Development the OO-Framework for OpenACS:

Improving Scalability and Applicability

Gustaf Neumann • Vienna University of Economics and Business Administration

Summary

Overview

- Summary Improvements to the Framework
- Complex Page Types and Compositions
- Towards an XoWiki based Portal System
### Improvements to the Framework

- Object oriented access/development for all types of acs-objects
- Ability to subclass packages (make packages first class citizens, reusability of package parameters)
- Scalability of property management (package parameters, portal parameters)
- Subclassing of Widgets
- More flexible cluster management

### OO Interface for DB

OO Interface for existing and application specific acs-object-types

Loading schema information from acs-object-types and acs-attributes and create automatically XOTcl classes from this information

Creating new XOTcl Classes as persistent classes using the OpenACS conventions by subtyping existent classes

```tcl
# Fetch base classes from OpenACS Schema
::xo::db::Class get_class_from_db -object_type party
::xo::db::Class get_class_from_db -object_type person

# Create new Object type with two additional attributes
::xo::db::Class create ::demo::employee
   -superclass ::xo::db::person
   -slots {
      ::xo::db::Attribute create salary -datatype integer
      ::xo::db::Attribute create dept_nr -datatype integer -default "0"
   }
```
Class Definition and generated SQL Table

```plaintext
# Create new Object type with two additional attributes
::xo::db::Class create ::demo::employee -superclass ::xo::db::person -slots {
    ::xo::db::Attribute create salary -datatype integer
    ::xo::db::Attribute create dept_nr -datatype integer -default "0"
}

dotlrn-test5=# \d demo_employee

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Modifiers</th>
</tr>
</thead>
<tbody>
<tr>
<td>dept_nr</td>
<td>integer</td>
<td>default 0</td>
</tr>
<tr>
<td>salary</td>
<td>integer</td>
<td></td>
</tr>
<tr>
<td>employee_id</td>
<td>integer</td>
<td>not null</td>
</tr>
</tbody>
</table>

Indexes:
"demo_employee_employee_id_pk"
    PRIMARY KEY, btree (employee_id)

Foreign-key constraints:
"demo_employee_employee_id_fkey"
    FOREIGN KEY (employee_id)
    REFERENCES persons(person_id)
    ON DELETE CASCADE
```

Subclassing Packages

Why subclassing of Packages?

Use Case s5-package:

- Inherits most from XoWiki, some own functionality
- No need to duplicate e.g. package parameter etc.

```plaintext
::xo::db::require package xowiki
::xo::PackageMgr create ::s5::Package \
    -package_key "s5" -pretty_name "S5" \ 
    -superclass ::xowiki::Package
```
Property Management

Property management (package parameters, portal parameters) of OpenACS/DotLRN is wasteful and not sufficiently scaling:

- For every package instance, one package value is copied for every package parameter
- Adding parameters is costly (#packages × #parameters)
- Changing parameter defaults has no effect for existing packages

Package Parameter Redundancy

Real world data (apm_parameter_values):

- Learn@WU: Currently 0.3 mio entries
- Galileo: > 2 mio entries
- Small OpenACS installations: 1000 entries (38 package instances)

High degree of redundancy (many values are stored 4000 with the default value)

Learn@WU: from 300000 entries → only 406 necessary entries needed (non-default)
**OO Package Parameter Interface**

Implementation of base functionality without redundantancy

- `parameter get_from_package_key`: 5x faster
- `parameter get`: 2x faster

Supports inheritance from different packages, uses oacs-datamodel, cluster-safe

Missing:

- user interface, alternate permissions for changing/deleting per-package-instance

---

**Additional Developments**

**Generalized Cluster Management**

- Basic OpenACS functionality: flush util-memoize caches
- New implementation: ability to perform configurable operations clusterwide
- Necessary for e.g. Univ.Valencia to make use of Ajax-based Chat in cluster configurations

**Subclassable Widgets**

- Form-Fields (see XoWiki tutorial)
Improvements in XoWiki

- Initial Microformat support for XoWiki
- Improved composability and personalization via Includelets and Form-fields
- Towards a more powerful portal system

Microformat support in XoWiki

Microformats:

- Make Semantic Web reality
- Microformats define HTML markups for embedding semantic information in HTML pages
- Examples: events, geo-information, relationships, product reviews, ...
- Specialized search engines: e.g. place offers in google-maps, event-summaries, ...
Microformat support in XoWiki

Current support in XoWiki:

- **hCalendar**: announcements of talks and workshops,
  - export e.g. via Firefox extensions *tails*, *operator*, and
  - *iCal* subscription (subscribe to XoWiki instance, similar idea as RSS)
- **rel-tags**: microformat for tags, see e.g. openacs.org/xowiki

Form for Announcement of a Talk
Development the OO-Framework for OpenACS:

Page with embedded Microformats

Export embedded events via tails
**Export embedded events via iCal**

![Subscription of iCal to XoWiki instance]

---

**XoWiki Portal System**

Analysis of the DotLRN Portal System

Rethink Concepts in a bigger picture

- composability
- personalization
- flexibility

Joint work with Michael Aram
Structure of DotLRN Portal System

Types of Portals in DotLRN

2 Types

- User Portal (home)
  Personalization through all memberships

- Community Portals
  - Classes
  - Communities
  - Subgroups

No personalization, Member and Non-member Portals
Structure of the DotLRN Portal System

Shortcomings:

- Rigid 3 level structure (portal-pages are tabs, portal pages have to be in portals, portlets have to be in portal pages, etc.)
- Weak framework integration limits flexibility (e.g. it is not possible to use a "Portlet" on the Start-page, summarizing e.g. contents of two communities)
- Old-fashioned and limited portal page composition (quite complex to add new kinds of portlets, ...)

XoWiki based Portal System (1)

More flexible rendering of base units:

- Separate structure classes ("model" in MVC) from renderers ("views", implemented via mixin classes)
- Provide different kind of renderers:
  - plain
  - mobile
  - css
  - ajax
XoWiki based Portal System (2)

- Provide new composite page types
  - for tabs (similar as portal + portal_pages)
  - and composite pages (with flexible drag&drop interface in Ajax mode; similar as portal_page and portal element)
- Every XoWiki page can be part of a composite page type
- Every includelet, every XoWiki page can be used as a portlet.
- Provide revisions of portals

TabView and Composite Page in XoWiki
XoWiki Page as Portal Page

3 Column Layout, Mashup Elements
Composite Page with different renderer


to use XoWiki portals in DotLRN:

- Define one includelet per DotLRN portlet
- Linkage of portlets to communities:
  - Every dotlrn-includelet has parameter community_id
  - "all" ⇒ use communities, the user is a member
- single or multiple community_ids
- Community_id can be as well provided form the composite page or context
Integration with DotLRN (2)

Consequences:

- Possible to have a portlet with information about
  - the current community,
  - some arbitrary community
  - some arbitrary communities
- Use DotLRN portlets outside DotLRN community pages
  - place e.g. the announcements of some communities to a start-page
- Similar approach with subsites possible.

Community Portal
Personal Portal

Community Portal in Edit-Mode
Community Portal after Drag & Drop

Personalization (1)

**Approach:**

- Tabs and composite pages are implemented via Form-Fields
- Form-Fields can be personalized via Roles
  - Currently defined roles: swa, register_users, unregistered_users admin, creator, app_group_member, community_member
- Foreach role:
  - Possible to show/hide includelets
  - Possible to provide completely different sets of includelets
Personalization (2)

**Consequences:**

Possible to have e.g. different portal pages for some communities

- for not logged in users
- for community admins
- for non-members (of communities or subsites)
- for ...

---

Plans and Ideas for the Portal System

- Developing a recommendation system for includelets
- Learn from Netvibes, iGoogle
- Announce configuration of includelets or let people publish configurations
- Manage context of includelets
- Organization of XoWiki instances in respect to DotLRN communities
- Predefined portals as prototype pages
- Combine User Pages as portlets with Community Pages
Examples

- Portal Page
- DotLRN Club Page
- DotLRN Club Page, edit-mode