# Building Drupal Modules for ODS

# Scope

*Present the development framework of ODS and Provide the elementary guidelines towards building Drupal modules and components for the ODS portal.*

# Syllabus

- Introduction to Drupal.
- ODS Portal Architecture.
- Implementing Drupal Modules for ODS portal.
- Best Practices and Optimizations.

# Introduction to Drupal

# What is Drupal?

## Content Management System

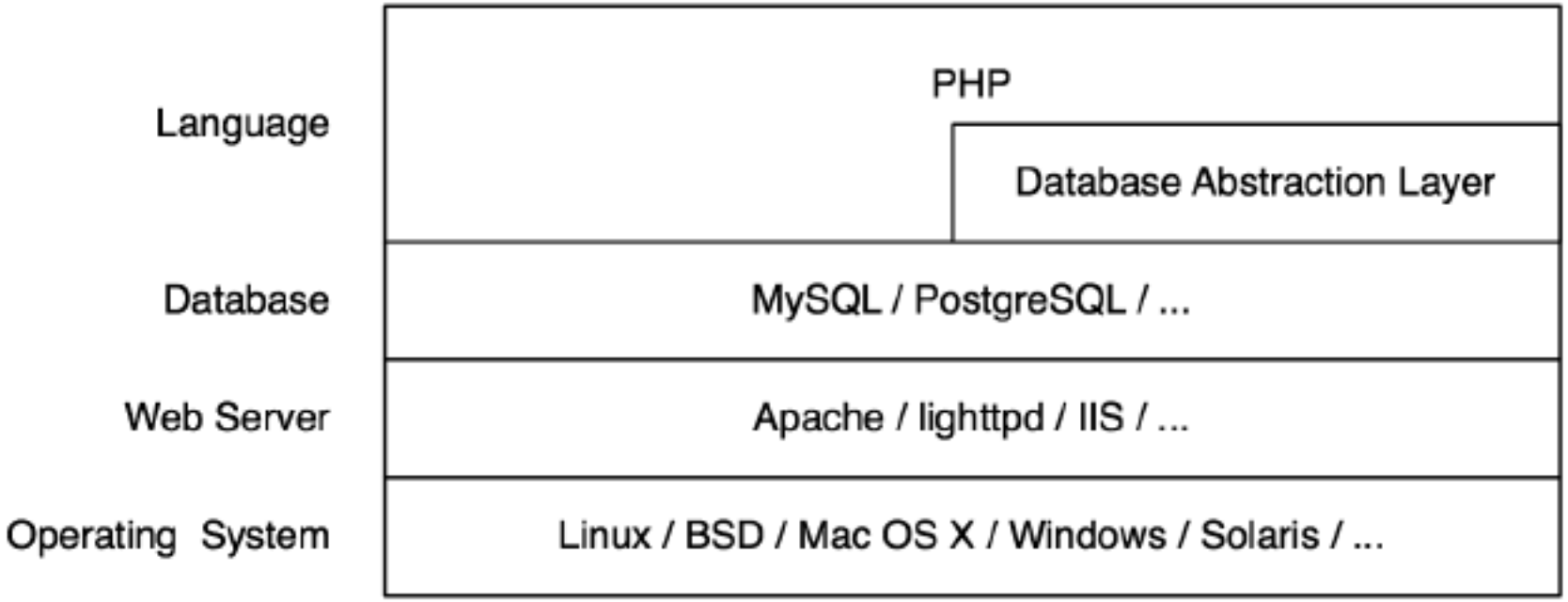
- Anonymous users navigate through content.
- Administrators manage configuration settings and users/role permissions.
- Editors manage content.
- Enhanced automated functionality.

## Content Management Framework

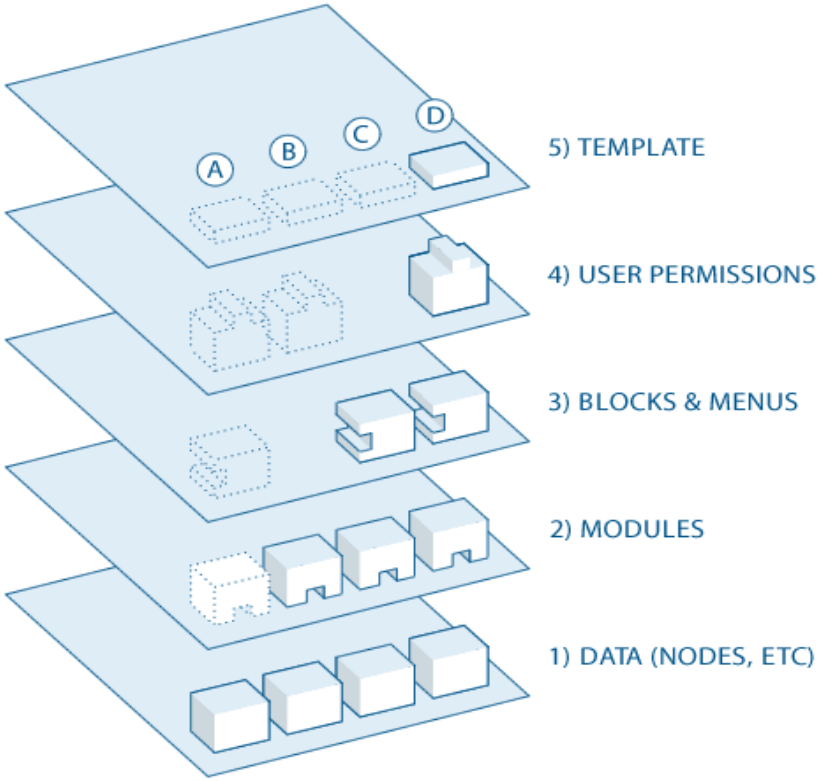
- Extend capabilities with modules/themes.
- Intercept and transform all aspects of Drupal behavior.

*“A sophisticated web application building tool.”*

# Drupal Technology Stack



# Drupal Architecture




1. Data pool: Collection of data (nodes, fields, users, blocks, etc) stored in the employed database management system (MySql). (https://drupal.org/files/er_db_schema_drupal_7.png)
2. Modules: Functional plugins extending or altering on Drupal's core functionality.
3. Blocks & Menus:
   - Blocks: Structural viewable elements places in various positions defined by the template (theme).
   - Menus: Manages the content that should be displayed at each defined menu path.
4. User Permissions:
   - What different kind of users are allowed to do and see.
5. Template/Theme: Define the "skin" of the Drupal generated content. Prescribes the position of each element and the CSS style that must be used.

# Drupal: Handling requests

- Visit: http://example.com/node/123.
- PHP execute Drupal's index.php to handle /node/123.
- Bootstrap:
  - Is executed from Core to initialize resources.
  - Invokes menu system to explore how to handle /node/123.
- Node System:
  - Loads the content associated with the ID 123 from the database.
- Theme System:
  - Applies formatting and style to the node information.
- Drupal Core:
  - Completes processing and returns data to the client.
- The browser transforms HTML and CSS to visual representation running involved JavaScipts along the way.

Custom functionality can be injected in this process using Drupal *"hooks"*.

# Drupal: Core

- Provides the basic drupal functionality:
  - Content management
  - User management
  - Session management
  - Url aliases
  - Templating
  - Localization
  - Logging
  - Library of common functions
- Allows Drupal to bootstrap when receiving a request.

# Drupal: Bootstrap Process

- Executed on every page request.
- Phases
  - Configuration: Sets global variables used in the bootstrap process.
  - Database: Initialization and registration of autoload functions.
  - Variables & Modules: Loads all system variables and all enabled bootstrap modules.
  - Session: Session management initialization.
  - Phase Header: Invokes hook_boot(), initializes locking system and sends the HTTP headers.
  - Language: Initializes defined language types.
  - Full: Validates and fixes input data.

# Drupal Modules

- Drupal is a modular framework.
- A module is software (code) that extends Drupal functionality.
- Module include sources to enhance/alter Drupal's core functionality.
- Modules Categories:
  - **Core** modules are those included with the main download of Drupal. (i.e. *Blog, Book, Poll*, or *Taxonomy)*.
  - **Contributed** modules are downloaded from the Modules download section of drupal.org, and installed within your Drupal installation.
  - **Custom** modules are created by us using PHP and the Drupal API.
- Collections of hook implementations.

# Drupal Hooks

- The way for modules to interact with Drupal Core.
- Hooks occur at various points in the thread of execution, where Drupal seeks contributions from all the enabled modules.
- 342 available hooks.
- A hook can be thought of as an event listener in the sense that an event triggers an action. The event in Drupal, such as deleting a node, would trigger the hook "hook_delete". If your module implemented hook_delete, that function would run when a node deletion occurred

# Drupal Nodes & Fields

- Nodes:
  - The heart of Drupal design.
  - Each item of content is stored as a node.
  - Structured elements composed of fields.
- All content types in Drupal are derived from the base type “Node”.
- Only a few things aren't nodes – users, groups, modules, and themes being the main ones.
- Other stuff, from calendar events, to RSS feed items, to page content is a node.

# Drupal Themes

- The presentation layer.
- Control the appearance (look and feel).
- Themes create the HTML (or Json or XML) that the browser will receive.
- Themes take over the output of content displayed from the database.

# Drupal Taxonomies

- A system for classifying content.
- Provides metadata information about nodes.
- Taxonomy has vocabularies
  - Vocabularies are lists of terms.
  - Relations can be added between terms usually hierarchical.

# Drupal Users

- Every visitor to the site, (both logged in and anonymous), is considered a Drupal user.
- Each user has a numeric user ID
  - Anonymous User ID = 0
  - Superuser ID = 1
  - Other users ID > 1
- Non-anonymous users also have a associated fields.

# Drupal Content Display

- Views: an interface for making customized lists of the data contained in the Drupal database.
- Panels: an interface for making customized layouts of nodes available to the panels module.
- Widgets: a general term for interactive form elements or graphs that are enabled by modules.

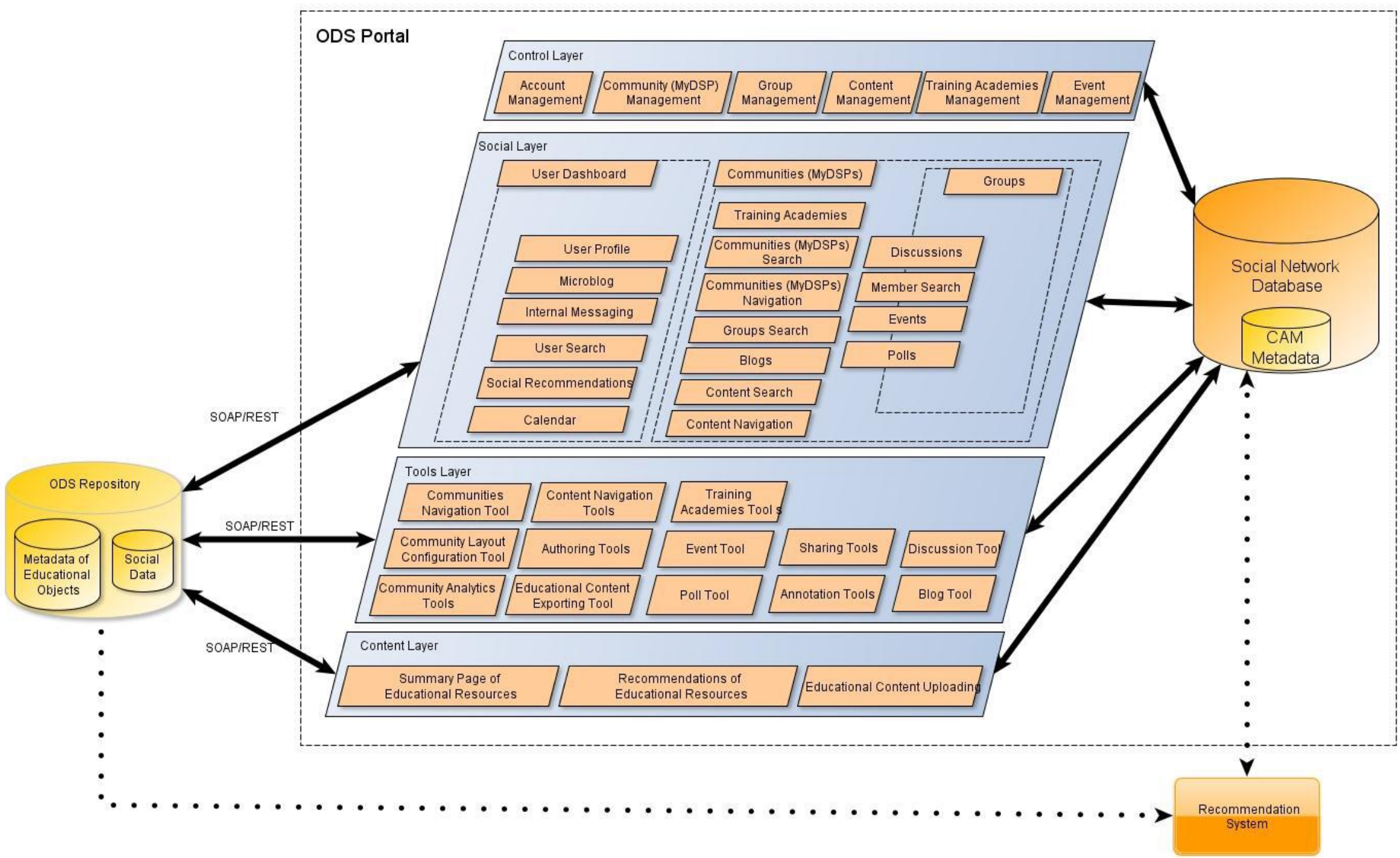
# Drupal APIs

- Module system (Drupal hooks)
- Database abstraction layer (Schema API)
- Menu system
- Form generation
- File upload system
- Field API
- Search system
- Node access system
- Theme system

# ODS Portal Architecture

# Before Building ODS

- Define content types and fields.
- Who/how inputs data in the portal?
- How will the data get out (views, panels, blocks, quicktabs, etc)?
- Who/how can create communities/groups and content in them?
- How to register in ODS?
- How to add educational resources?
- How to classify content and search efficiently?
- Theming.


ODS Portal
Control Layer
Account Management
Community (MyDSP) Management
Group Management
Content Management
Training Academies Management
Event Management
Social Layer
User Dashboard
Communities (MyDSPs)
Groups
Training Academies
User Profile
Communities (MyDSPs) Search
Discussions
Microblog
Communities (MyDSPs) Navigation
Member Search
Internal Messaging
Groups Search
Events
User Search
Blogs
Polls
Social Recommendations
Content Search
Calendar
Content Navigation
Social Network Database
CAM Metadata
SOAP/REST
ODS Repository
Metadata of Educational Objects
Social Data
SOAP/REST
Tools Layer
Communities Navigation Tool
Content Navigation Tools
Training Academies Tool s
Community Layout Configuration Tool
Authoring Tools
Event Tool
Sharing Tools
Discussion Tool
Community Analytics Tools
Educational Content Exporting Tool
Poll Tool
Annotation Tools
Blog Tool
SOAP/REST
Content Layer
Summary Page of Educational Resources
Recommendations of Educational Resources
Educational Content Uploading
Recommendation System

# Control Layer

- Educational and social content management.
  - Account management
  - Community management
  - Group management
  - Content management
  - Training academies management
  - Event management

# Social Layer

- Manages user interactions and direct communications.
- Provides tools for building communities and groups, organizing events, creating polls and perform training activities.

**Components:**

- User Dashboard
- User Profile
- Microblogging
- Internal messaging
- User Search
- Calendar
- Community management
- Group Search
- Blogs
- Discussions
- Polls
- Events

# Tools Layer

- Provides modules that enhance the functionalities of the social network:
  - Communities navigation tool
  - Community layout configuration tool
  - Community analytics tool
  - Content navigation tools
  - Authoring tools
  - Educational Content Exporting Tool
  - Training Academies Tools
  - Event Tool
  - Poll Tool
  - Sharing Tools:
  - Annotation Tools
  - Discusssion Tools
  - Blog tool

# Content Layer

- Handles the presentation of the educational resources (EC, LAs and LSs), their metadata and social data, as well as the uploading of the educational resources and the content recommendations.
  - Summary Page of Educational Resources
  - Recommendations of Educational Resources
  - Educational Resource Uploading

# ODS Content Types

- Every node belongs to a single “node type” or “content type”.
- Every “content type” defines various default settings for nodes of that type, such as whether the node is published automatically and whether comments are permitted.
- Content types have different fields and custom modules may define their own content types.
- ODS Content Types:
  - http://10.240.41.41/~devel/beta/admin/structure/types

# ODS Provided & Custom Modules

- http://portal.opendiscoveryspace.eu/beta/admin/modules

# ODS Content Organization

- Content Types:
  - Activity
  - Community
  - Group
  - Discussion
  - Event
  - Group
  - Poll
  - School
  - Training Academy
  - Training Activity
  - Educational Object
- Roles:
  - Administrator
  - Teacher
  - Parent
  - Expert
  - Translator
  - Template Editor
  - Training Activity Contributor
- Taxonomies:
  - ODS AP Vocabularies
  - Tags
  - Repositories
  - Content related vocabs

# ODS Structural Elements for Content Display

- Use custom pages with panels to display content.
  - Views + Quicktabs + blocks + minipanels, organized in custom pages through panels.
- http://portal.opendiscoveryspace.eu/beta/admin/structure/pages/edit/node_view

# Principles followed during development

- Look if there is a module providing the desired functionality.
  - If yes configure it for our case.
  - If no develop a custom module for the functionality.
- Create MySql tables using schema API, only if you can’t store the information in existing fields.
- Follow the Form API and strictly employ the DB abstraction API.
- Minimize code additions in panels and views.
- Every natural language string that may be displayed to a user should be wrapped in the t() function.
  - t() is responsible for translating strings from one language to another.
- Always validate form data before submission

# Application Profile

*“An Application Profile (AP) is a metadata scheme, which consists of metadata elements selected from one or more standard metadata schemes and it is created for allowing a given application to meet its functional requirements.”*

*“An assemblage of metadata elements selected from one or more metadata schemas and combined in a compound schema. Application profiles provide the means to express principles of modularity and extensibility. The purpose of an Application Profile is to adapt or combine existing schemas into a package that is tailored to the functional requirements of a particular application, while retaining interoperability with the original base schemas”*

# ODS LOM Application Profile

- IEEE LOM Application Profile
- Assists metadata annotation of LO for enhancing search and retrieval process.

# ODS LOM Application Profile

- Mandatory Elements
  - ***General.Title:*** Human readable name of the LO.
  - ***Technical.Location:*** URL/URI of the resource.
  - ***Classification. Purpose:*** the purpose of classifying a learning object under a given classification system.
  - ***Classification.Taxon Path:*** a taxonomic path in a specific classification system.
    - ***Source:*** The name of the classification system.
    - ***Taxon:*** a particular term in a given taxonomy.
    - ***Id:*** the taxon identifier.
    - ***Entry:*** The taxon textual label.

# ODS LOM Application Profile

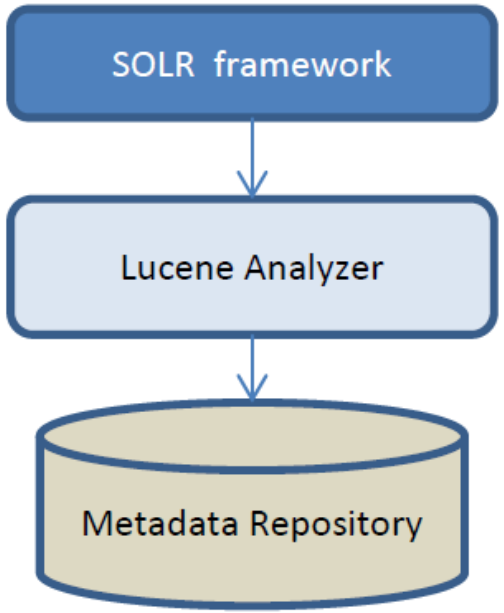
- Recommended elements
  - General.Identifier
  - General.Keyword
  - General.Aggregation Level
  - Educational Context
  - Educational.Typical Age Range
  - Educational.Difficulty
  - Educational.Typical Learning Time
  - General Description
  - LifeCycle.Contribute
  - Educational.Learning Resource Type
  - Meta-Metadata.Identifier
  - Meta-Metadata.Contribute
  - Rights.Cost
  - Rights.Copyright
  - Other.Restrictions
  - Rights.Description

# The educational Object content type

| Educational Object Field | Machine name | Type | Widget | ODS-AP XPath |
|---|---|---|---|---|
| **Title** | title_field | Text | Text field | /lom/general/title |
| **Author Fullname** | field_author_fullname | Text | Text field | /lom/lifeCycle/contribute/entity |
| **Author Organization** | field_author_organization | Text | Text field | |
| **Body** | body | Text | Text area with a summary | /lom/general/description |
| **LO Identifier** | field_lo_identifier | Text with summary | Text field | /lom/general/identifier/entry |
| **Language** | language | language selection | | /lom/general/language |
| **Resource Link** | field_resource_link | link | link | /lom/technical/location |
| **Social Taggers** | field_social_taggers | Entity Reference | Autocomplete (Tags style) | |
| **URL path settings** | path | Path module form elements | | |
| **XML sitemap** | xmlsitemap | XML sitemap module element | | |
| **Educational TypicalAgeRange** | field_educational_typicalagerang | Text | Text field | /lom/educational/typicalAgeRange |
| **Copyright** | field_copyright | Text | Text field | /lom/rights/copyrightAndOtherRestrictions |
| **Classification TaxonPath** | field_classification_taxonpath | Text | Text field | /lom/classification (Extended) |
| **Classification term** | field_classification_taxonpath_discipline | Term reference | | |
| **Data Provider** | field_data_provider | Term reference | Select list | |
| **Educational Context** | field_educational_context | Term reference | Select list | /lom/educational/context |
| **Edu Tags** | field_edu_tags | Term reference | Autocomplete term widget (tagging) | /lom/general/keyword |
| **Update Date** | field_update_date | Date | Pop-up calendar | /lom/lifeCycle/contribute/date |
| **Groups audience** | og_group_ref | Entity Reference | OG reference | |
| **ODS Object** | field_ods_object | Boolean | Single on/off checkbox | |
| **Vtab Group** | group_vtab_group | | | |
| **Content** | group_content | | | |
| **Wrapper** | group_wrapper | | | |
| **Object Link** | field_eo_link | Link | Link | |
| **Object File** | field_eo_file | File | File | |
| **Guided Tags** | group_guided_tags | | | |
| **Educational Context** | field_gt_educational_context | Term reference | Select list | /lom/educational/context/value |
| **Educational Objectives** | group_educational_objectives | | | /lom/classification/taxonpath/taxon/entry |
| **Affective** | field_gt_affective | Term reference | Select list | |
| **Cognitive Knowledge** | field_gt_cognitive_knowledge | Term reference | Select list | |
| **Cognitive Process** | field_gt_cognitive_process | Term reference | Select list | |
| **Psychomotor** | field_gt_psychomotor | Term reference | Select list | |
| **XML File Path** | field_xml_path | Text | Text field | |
| **Form State** | field_form_state | Long text | Text area (multiple rows) | |
| **Aggregation Level** | field_aggregation_level | int | Text field | /lom/general/aggregationlevel/value |
| **Cost** | field_rights_cost | int | Text field | /lom/rights/cost/value |

# ODS Search/Indexing

- Apache Solr
- Indexes all the elements of the ODS LOM AP enabling fast search on top of them.

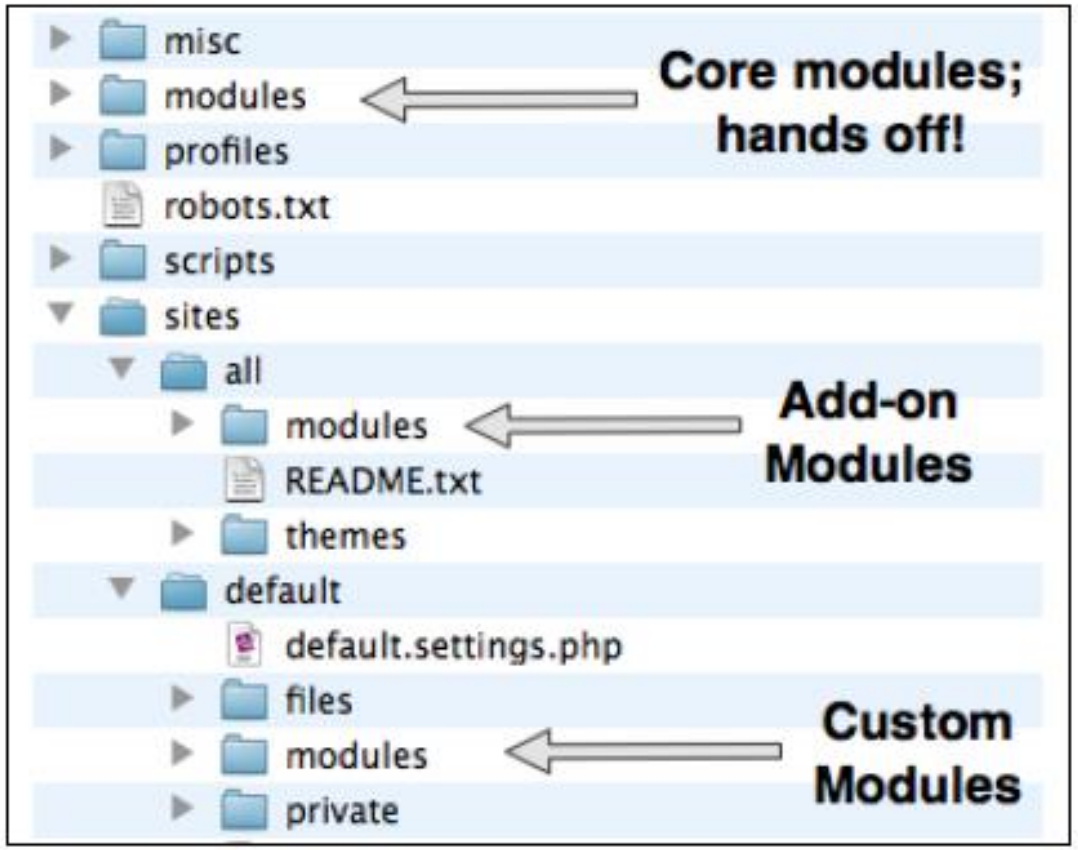


# Implementing Drupal Modules for ODS

# How modules work

- A collection of php files designed to implement some find of functionality
  - /sites/default/modules
  - ods_custom_module.info
  - ods_custom_module.install
  - ods_custom_module.module

# Step 1: Creating the files




- All Custom modules in sites/default/modules
- Naming convention: ods_custom_module
- Creating the following folder:
  - sites/default/modules/ods_custom_module
- Create a file named ods_custom_module.info
  - name = ODS Custom Module
  - description = A custom ods module
  - core = 7.x
  - package = ODS
  - configure = admin/config/content/cm
  - dependencies[] = views
- Create the PHP file ods_custom_module.module
  - Opening tag <?php

- Create the ods_custom_module.install file
  - Handles operation during installing/uninstalling a module
  - Uses the Drupal Schema Api to define database tables.

# Step 2: Identify the hooks

- hook_help
- hook_menu()
- hook_permission()
- hook_form_alter()
  - hook_form_FORM_ID_alter()
- hook_mail()
  - hook_mail_alter()
- hook_node_load()
- hook_node_submit()
- hook_user_insert()
- hook_install()
- hook_schema()

https://api.drupal.org/api/drupal/includes!module.inc/group/hooks/7

# Step 3: Menu System

- Navigation system from a user perspective
- Callback system used to respond to URLs passed from the browser.
- Follows a simple hierarchy defined by paths.
- Implementations of hook_menu() define menu items and assign them to paths (which should be unique).
- Each path is matched with php code to be executed.

# Step 3: Menu System – hook_menu

```
function ods_content_update_notifications_menu() {
  $items = array();
  $items['admin/config/system/ods_content_update_notifications'] = array(
    'title' => 'ODS Content Update Notifications',
    'description' => 'Configure ODS Content Update Notifications',
    'page callback' => 'drupal_get_form',
    'page arguments' => array('ods_content_update_notifications_form'),
    'access arguments' => array('administer modules'),
    'type' => MENU_NORMAL_ITEM,
  );
  return $items;
}
```

# Step 4: Form API

- This system provides a robust programmatic tool for defining, displaying, validating, and submitting forms.
- https://api.drupal.org/api/drupal/developer!topics!forms_api_reference.html/7
- https://www.lullabot.com/files/formapi.pdf

# Step 5: Database Abstraction Layer

- Allows the use of different database servers using the same code base.
- Removes the complexities of interacting with a database.
- Based on PHP's Data Object (PDO) library
- Keeps code from being tied to a single database.
- Sanitizes user-submitted data placed into queries to prevent SQL injection attacks.
- http://wizzlern.nl/sites/wizzlern.nl/files/11/apr/drupal7db.pdf

# Best Practices and Optimizations

# Best Practices

- Avoid hacking core.
- Before modifying an existing module check if you can hook into the functionality.
- Use coding standards http://drupal.org/coding-standards
- Create backup before modifying.
- Use check functions on output to prevent cross site scripting attacks
- Use the database abstraction layer to avoid SQL injection attacks

# Performance Optimization

- ODS uses APC (Alternative PHP Cache) for caching and optimizing php intermediate code.
- ODS uses Memcache for object caching.
- Caching static blocks and views.