SeDiCI
SERVICIO DE DIFUSIÓN
DE LA CREACIÓN INTELECTUAL

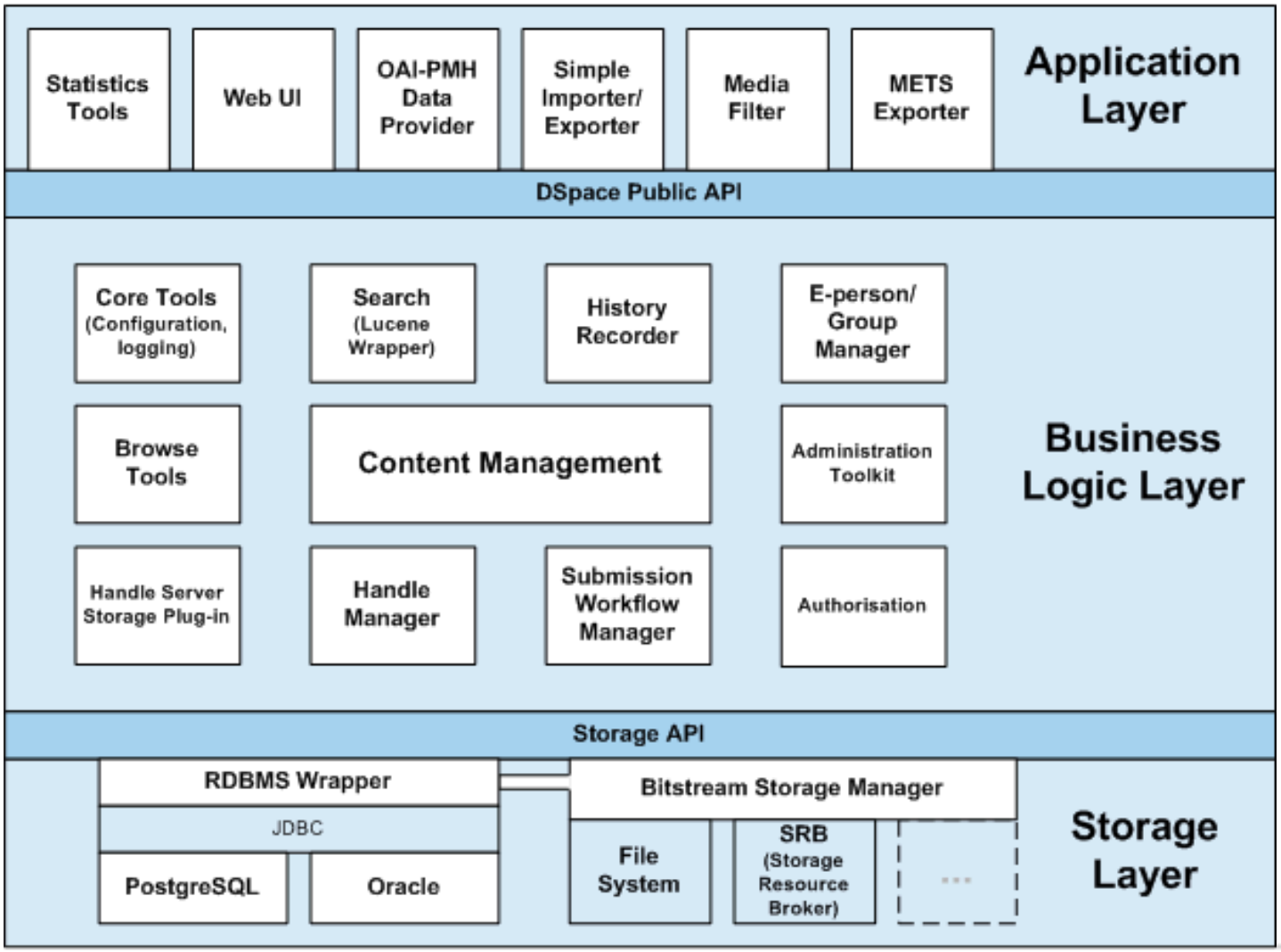
Aspectos técnicos de DSpace

Bloque 2.1: contenido

- Arquitectura
- Modelo de datos
- Usuarios y autorizaciones
- Workflow y workflow configurable
- Discovery y Apache Solr
- Estructura del proyecto
- Maven como gestor de dependencias
- Introducción a XMLUI
  - DRI, Cocoon, Temas, Aspectos

Arquitectura de DSpace

Arquitectura de DSpace



Arquitectura de DSpace

Cada capa ofrece servicios a la capa superior por medio de APIs, y utiliza los servicios de la capa inferior

El código fuente se organiza en paquetes que representan esta arquitectura en capas:

- org.dspace.app — Capa de aplicación
- org.dspace — Capa de lógica del negocio
- org.dspace.storage — Capa de almacenamiento

Arquitectura de DSpace
Capa de almacenamiento

Interacción con la base de datos

- Items y sus metadatos
- Personas y grupos
- Información de autorización
- Trabajos en curso (workflow)
- Indices de búsqueda y exploración

Arquitectura de DSpace
Capa de almacenamiento

Almacenamiento de bitstreams

- Local: el almacenamiento se realiza en el sistema de archivos local al servidor en el que funciona la aplicación

- Storage Resource Broker (SRB): permite tener un sistema de archivos distribuido

Arquitectura de DSpace
Capa de lógica de negocios

Ofrece

- Administración
- Búsqueda
- Exploración
- Gestión de usuarios y grupos
- Autorización
- Carga de documentos
- Workflow
- Handle manager
- Abstracción en Comunidades, Colecciones e Items

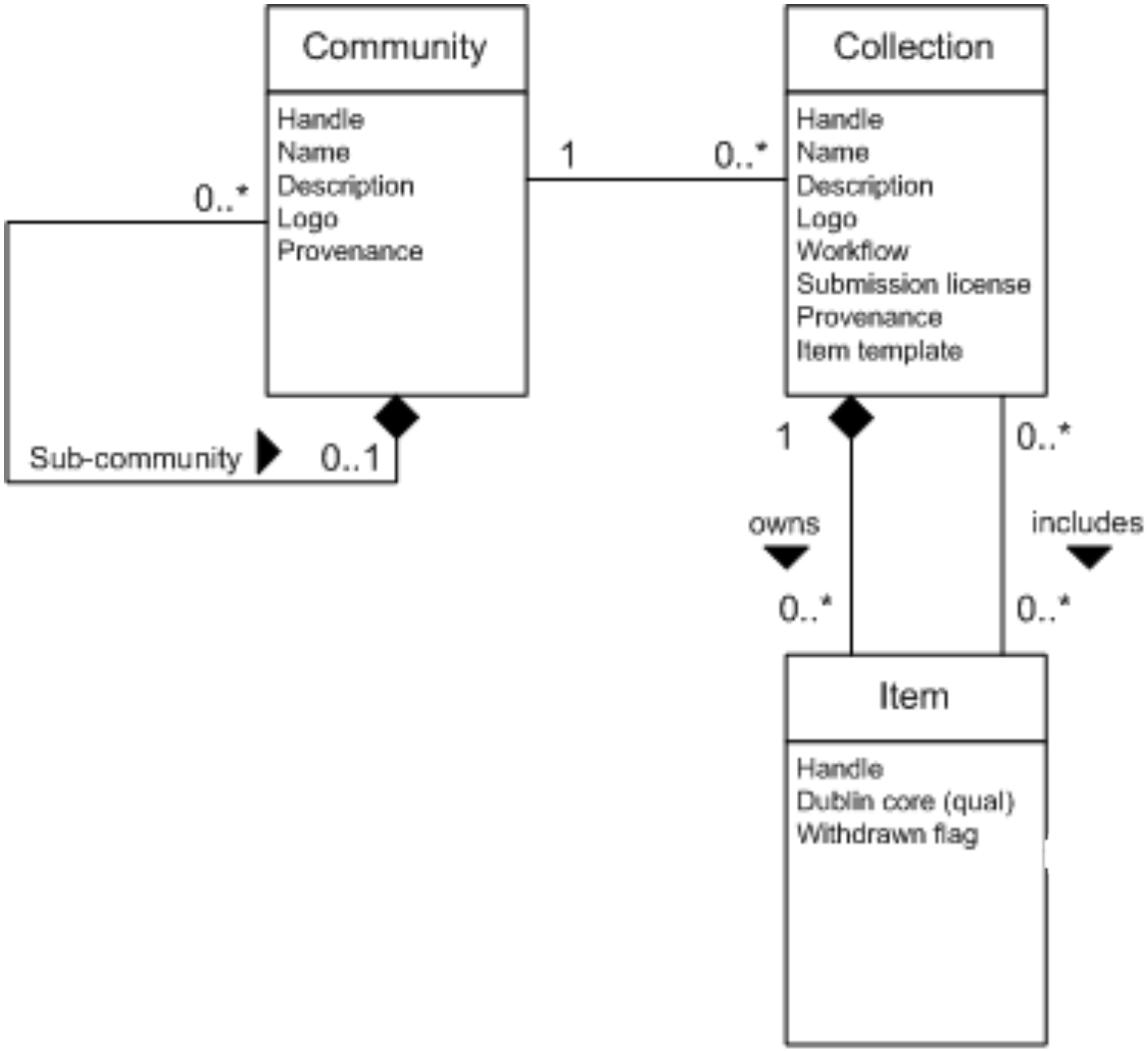
Arquitectura de DSpace
Capa de aplicación

Conjunto de módulos que permiten la interacción con el mundo exterior

- Aplicación web: JSPUI y XMLUI
- OAI-PMH Data Provider
- Estadísticas
- Importar/Exportar
- MediaFilter

Modelo de datos

Modelo de datos
Organización de contenidos



Modelo de datos
Organización de contenidos

Características:

- Las comunidades pueden contener sub-comunidades o colecciones, pero no ítems
- Las colecciones sólo pueden contener ítems
- Un ítem pertenece a una sola colección, pero puede estar asociado a otras colecciones

Modelo de datos
Organización de contenidos

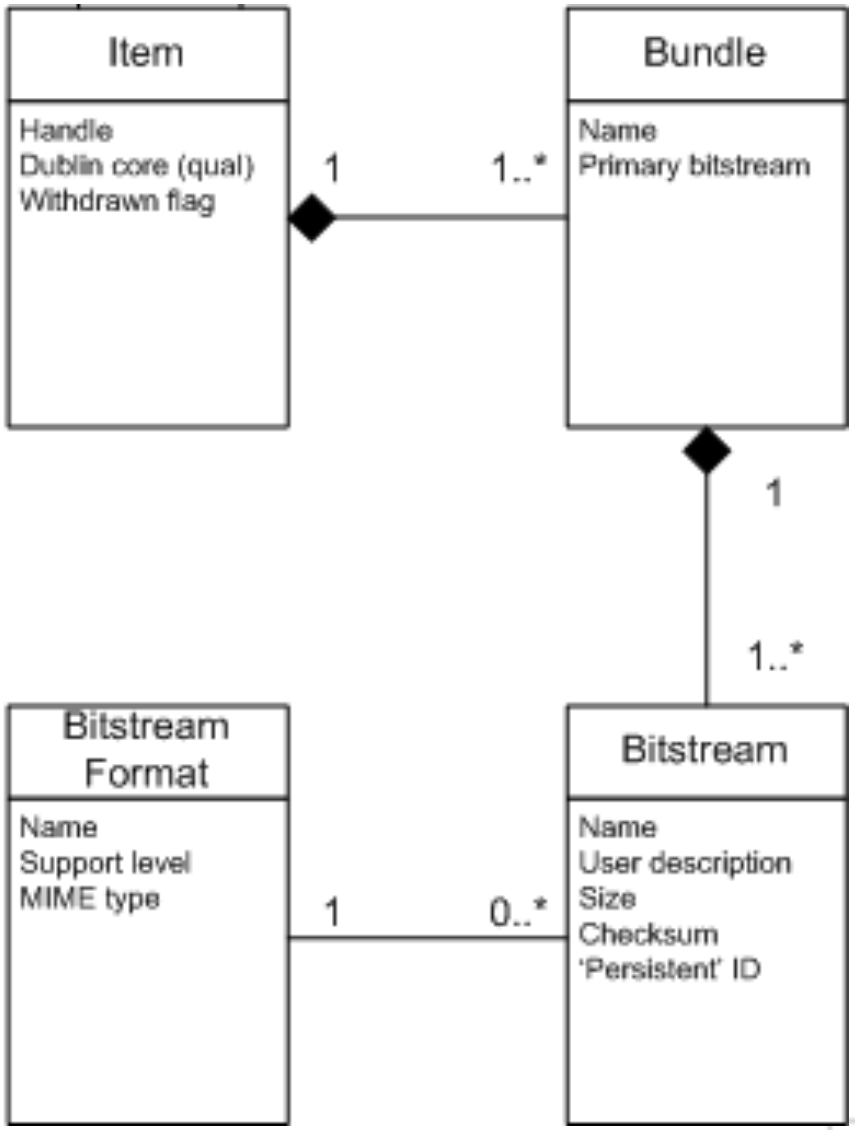
Ventajas:

- Permite establecer restricciones de acceso y modificación específicos para cada nivel
- Permite plantear un esquema navegacional de forma simple

Desventajas:

- Obliga a crear y mantener una estructura estática
- Tiende a generar estructuras redundantes

Modelo de datos Archivos



Modelo de datos
Archivos

Los archivos se representan como Bitstreams

Los Bitstream contienen información de validación (checksum), descriptiva, y de preservación (formato y nivel de soporte del formato)

Modelo de datos
Archivos

Los archivos se agrupan en Bundles, según su naturaleza:

ORIGINAL: archivos originales subidos por el usuario

TEXT: archivos de texto extraído de forma automática a partir de los archivos cargados en el Bundle ORIGINAL

LICENSE: archivos de licencia asociados al ítem

Usuarios y autorizaciones

Gestión de usuarios

- Denominados **E-Person**
- Agrupados en **E-Group**

Un E-Group puede contener:

- múltiples E-Person
- otros E-Group

Un E-Person puede pertenecer a muchos grupos (directa o indirectamente)

Gestión de autorizaciones

Listado de derechos con tres elementos (**Objeto**, Usuario, Derecho)

Objeto puede ser:

- Comunidad
- Colección
- Item
- Bitstream

Gestión de autorizaciones

Listado de derechos con tres elementos (Objeto, **Usuario**, Derecho)

Usuario puede ser:

- E-Person
- E-Group

Gestión de autorizaciones

Listado de derechos con tres elementos (Objeto, Usuario, **Derecho**)

Derecho puede ser:

- READ: ver o descargar
- WRITE: modificar datos
- ADD: agregar a un contenedor
- REMOVE: eliminar de un contenedor
- ADMIN: administración de elementos

Gestión de autorizaciones

- Existen dos grupos del sistema: **Anonymous**, **Administrator**
- Todos los usuarios pertenecen al grupo **Anonymous**
- Por defecto, todas las comunidades, colecciones e ítems tienen permiso de **READ** para el grupo **Anonymous**
- Existen **Derechos** especiales para el **Workflow** (no XML)

Roles de usuario

- **Administradores**: tiene control sobre el elemento que administra
  - Administrador del sitio
  - Administrador de comunidad
  - Administrador de colección

- **Revisores**: usuarios asociados a algún paso del workflow de revisión

Roles de usuario

- **Submitters**: pueden realizar envíos de documentos en alguna colección

- **Anónimos**: tienen acceso de sólo lectura (incluye a los usuarios registrados que sólo pertenecen al grupo Anonymous)

Roles de usuario

Se definen según el grupo de pertenencia del usuario y las autorizaciones que dicho grupo (o el usuario mismo) posea en el sistema

Un usuario es considerado **Submitter** sólo si tiene permiso de **ADD** en alguna colección

Workflow de revisión

Workflow configurable

Workflow de revisión

2 opciones: original y configurable

Versión por defecto: **3 pasos de revisión**

- Sólo visualización, aceptación o rechazo
- Edición de metadatos, aceptación o rechazo
- Edición de metadatos y aceptación (no se permite el rechazo)

Workflow de revisión

- Lista general de tareas pendientes
- Cualquier revisor (usuario asociado al grupo con el permiso de workflow correspondiente) puede asignarse una tarea disponible
- En cualquier momento, un revisor puede liberar una tarea asignada para que sea tomada por otro revisor

Workflow de revisión

En caso de rechazo del ítem

- Deja de estar disponible en el área de trabajo
- Pasa a figurar como una carga incompleta para el usuario que lo envió
- Se le envía un email al usuario informando la causa del rechazo (escrita por un revisor)

Workflow de revisión

El Workflow configurable permite:

- Definir una secuencia de *Pasos* de revisión (puede ser una implementación propia)
- Definir las reglas para secuenciar los pasos del workflow
- Definir workflows distintos por colección
- Definir distintos mecanismos de asignación de tareas por Paso: claimAction, autoassingAction, etc

Workflow de revisión

El Workflow configurable permite:

- Definir un conjunto de roles
- Definir un ***scope*** para los roles
  - *repository*: debe ser un grupo existente en el repositorio
  - *collection*: se asigna por colección
  - item: se asume que algún paso en el workflow asignará una persona o grupo para revisar un ítem en particular

Apache Solr

Apache Solr

- Provisto por el módulo de **Discovery**

- Aporta una mejora considerable en el tiempo de respuesta en las búsquedas
- Capacidad para personalizar la interpretación de las consultas
- Sugerencias de búsquedas
- Facilidades para *faceting* de textos, números y fechas

Apache Solr

- Importante incremento en las capacidades de búsqueda usando los filtros en la indexación
  - stemmer
  - edge n-grams
  - stop words
  - sinónimos
  - tokenizing
  - soundex

Estructura del proyecto

Estructura del proyecto

Un proyecto principal (*dspace-parent*) con múltiples subproyectos, en varios niveles:

- Un subproyecto por módulo
- Un subproyecto especial para la generación del instalador: *dspace*

  - dspace-parent
    - dspace-xmlui
      - dspace-xmlui-api
      - dspace-xmlui-webapp

Estructura del proyecto

La relación entre proyectos se mantiene con Maven

- verticalmente entre proyectos y subproyectos, configurando *módulos*
- horizontalmente entre proyectos no emparentados, mediante dependendencias

Estructura del proyecto

Tres tipos de proyectos y subproyectos:

- Agrupamiento de módulos (no genera archivos)
- Librerías (archivos jar)
- Aplicaciones web (archivos war)

Se usan en:

- Directorio de librerías (usado en el classpath)
- Overlay de aplicaciones web

Maven

Gestión de dependencias

Maven

Maven es responsable de la compilación y empaquetado (generación de jar o war)

El ***Reactor*** de Maven es un componente que lista y ordena los proyectos a procesar.

Este orden de procesamiento se determina según las dependencias entre proyectos

Maven
POM

Cada proyecto define un archivo descriptor denominado POM (pom.xml)

En el POM (Project Object Model) se especifica toda la información asociada al proyecto:

- ***Descriptiva***: grupo, id, descripción, autores, etc
- ***Dependencias***: listado de proyectos y sus respectivas versiones
- **Objetivos**: configuración específica para cada etapa de procesamiento (compilación, pre-empaquetado, empaquetado, etc)

Maven
POM

También permite definir:

- **Plugins**: componentes independientes que agregan funciones sobre los proyectos
- **Repositorios**: ubicación de respositorios para descarga de dependencias
- **Perfiles**: permite establecer parámetros de configuración a demanda (según algún criterio de activación)
- **Módulos**: definición de subproyectos
- **Licencia** del proyecto

Maven
Herencia

Los POM heredan algunos elementos de configuración de un POM

Si no se define padre, implícitamente se hereda de ***Super POM***

Esto permite centralizar información como:

- dependencias
- desarrolladores
- plugins habilitados y su configuración
- recursos (define como tratarlos)

Maven
Ejemplo con XMLUI-Webapp

Identificación y descripción del proyecto

```
<groupId>org.dspace</groupId>
<artifactId>dspace-xmlui-webapp</artifactId>
<packaging>war</packaging>
<name>DSpace XML-UI (Manakin) :: Web Application Resources</name>
<url>http://projects.dspace.org/dspace-xmlui/dspace-xmlui-webapp</url>
<description>
    DSpace/Manakin XML-UI Based Web Application using the Cocoon and
    Wing frameworks
</description>
```

Maven
Ejemplo con XMLUI-Webapp

Definición del POM padre

```
<parent>
    <groupId>org.dspace</groupId>
    <artifactId>dspace-xmlui</artifactId>
    <version>1.8.2</version>
    <relativePath>..</relativePath>
</parent>
```

Maven
Ejemplo con XMLUI-Webapp

Listado de dependencias

```
<dependencies>
    <!-- DSpace XMLUI API -->
    <dependency>
        <groupId>org.dspace</groupId>
        <artifactId>dspace-xmlui-api</artifactId>
    </dependency>
</dependencies>
```

Maven
Compilación y empaquetado

Compilar y empaquetar: *mvn package*

- Se analiza el pom.xml ubicado en el directorio de trabajo y se agrega al ***Reactor***
- Si hay módulos definidos, se incluyen en la lista de proyectos del ***Reactor***
- Luego de revisados todos los subproyectos, el ***Reactor*** establece el orden de compilación basado en la definición de las dependencias

Maven
Compilación y empaquetado

- Se ejecuta el empaquetado sobre cada uno de los proyectos del ***Reactor***, en el orden establecido.
- Para cada proyecto se descargan todas las dependencias necesarias desde algún repositorio de Maven
- Para cada proyecto se crea un directorio ***target*** que contiene los archivos compilados y el empaquetado (archivos jar o war)

Maven
Overlay de proyectos

Hay ***overlay*** cuando una aplicación web (se empaqueta como *war*) tiene dependencia de otra aplicación web.

Overlay es la *"mezcla"* de los archivos y directorios entre dos aplicaciones web, respetando un orden de prioridades (configurable)

Maven
Overlay de proyectos

MyWebApp
- index.jsp
- image.png

Depende de

BaseWebApp
- index.jsp
- other-image.png

FinalWebApp
- index.jsp
- image.png
- other-image.png

Maven
Overlay de proyectos

Permite "extender" una aplicación web, creando o redefiniendo sólo los elementos de interés

Maven permite configurar explícitamente el orden en el que se realizará el overlay y qué recursos deben considerarse en el proceso