Repositorios DSpace con múltiples contextos OAI - PMH
Universidad Nacional de La Plata

Marisa R. De Giusti, Facundo Gabriel Adorno, Ariel J. Lira

Ilustraciones en presentación realizadas por Lucas Folegotto.

Interoperabilidad

¿Cómo se define la interoperabilidad?

Capacidad de intercambio transparente de información entre dos o más sistemas.

¿Por qué interoperar?

- Aumentar la visibilidad de la información,
- Dejar accesible en un único punto de acceso información obtenida de diversos sistemas,
- Crear servicios de calidad a partir de la información intercambiada, etc.

Interoperabilidad entre repositorios digitales (protocolos)

- Intercambio de registros de metadatos (OAI-PMH),
- Depósito remoto de documentos (SWORD)
- etc...

OAI-PMH

- Protocolo de interoperabilidad para intercambio de registros de metadatos.
- Define **6 verbos** o mensajes
- Define 2 roles entre los sistemas de información:
  - **Data Provider** y **Service Provider**
- Funciona sobre **HTTP**, realizando solicitudes mediante GET o POST.
- Define mecanismos para:
  - Paginación de resultados,
  - detección de registros eliminados,
  - recuperar registros en formatos de metadatos específicos, etc.

Directrices de interoperabilidad

Debido a la vasta diversidad en definición y uso de metadatos en los distintos repositorios del mundo, surgen diversas *directrices de interoperabilidad* para normalizar la sintaxis y semántica del contenido interoperable.

Ejemplos:

- Driver Guidelines: describe la interoperabilidad en 2 capas:
  - *sintáctica* y *semántica*
- OpenAIRE Guidelines: extienden Driver; perfiladas para la compartición de resultados de investigaciones en proyectos de la Unión Europea.
- Directrices SNRD: extienden de Driver y OpenAIRE; definidas para la interoperabilidad entre repositorios en Argentina.

Directrices de interoperabilidad

Algunas similitudes y diferencias

	Driver	SNRD	OpenAIRE
Condiciones de acceso al contenido	(a) en acceso abierto (b) con embargo (no recomendable)	(a) en acceso abierto (b) con embargo	(a) en acceso abierto (b) resultantes de algún proyecto de investigación (cualquier tipo de acceso)
Uso de <dc:format>	Uso: Recomendado Esquema: MIMEtype de IANA	Uso: Obligatorio Esquema: MIMEtype de IANA	Ídem a Driver.
Vocabulario tipologías documentales	Implementa un vocabulario propio (*info: eu-repo/semantics*) y obliga su uso.	Implementa una tipología propia obligatoria. También obliga el uso de las tipologías Driver.	Ídem a Driver.

DSpace - OAI 2.0

DSpace es un software open-source para la construcción de repositorios digitales abiertos.

- Permite alojar, preservar y acceder abiertamente a todo tipo de contenido digital.
- Compuesto por un módulo que permite la **interoperabilidad**.

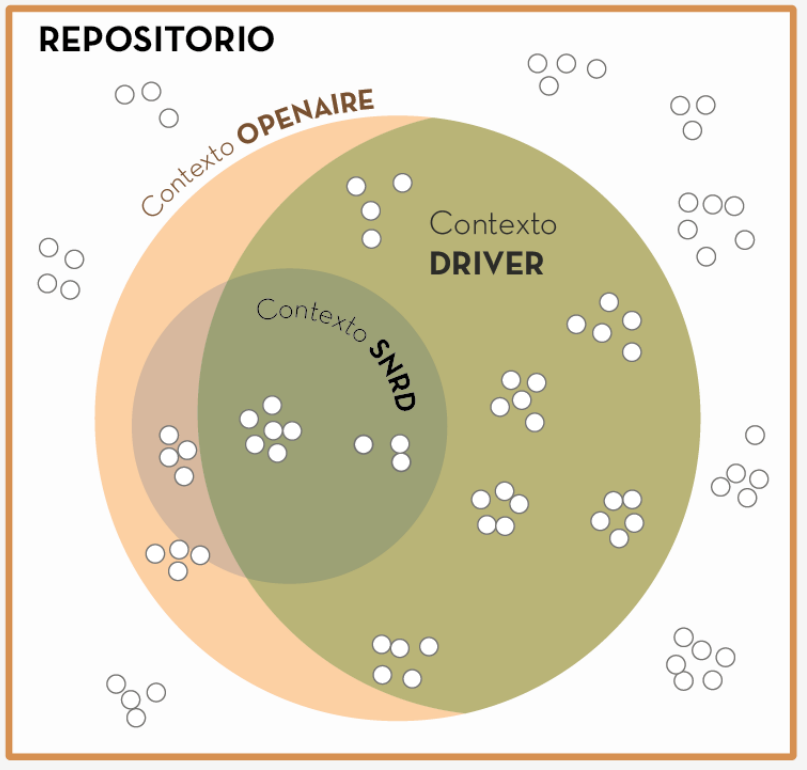
OAI 2.0 es un módulo desarrollado para su uso en DSpace para habilitar la interoperabilidad.

- Desarrollado utilizando XOAI Java Toolkit.
- Es una implementación de un Data-Provider.
- Requiere su activación luego de la instalación de DSpace.

XOAI Java Toolkit

- Librería Java que provee una serie de clases para la implementación de un Service o Data Provider, desarrollada por **Lyncode**.
- Se ajusta a las especificaciones del protocolo OAI-PMH.
- Permite que un mismo servidor OAI exponga metadatos en distintas interfaces virtuales de cosecha o **contextos**. Cada una de estos contextos simula ser un Data-Provider específico, adaptado a diversas especificaciones. Estas especificaciones pueden, p.e., extraerse de distintas directrices de interoperabilidad.

XOAI - Contextos



Ejemplo.

- 3 contextos adaptados a directrices de interoperabilidad.
- Cada contexto incluye una porción de los objetos digitales (OD) del repositorio.
- Superpuestos entre ellos.
- Algunos OD pueden no incluirse en el algún contexto.
  - ≠ Contexto *request*.

XOAI (continuación)

- Se basa en cuatro conceptos fundamentales:
  - **Contexto**
  - **Filtro**
  - **Transformador**
  - **Mapeador/Formato**
- La *combinación y agrupamiento* de distintos filtros, transformadores y mapeadores permiten determinar diversos contextos de cosecha.

XOAI (continuación)

- **Contexto**
  - Denota una selección arbitraria de objetos digitales (OD).
  - Cada OD debe cumplir ciertas características determinadas.
  - Pueden coexistir varios contextos en un único Data Provider XOAI.
  - Cada contexto es referenciado mediante una URL.
  - Puede definir sus propios filtros, transformadores y formatos.

- **Filtro**
  - Determina si un OD del repositorio debe o no incluirse en el contexto donde se define.
  - Define un criterio de selección o filtrado de ODs según ciertas características de los mismos.
  - Un filtro puede ser usado en varios contextos a la vez.

P.e. para el contexto DRIVER y SNRD:

http://sedici.unlp.edu.ar/xoai/**driver**?verb=Identify

http://sedici.unlp.edu.ar/xoai/**snrd**?verb=Identify

XOAI (continuación)

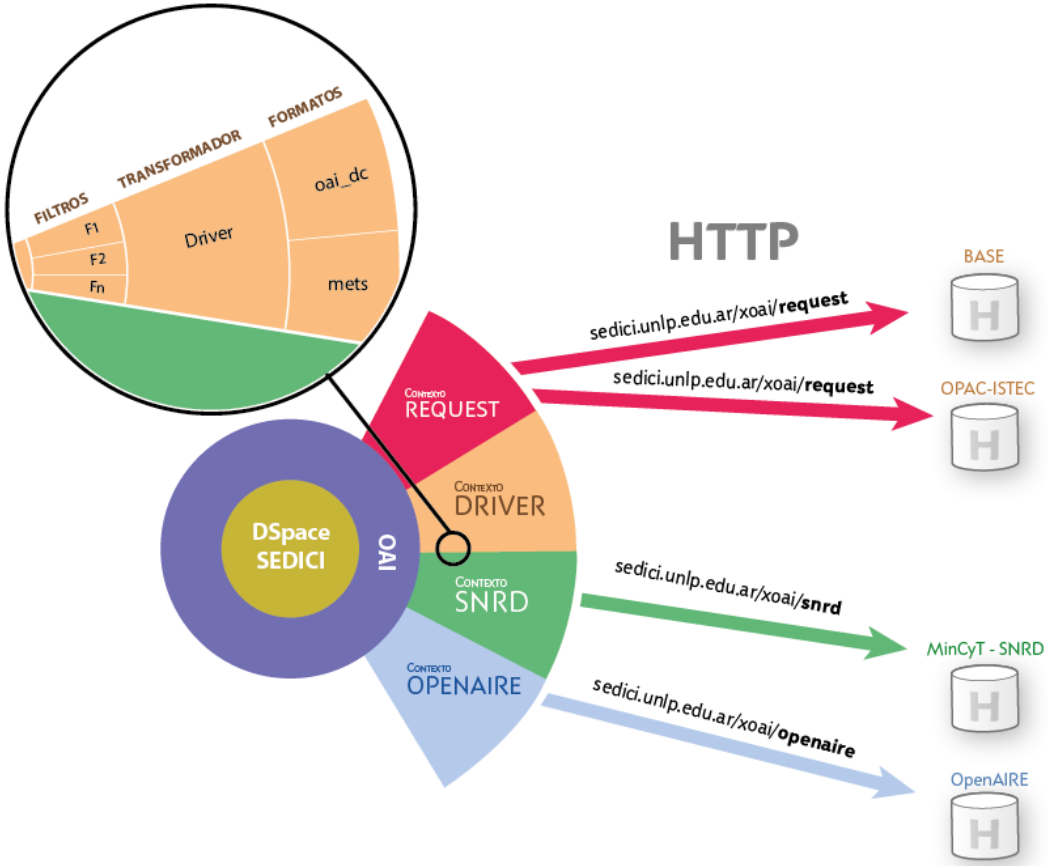
- Transformador

- Actúa sobre los metadatos de los objetos digitales (OD) filtrados (metadatos en esquema XOAI) mediante XSLT.
- Convierte un conjunto de metadatos de entrada en otro de salida.
- Cada transformación puede implicar una reducción o incremento en los metadatos de entrada, además de una alteración o normalización en los valores de dominio, p.e.:
  - códigos estándares (p.e ISO 639-3).
  - vocabularios específicos (p.e info: eu-repo/semantics)

- Mapeador/Formato

- Actúa sobre los metadatos resultantes de la aplicación de los transformadores, mediante XSLT.
- Representa un formato o esquema de metadatos (p.e. Dublin Core, METS, etc.)
- Pueden ser utilizados en varios contextos simultáneamente.
- Es el único elemento cuya existencia es *obligatoria* dentro de un contexto específico.

XOAI - Ejemplificación Data-Provider



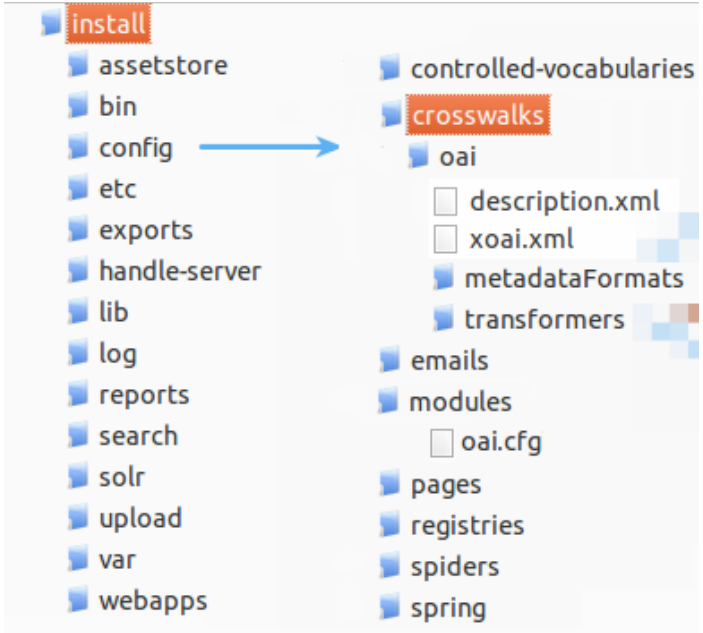
Módulo OAI 2.0

- Todos los conceptos válidos en XOAI (contextos, filtros, transformadores, formatos) son “heredados” por OAI 2.0.
- Extiende algunos puntos de XOAI para adecuarlo a DSpace. Por ejemplo:
  - Permite el uso de diversas fuentes de datos (base de datos PostgreSQL o índice SOLR) por el Data Provider.
  - Permite el uso de filtros propios para actuar sobre ambas fuentes de datos.
- Posee una caché de solicitudes *(caché request)*, de tal forma que al realizar las mismas consultas al Data-Provider, la velocidad de respuesta es mejorada.

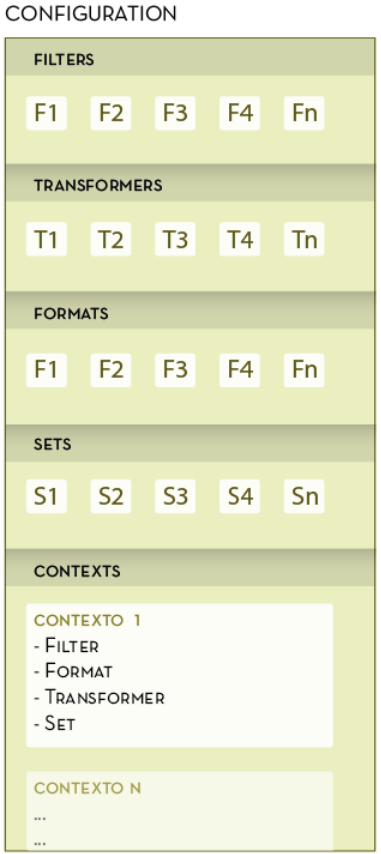
Módulo OAI 2.0 - Configuración

OAI 2.0 puede configurarse desde 2 archivos principales:

- **xoai.xml**: archivo de configuración original de XOAI. Permite configurar todos los aspectos de la aplicación:
  - Contextos, filtros, transformadores, formatos, sets.
  - Tamaño de lotes, hoja de estilos, etc...
- **oai.cfg**: archivo propio de DSpace. Permite configurar:
  - Fuente de datos (base de datos o índice solr).
  - URL de acceso al índice SOLR
  - Prefijo de los identificadores únicos OAI-PMH de registros.
  - etc...



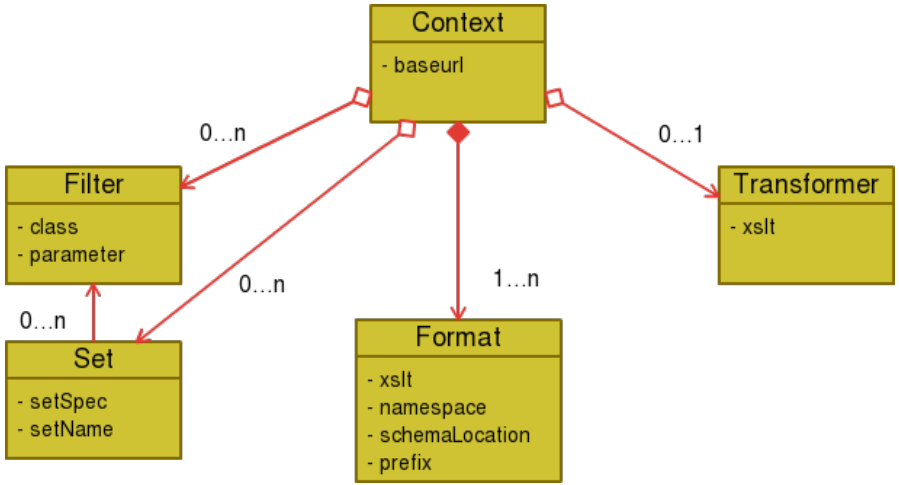
Módulo OAI 2.0 - Configuración xoai.xml



- Cada Filter indica la clase JAVA que define su comportamiento. Dentro de ésta etiqueta se puede parametrizar la clase indicando el nombre del parámetro.
- Cada Format se corresponde con un formato o esquema de metadatos. Indica qué archivo de transformación utilizará, el namespace correspondiente al formato, como se lo referenciará mediante el *metadataPrefix.*
- Cada Transformer, generalmente, se corresponde con un único contexto, e indica sólo el nombre del archivo XSL de transformación correspondiente.
- Cada Set indica la representación de un set OAI-PMH, definiendo su setSpec y setName.
- Cada Context **referencia** los filtros, formatos, sets y transformador que utilizará. Posee un atributo (*baseurl*), indica como se debe referenciar al contexto en la URL.

Módulo OAI 2.0 - **Configuración** xoai.xml

- A grandes rasgos, la configuración en contextos, filtros, transformadores y formatos, presenta un estructura similar a la que muestra el gráfico.
- Se observa claramente que la única obligatoriedad es que un *contexto* tenga al menos un *formato* para diseminar metadatos.
- También se permite la definición de sets OAI-PMH mediante la utilización de filtros.



Módulo OAI 2.0 - Funcionamiento en etapas

- Ante cada solicitud OAI-PMH que se realiza al módulo OAI 2.0, se utilizan los *componentes configurados* (filtros, formatos, etc.) para responder la misma.
- Durante la generación de la respuesta, los componentes que se utilizan varían en función del **verbo** requerido**.**

	Identify	**ListRecords**	**GetRecord**	**ListSets**	**ListMetadata Formats**	**ListIdentifiers**
Filtro	No	Sí	Sí	No	No	Sí
Transformador	No	Sí	Sí	No	No	No
Formato	No	Sí	Sí	No	No	No

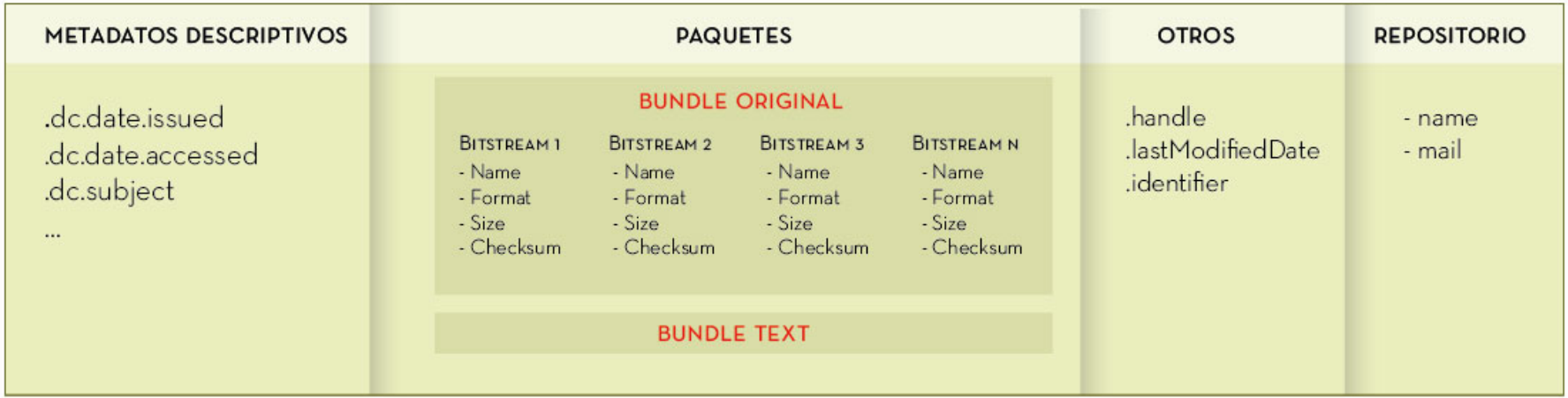
Módulo OAI 2.0 - Funcionamiento en etapas

La generación de la respuesta a la solicitud de un `GetRecord` o `ListRecords`, implica las siguientes etapas:

I. **Inicio**: Se analiza la solicitud OAI-PMH y se determina el contexto requerido, además del resto de los parámetros en la URL
II. **Selección**: Según el verbo OAI-PMH
    a. `GetRecord`. Se recupera el OD solicitado de la base de datos, luego cada filtro evalúa el OD para validarlo; si algún filtro no lo valida, entonces se aborta la solicitud.
    b. `ListRecords`. Se itera sobre cada filtro, generando una consulta a la fuente de datos (postgreSQL o Solr) para obtener todos los ítems pertenecientes al contexto solicitado.
III. **Transformación**: Se altera el contenido de los metadatos de los ítems para adecuarlo a algún vocabulario o código particular (p.e. ISO 639-3).
IV. **Mapeo**: la estructura de metadatos de cada objeto digital (OD) es transformada al formato de metadatos solicitado en la solicitud OAI-PMH (p.e. oai_dc).

Módulo OAI 2.0 - XML schema (XOAI)

- Durante la etapa de transformación como de mapeo, se realizan transformaciones XSLT para adecuar los metadatos a cada solicitud.
- Cada transformación actúa sobre un formato llamado XOAI, un esquema XML definido por el módulo de interoperabilidad. Éste formato es una **representación interna** de los objetos digitales del repositorio, como se ve en el siguiente gráfico.



Módulo OAI 2.0 - Filtros propios

Es posible que necesitemos agregar nuevos filtros al módulo cuando los que vienen por defecto no nos sirven.

Para crear un filtro nuevo, tener en cuenta:

- Crear una subclase de DSpaceFilter.
- Definir los siguientes métodos de clase:
  - **getWhere**(): Devuelve una condición SQL a utilizar en la consulta sobre la Base base de datos usada por DSpace.
  - **getQuery**(): Devuelve una condición SOLR a utilizar en la consulta sobre el motor de indexación.
  - **isShown**(): Verifica si dado un objeto digital, el mismo es apto para ser incluido en el contexto.

DSpace - próxima versión

- En la próxima versión de DSpace (5.0), posiblemente se incluyan actualizaciones sobre el módulo de interoperabilidad, potencialmente llamado OAI 2.1 y basado en la última versión estable de XOAI.

- Para seguir el desarrollo de las últimas versiones de XOAI, ir al siguiente link:

https://github.com/lyncode/xoai

http://sedici.unlp.edu.ar

¡MUCHAS GRACIAS!

Investigadores

Marisa R. De Giusti marisa.degiusti@sedici.unlp.edu.ar
Facundo G. Adorno facundo@sedici.unlp.edu.ar
Ariel j. Lira alira@sedici.unlp.edu.ar

Ilustrador

Lucas Folegotto lucas@sedici.unlp.edu.ar