The abstract Master class will define the title attribute/property. The abstract Display class will control the display of the title. The title value will be defined for each instance.*
- A configurable classification field shall be in the upper right corner  
*See above*
- Shall have a configurable title for first panel  
*See above*
- The Configurator shall be able to define property fields for files that will appear beneath the title line in the Add File window  
*Property fields map directly onto UML class attributes. It is proposed below that there be a fixed set of attributes/properties that are inherited from an abstract class and a set of configurable attributes/properties defined for each UML class.*

*IDPR can use the fixed properties directly and interpret the configurable fields at run time.*

- The property fields that the Configurator has defined for files in the specified Document Storage Area shall show up as default property fields for the user to enter values, on the Batch Upload page  
*Same comment as for “Add File” above*

### **Behavioral**

- The Configurator shall have the ability to select one of three collaboration strategies: *permissive, informative, or restrictive.*  
*The collaboration attribute would be defined in the top level UML class. The value would be defined at instantiation time. The behavior would, most likely, be hard coded into IDPR.*
- Configurators shall have the ability to enable or disable any of the workflows.  
*Best accomplished with a dedicated configuration page*
- Configurators shall have the ability to assign roles to users for the workflows.  
*Best accomplished with a dedicated configuration page*
- On the bottom of the first and second panels, there shall be a configurable filter that can be turned on or off (by selecting or de-selecting it), that will determine (to an extent) what information is displayed in that panel, based on the properties of the objects being displayed  
*The behavior will be associated with the abstract Display class. (Does it make sense to define abstract Behavior classes – along the lines of Model/View/Controller – and then think about adaptors between an inherited display class and an inherited behavior class?)*
- Information Category Objects (elements in the second panel) shall have a title, description, type, a list of acceptable child Information Category Objects, a field to indicate whether the user can do cut/copy/paste/delete (all set up by the Configurator, and not viewable to a Contributor or Reader), a field to indicate whether or not there is an associated extension to the object (set up by the Configurator), and a field indicating display order  
*All of this can be defined in the UML class if not inherited from the abstract Master class. Display of Master class attributes can be “hard coded” in the abstract Display class. Other class attributes can be displayed by an interpreter program based on the attribute/property type.*
- Each Information Category Object shall be configurable as to whether the user can add sub-objects, and the type of sub-objects that can be added (restrictions set up by Configurator). If a user can add a sub-object, then the ability to do so shall be reflected in the pop-up “modify” menu in the third column.  
*Default behavior based on UML class attributes in the inherited master class*
- Setting Priorities: the first element/option shall be “none,” and then the rest shall be configurable as to title and icon – by the Configurator, and holding application-

- wide (no icon shall be displayed next to title for a priority setting of “none”)  
*Default behavior based on UML class attributes in the inherited master class*
- Each information object in panel three shall be able to be configured for a set of user actions, which shall appear in the Modify popup menu (configuration defined in the Default Hierarchy prototype  
*Default behavior based on UML class attributes in the inherited master class*
  - Availability of the specific extensions to any particular Information Category Object shall be set up by the Configurator  
*Defined by the UML classes*
  - The Risk Matrix shall have a configurable number of rows and columns, with each cell of the matrix assigned a level represented by a color  
*Best handled with attributes I the UML class*
  - The Configurator shall be able to configure the number of rows and columns, the labels for same, and the color and priority of each cell in the matrix  
*Best handled with attributes I the UML class*
  - The “Properties” Information Category Object Extension shall display property names and editable property value fields for any configurable Information Category Object property  
*Defined as attributes in the UML class and interpreted for display at run time as part of the default behavior*
  - “Add Document” shall be an available user action in the pop-up “modify” menu for any Information Category Object in the third panel for which an associated Document Storage Area has been set up by the Configurator.  
*Defined as an attribute of the UML class*

## Design Approach

IDPR will have a set of data structures, display screens and behaviors that are beyond the view of the Configurator. Other items will be visible to, and used by, the Configurator, but will not be modifiable by the Configurator. Visible items include Master properties and selectable Display options. Other items will be modifiable by the Configurator. Modifiable items include hierarchies and data objects.

There are several UML editors supported by ArchGenXML. From the ArchGenXML web page<sup>2</sup>:

---

<sup>2</sup> <http://plone.org/documentation/archetypes/archgenxml-manual>

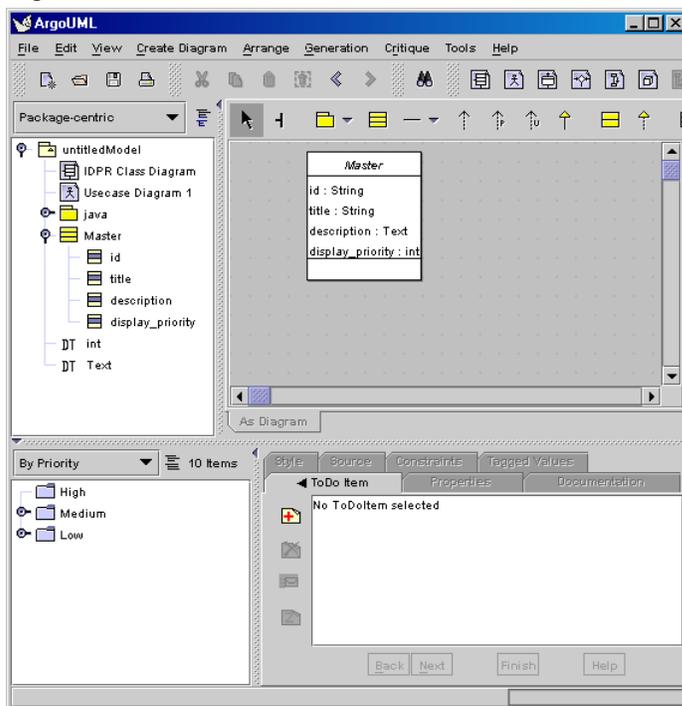
## IDPR ArchGenXML Integration

ArchGenXML is tested with the XMI output from the following tools:

<a href="#">ObjectDomain</a> <sup>3</sup>	commercial, free demo for <= 30 classes	provides the possibility to export the model to XMI
<a href="#">Powerdesigner</a> <sup>4</sup>	commercial, demo download	supports model export as XMI (XMI version 1.1)
<a href="#">ArgoUML</a> <sup>5</sup>	free	stores the model native as xmi + diagram information in .zargo files (zip files) (xmi version 1.0)
<a href="#">Poseidon</a> <sup>6</sup>	commercial, based on ArgoUML	stores the model native as xmi + diagram information in .zuml files (zip files) (xmi version 1.2)
<a href="#">KDE's Umbrello</a> <sup>7</sup>	free, Linux/KDE	umbrello saves the models native in XMI (not fully standard compliant)

### **argoUML**

Start with a set of predefined abstract classes in a supported UML editor such as argoUML



<sup>3</sup> [http://www.objectdomain.com/\\_odR30/odR3download.html](http://www.objectdomain.com/_odR30/odR3download.html)

<sup>4</sup> <http://www.sybase.com/>

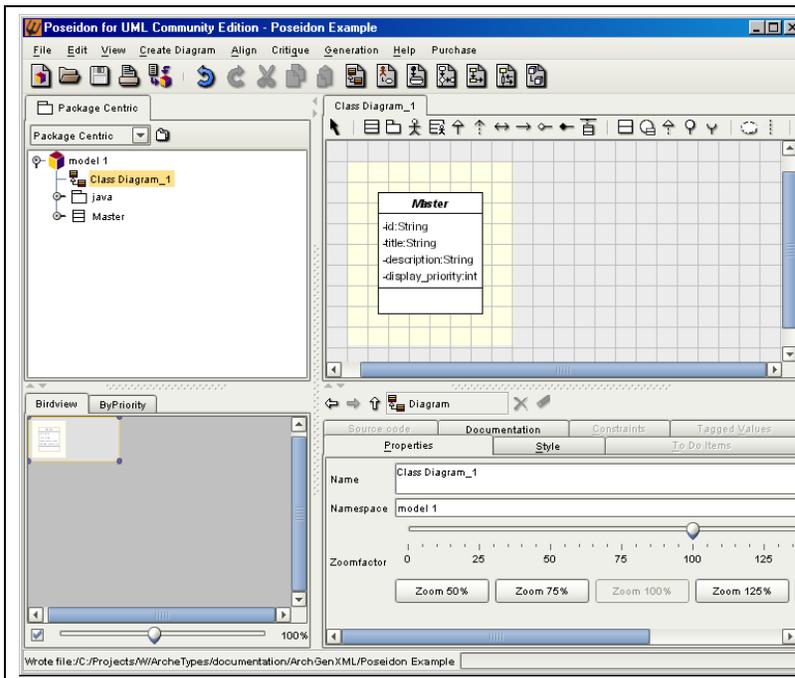
<sup>5</sup> <http://argouml.tigris.org/>

<sup>6</sup> <http://www.gentleware.com/>

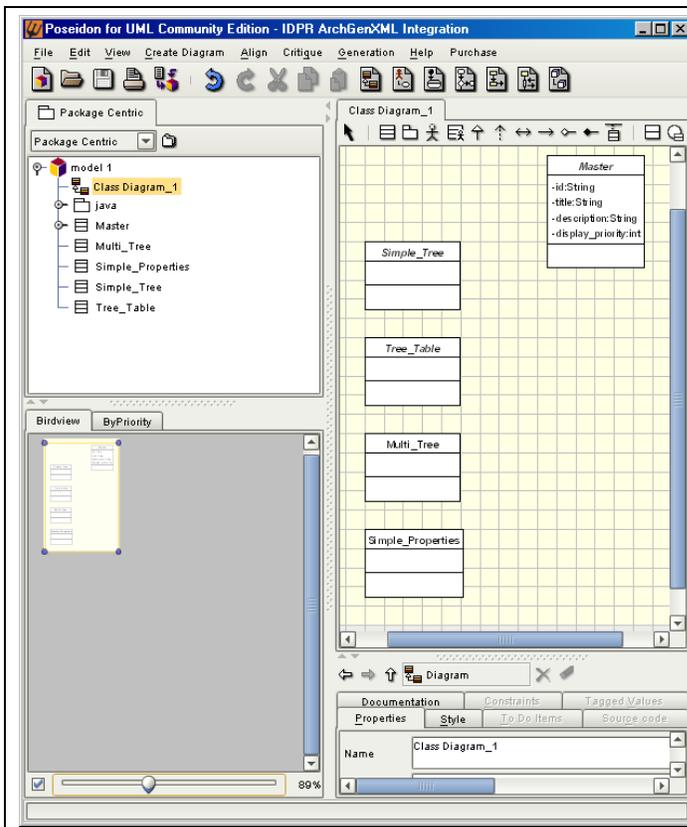
<sup>7</sup> <http://www.umbrello.org/>

## Poseidon

Poseidon has a similar interface

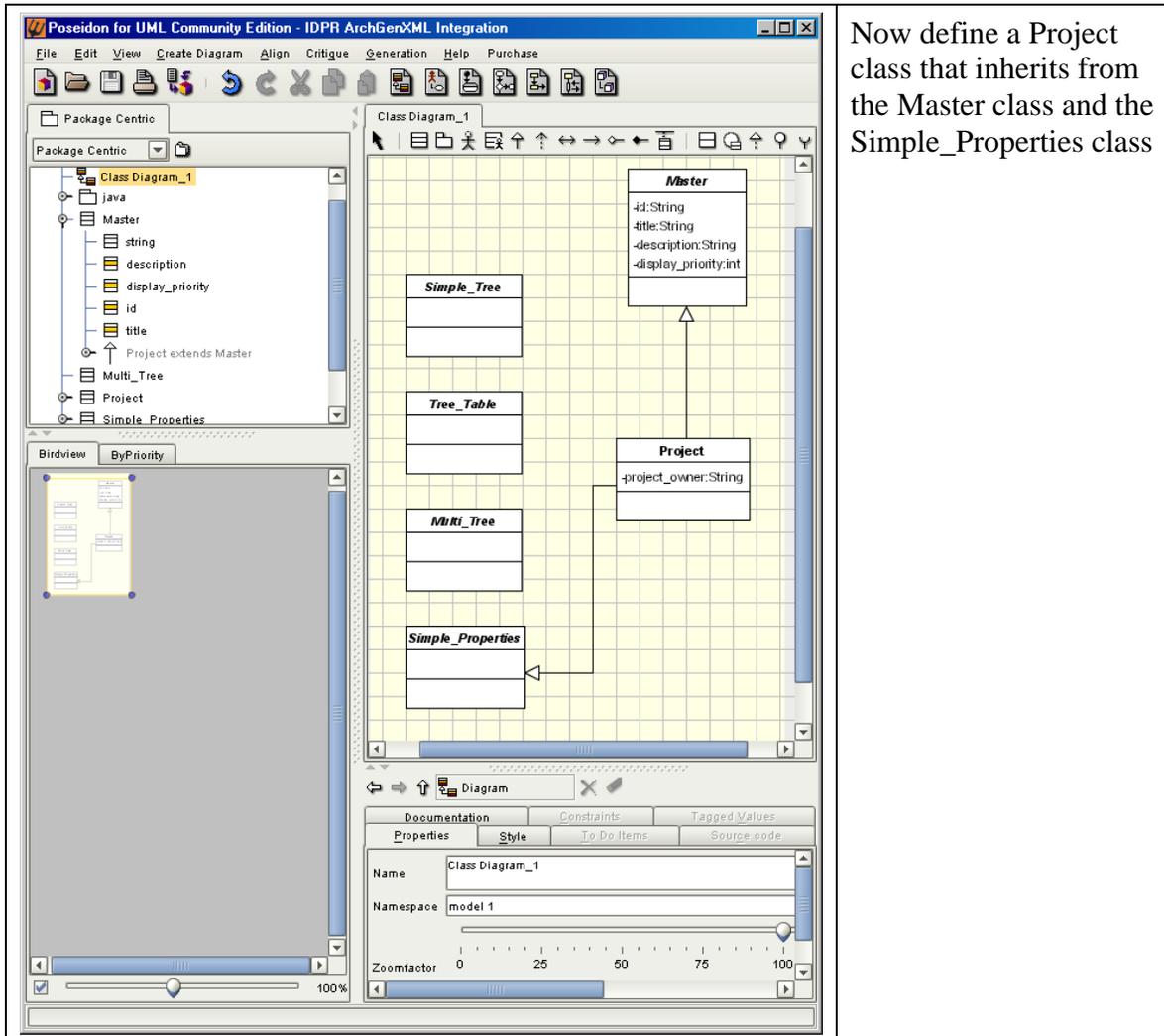


The abstract Master class will define all the attributes common across IDPR objects (such as id, title, description, display\_priority, ). Each configurable class (such as a Function Element) will inherit from the master abstract class.



Define a set of abstract Display classes. One Display abstract class will display a simple hierarchy for a defined class/product. Another Display abstract class will display a series of panels (perhaps implemented as <IFRAMES/> ), one panel per contained folderish class/product. Each configurable class (again, such as a Function Element) will inherit from one of a pallet of abstract Display classes.

## IDPR ArchGenXML Integration



Now define a Project class that inherits from the Master class and the Simple\_Properties class

Now we can export the definition as an xmi file.

It is best to create a batch file for running ArchGenXML. A command such as the following will produce an Archetype product.

```
python ..\..\ArchGenXML\ArchGenXML\ArchGenXML.py -a yes -o "IDPR ArchGenXML Integration" "IDPR ArchGenXML Integration.xmi"
```

## IDPR ArchGenXML Integration

The directory structure looks like:



Project.py contains the schema:

```
schema=BaseSchema + Master.schema + Simple_Properties.schema + Schema((
    StringField('project_owner',
        widget=StringWidget(description='Enter a value for project_owner.',
            description_msgid='IDPR ArchGenXML Integration_help_project_owner',
            i18n_domain='IDPR ArchGenXML Integration',
            label='Project_owner',
            label_msgid='IDPR ArchGenXML Integration_label_project_owner',
        ),
    ),
),
),
)
```

Note that the schema pulls in the Master.schema and the Simple\_Properties.schema. The Simple\_Properties.schemas is empty because we did not define any attributes in UML:

```
schema=BaseSchema + Schema((
),
)
```

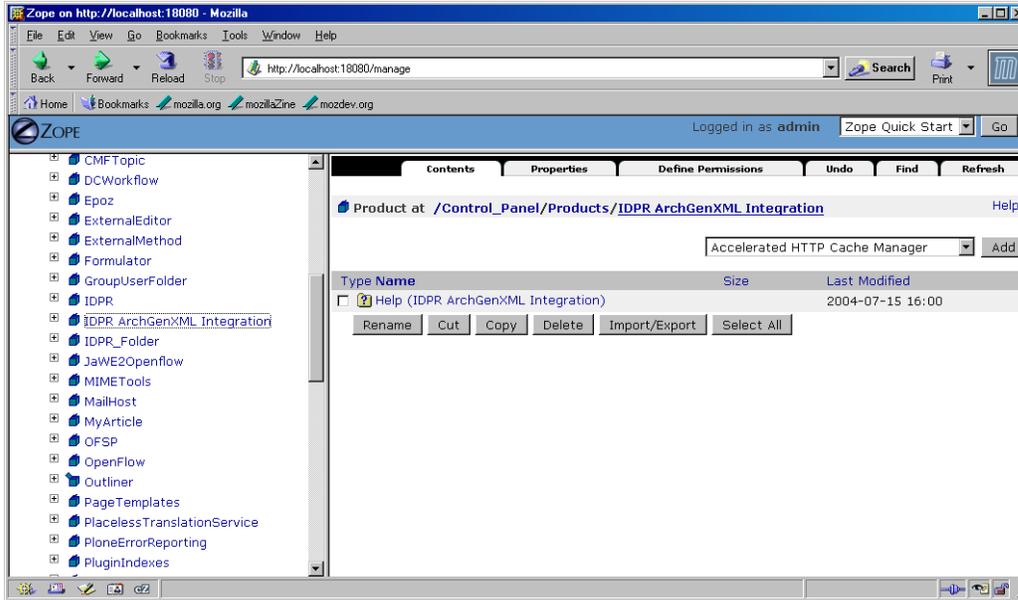
## IDPR ArchGenXML Integration

The Master.schema contains:

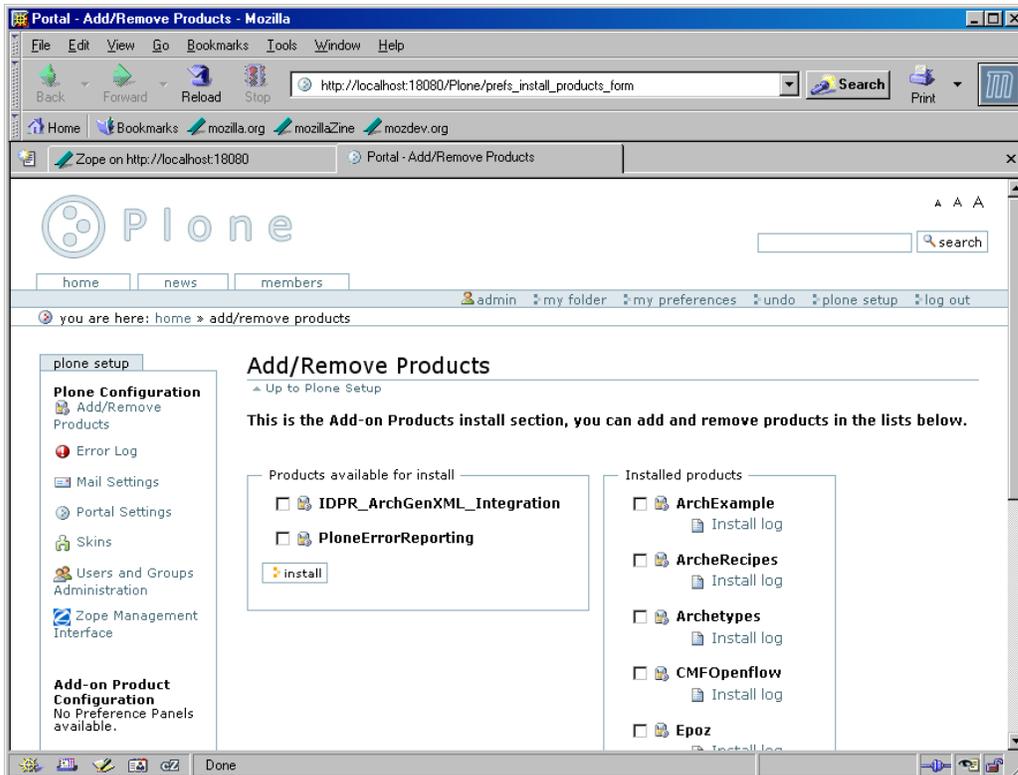
```
schema=BaseSchema + Schema((
  StringField('id',
    widget=StringWidget(description='Enter a value for id.',
      description_msgid='IDPR ArchGenXML Integration_help_id',
      i18n_domain='IDPR ArchGenXML Integration',
      label='Id',
      label_msgid='IDPR ArchGenXML Integration_label_id',
    ),
  ),
  StringField('title',
    widget=StringWidget(description='Enter a value for title.',
      description_msgid='IDPR ArchGenXML Integration_help_title',
      i18n_domain='IDPR ArchGenXML Integration',
      label='Title',
      label_msgid='IDPR ArchGenXML Integration_label_title',
    ),
  ),
  StringField('description',
    widget=StringWidget(description='Enter a value for description.',
      description_msgid='IDPR ArchGenXML Integration_help_description',
      i18n_domain='IDPR ArchGenXML Integration',
      label='Description',
      label_msgid='IDPR ArchGenXML Integration_label_description',
    ),
  ),
  IntegerField('display_priority',
    widget=IntegerWidget(description='Enter a value for display_priority.',
      description_msgid='IDPR ArchGenXML Integration_help_display_priority',
      i18n_domain='IDPR ArchGenXML Integration',
      label='Display_priority',
      label_msgid='IDPR ArchGenXML Integration_label_display_priority',
    ),
  ),
),
)
```

## Plone Products Folder

We can now copy the IDPR ArchGenXML Integration folder to the Plone products directory and restart Zope to see the IDPR\_ArchGenXML\_Integration product in the Products directory:

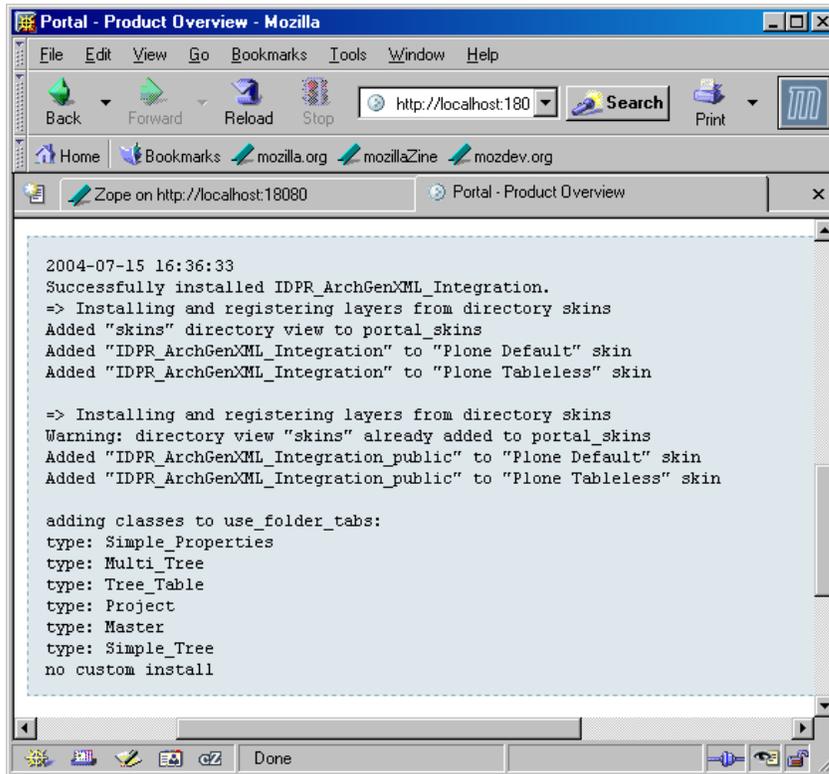


If we go to the Plone setup.Add/remove Products page, we see our new product.



## IDPR ArchGenXML Integration

Installing the product results in the following install log transcript:



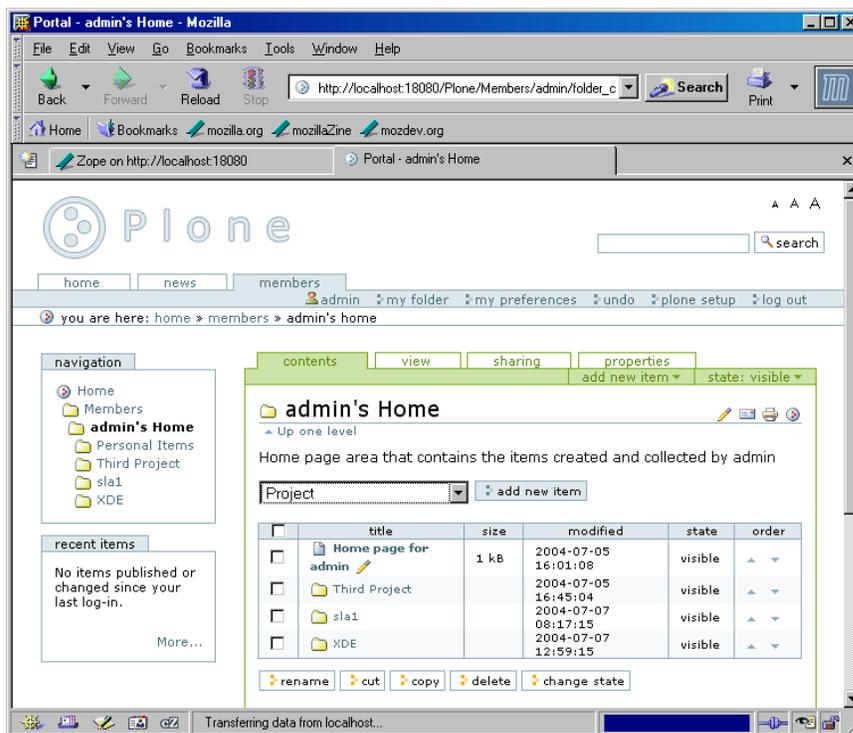
The screenshot shows a Mozilla browser window titled "Portal - Product Overview - Mozilla". The address bar shows "http://localhost:18080". The main content area displays a log transcript with the following text:

```
2004-07-15 16:36:33
Successfully installed IDPR_ArchGenXML_Integration.
=> Installing and registering layers from directory skins
Added "skins" directory view to portal_skins
Added "IDPR_ArchGenXML_Integration" to "Plone Default" skin
Added "IDPR_ArchGenXML_Integration" to "Plone Tableless" skin

=> Installing and registering layers from directory skins
Warning: directory view "skins" already added to portal_skins
Added "IDPR_ArchGenXML_Integration_public" to "Plone Default" skin
Added "IDPR_ArchGenXML_Integration_public" to "Plone Tableless" skin

adding classes to use_folder_tabs:
type: Simple_Properties
type: Multi_Tree
type: Tree_Table
type: Project
type: Master
type: Simple_Tree
no custom install
```

We can add a new Project:



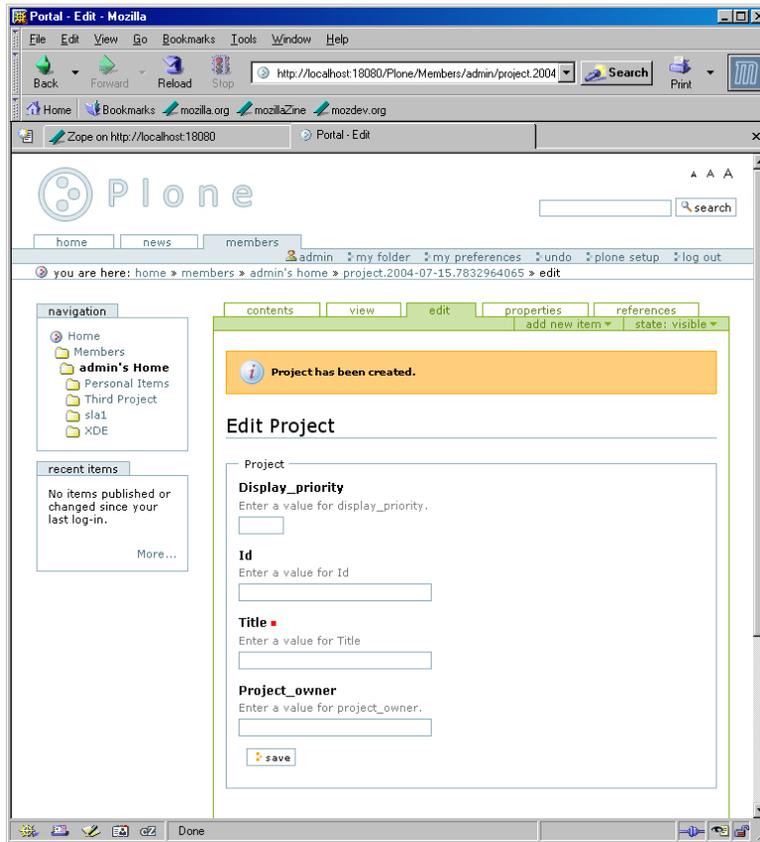
The screenshot shows the Plone web interface in a Mozilla browser window. The address bar shows "http://localhost:18080/Plone/Members/admin/folder\_c". The page title is "Plone". The navigation menu includes "home", "news", and "members". The "members" menu is expanded, showing "admin", "my folder", "my preferences", "undo", "plone setup", and "log out". The "you are here" breadcrumb shows "home > members > admin's home". The main content area shows the "admin's Home" folder with a description: "Home page area that contains the items created and collected by admin". A "Project" item is being added to the folder. The table below shows the list of items in the folder:

	title	size	modified	state	order
<input type="checkbox"/>	Home page for admin	1 kB	2004-07-05 16:01:08	visible	▲ ▼
<input type="checkbox"/>	Third Project		2004-07-05 16:45:04	visible	▲ ▼
<input type="checkbox"/>	sla1		2004-07-07 08:17:15	visible	▲ ▼
<input type="checkbox"/>	XDE		2004-07-07 12:59:15	visible	▲ ▼

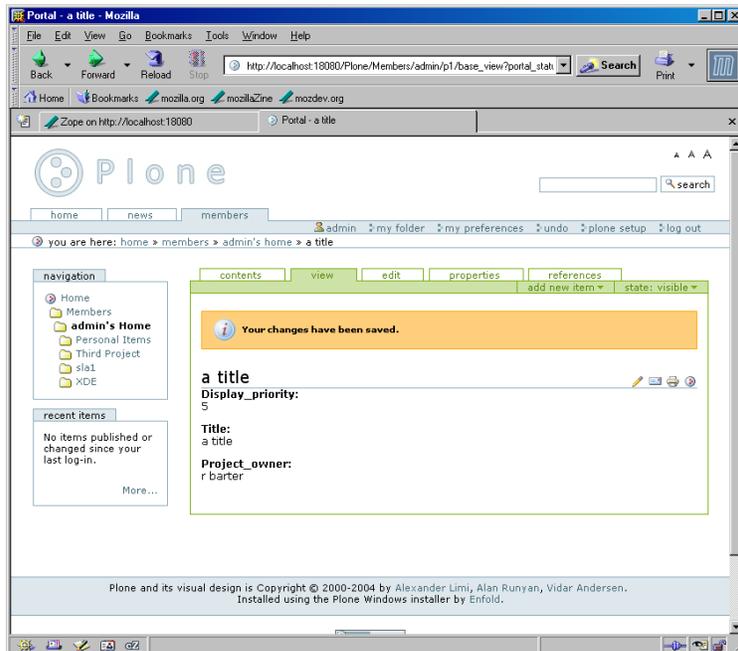
Below the table are buttons for "rename", "cut", "copy", "delete", and "change state".

## IDPR ArchGenXML Integration

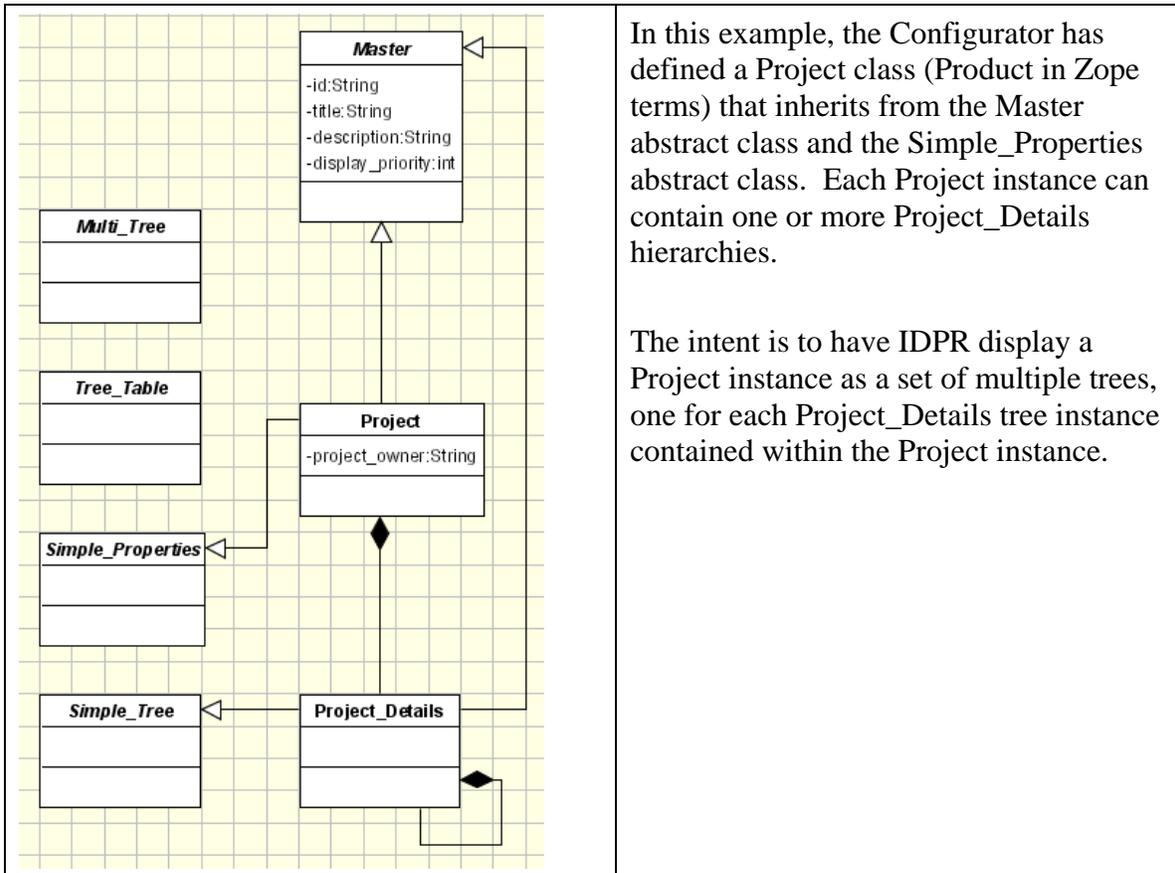
When we add a new instance of the Project product we get the following input screen:



The new Project object has the following default view:



Here is a slightly more interesting example:

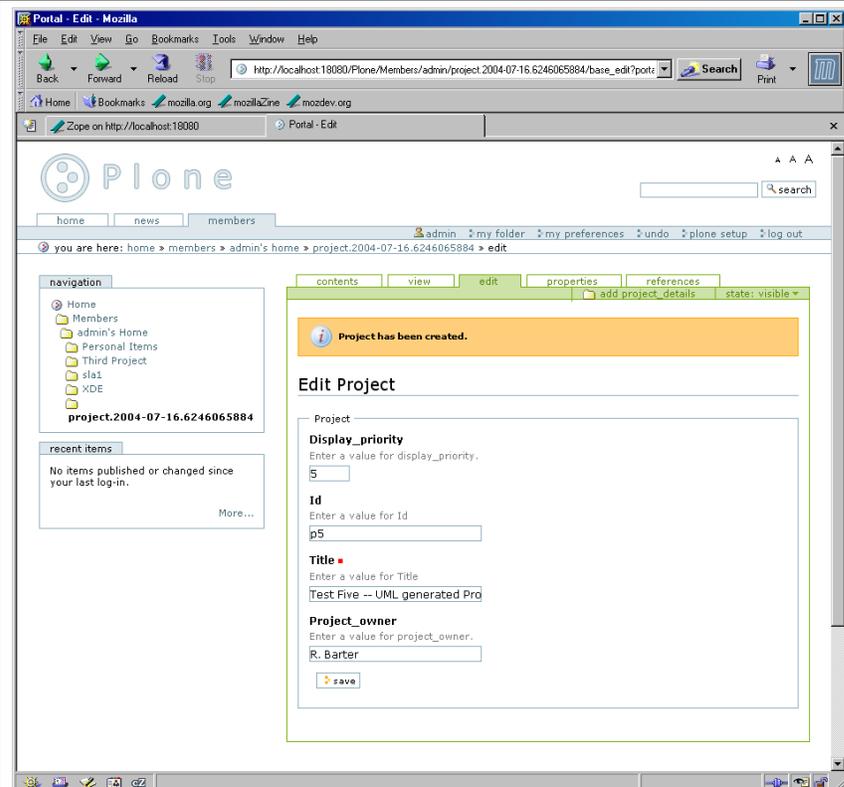


In this example, the Configurator has defined a Project class (Product in Zope terms) that inherits from the Master abstract class and the Simple\_Properties abstract class. Each Project instance can contain one or more Project\_Details hierarchies.

The intent is to have IDPR display a Project instance as a set of multiple trees, one for each Project\_Details tree instance contained within the Project instance.

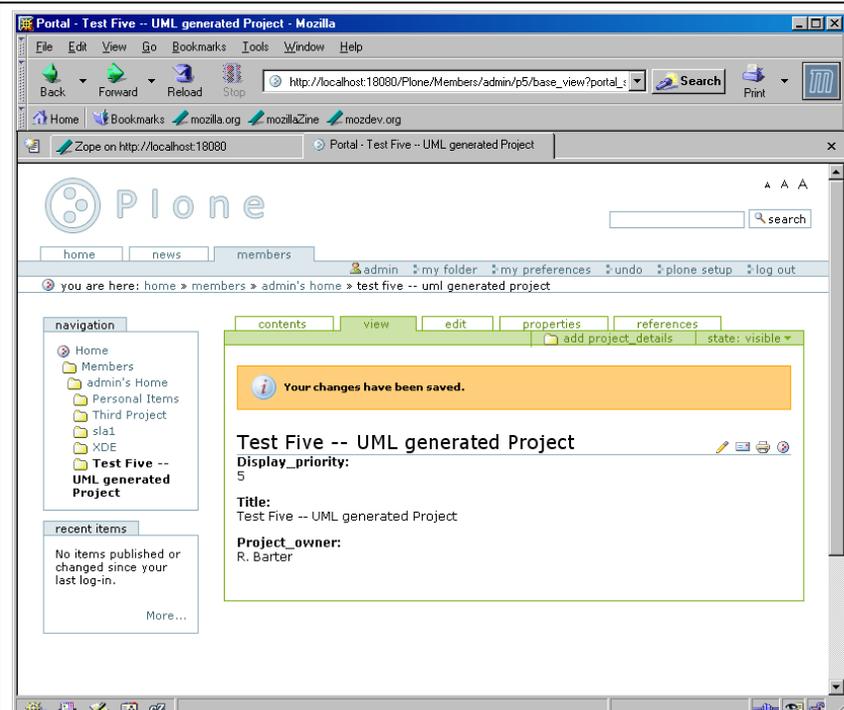
## IDPR ArchGenXML Integration

Now we can add a Project using the new structure and go to the Contents Tab in Plone and see that we can add a Project Detail



The screenshot shows a Mozilla browser window displaying the Plone 'Edit Project' form. The browser's address bar shows the URL: `http://localhost:18080/Plone/Members/admin/project.2004-07-16.6246065884/base_edit?portal...`. The Plone interface includes a navigation sidebar on the left with a tree view showing 'Members', 'admin's Home', 'Personal Items', 'Third Project', 'sls1', 'XDE', and 'project.2004-07-16-6246065884'. The main content area has tabs for 'contents', 'view', 'edit', 'properties', and 'references'. A message box at the top states 'Project has been created.'. Below this is the 'Edit Project' form with fields for 'Display\_priority' (value: 5), 'Id' (value: p5), 'Title' (value: Test Five -- UML generated Pro), and 'Project\_owner' (value: R. Barter). A 'save' button is at the bottom of the form.

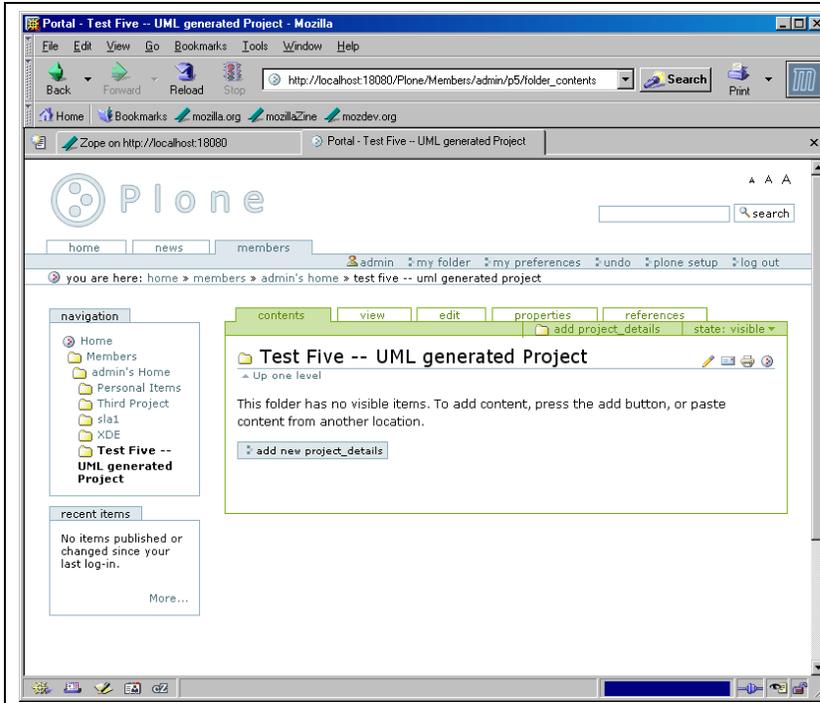
Create a new Project



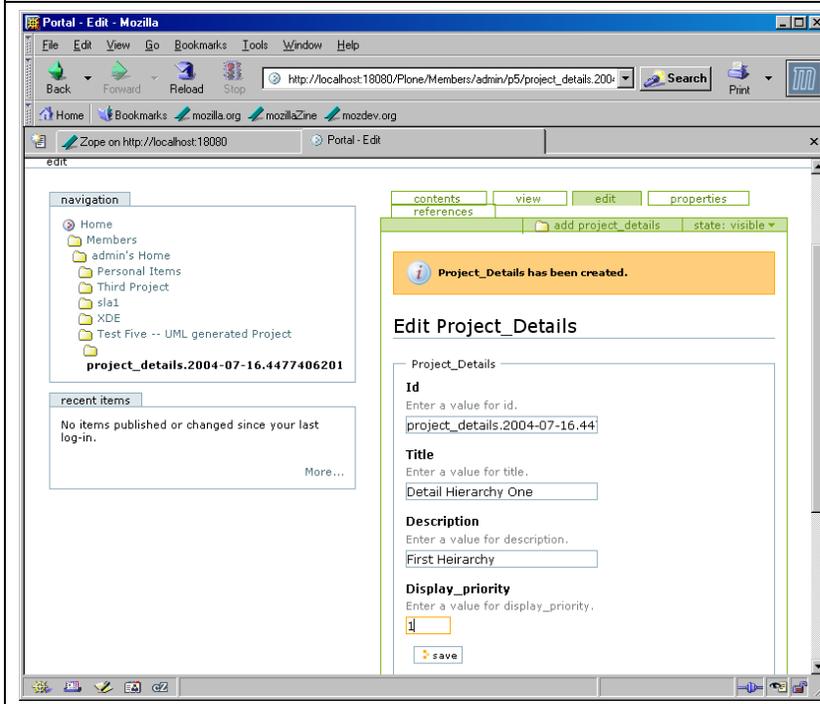
The screenshot shows a Mozilla browser window displaying the Plone 'View Project' page. The browser's address bar shows the URL: `http://localhost:18080/Plone/Members/admin/p5/base_view?portal...`. The Plone interface is similar to the previous screenshot, but the main content area shows the project details: 'Test Five -- UML generated Project', 'Display\_priority: 5', 'Title: Test Five -- UML generated Project', and 'Project\_owner: R. Barter'. A message box at the top states 'Your changes have been saved.'. The navigation sidebar on the left now highlights 'Test Five -- UML generated Project'.

View Project information

# IDPR ArchGenXML Integration

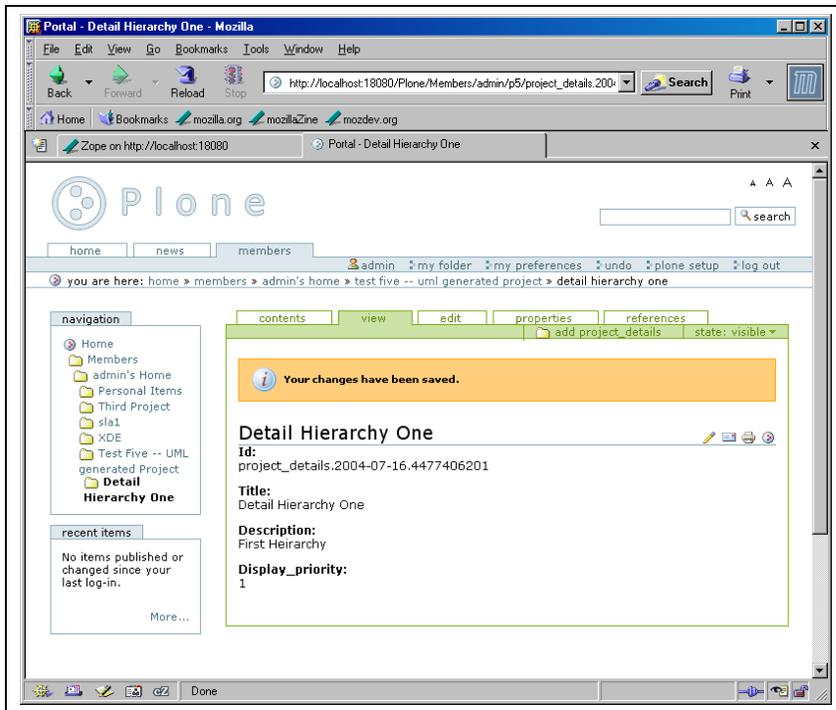


Contents view of Project



Add Project\_Details

## IDPR ArchGenXML Integration



The screenshot shows a Mozilla browser window with the address bar at `http://localhost:18080/Plone/Members/admin/p5/project_details.2004-07-16.4477406201`. The page title is "Portal - Detail Hierarchy One - Mozilla". The main content area displays a message: "Your changes have been saved." followed by the following details:

- Id:** project\_details.2004-07-16.4477406201
- Title:** Detail Hierarchy One
- Description:** First Heirarchy
- Display\_priority:** 1

To the right of the screenshot, the text "View Project\_Detail information" is displayed.

There may be other abstract classes having to do with email notification, data access, or workflow.