Control de integridad y calidad en repositorios DSPACE

Condiciones actuales

Fuerte crecimiento de la cantidad de repositorios

Mayor cantidad de recursos para almacenar, compartir y preservar

Problemática: Control de datos

1. Con el tiempo cambian los requerimientos mínimos que debe cumplir un ítem en el repositorio
   - Surge la necesidad de verificar periódicamente estos requerimientos mínimos
2. El origen de los datos es diverso (autoarchivo, cosechas OAI-PMH, depósito vía SWORD, etc) y por tanto varían sus características
   - Es necesario validar que los ítems archivados respeten la política de contenidos.
3. A medida que aumenta el volumen de datos, cada vez es más complicado detectar anomalías
   - Se necesitan mecanismos de control/monitoreo de metadatos y archivos.

Herramientas para Control de datos en DSpace

- ChecksumChecker
  - Verifica automáticamente la integridad de cada bitstream
- MediaFilter
  - Transforma un bitstream de un item o extrae uno nuevo derivado del anterior. Ejemplo: miniaturas, fulltext, etc.
- EmbargoChecker
  - Verifica que todas las partes de un item embargado (item, bundle, bitstream) se mantengan ocultas
- *Curation Tasks* o Tareas de *Curation*
  - permite un control semi-automático de objetos del repositorio

Curation Tasks en DSpace

- Permite aplicar una “pieza de software” o simplemente *tarea* sobre uno o varios objetos del repositorio.
- Cada *tarea* se aplica sobre un objeto puntual (Item, Collection, Community) y tiene la facultad de acceder y alterar cualquier aspecto de su contexto como metadatos, bitstreams, etc.
- Provee varias tareas predefinidas. Por ejemplo:
  - Format Profiling
  - Required metadata
  - ClamScan
  - Microsoft translator
  - Link Checker

Tareas para calidad y preservación

- Se propone el uso de tareas específicas destinadas a dar soporte para la preservación a partir del monitoreo continuo de los datos del repositorio

- Objetivos
  - chequeos de calidad,
  - control de integridad y
  - extracción/generación de nuevos metadatos

B) Modificaciones al módulo de Curation DSpace

Modificación de la estrategia de ejecución de las curation task para:

- minimizar el impacto en la performance de la aplicación.
- flexibilizar los criterios de selección de los recursos a procesar.

Parte 1

(dejo esta diapo solo para separar)

Tarea 1: Chequeo de links muertos

Casi todos los ítems de repositorio tienen al menos un link en sus metadatos. Por ejemplo:

- URL a la licencia de uso
- URL de acceso al recurso (para recursos externos)
- URL al origen del registro
- URL a otras versiones o trabajos relacionados

Con el tiempo los links suelen dejar de funcionar:

- temporalmente: error interno, servidor en proceso de actualización, etc
- permanentemente: Cambio en el dominio o ruta, servicio discontinuado, etc

Tarea 1: Implementaciones DSpace

DSpace provee 2 implementaciones básicas para verificación de enlaces en los metadatos:

1. BasicLinkChecker : chequea y genera un reporte para cada metadato cuyo calificador es “uri” (i.e. dc.rights.uri).
2. MetadataValueLinkChecker: selecciona todos los metadatos de un ítem y analiza para cada uno si su contenido comienza con “http://” o “https://”. En caso afirmativo, chequea el enlace y reporta el resultado de la prueba.

Tarea 1:
Implementaciones DSpace (2)

Ambas tareas presentan los mismos inconvenientes. No permiten:

- configurar el metadato que debe validarse
  - → No queremos chequear dc.rights.uri
  - → Sí queremos chequear dc.relation.isPartOf
- Definir timeout de conexión
- Cache de respuestas
  - → El 50% de los ítems en SEDICI tiene una de las 6 licencias CC con las mismas URL. En promedio cada URL de licencia CC se chequea unas 2500 veces. Ej. http://creativecommons.org/licenses/by-nc/3.0/
- Redirecciones Web (i.e. STATUS 3xx)

Estos problemas hacen que la ejecución de la tarea sea extremadamente lenta, ineficiente y que el reporte sea inexacto, dado que pueden haber URL reportadas como fallidas que no lo estan.

Tarea 1: Solución propuesta. ConfiguraLinkChecker

Tarea de Curation que permita :

- indicar cuáles metadatos deben validarse
- manejar **redirecciones** 301 (permanentes), 302 (temporales) y 303 (*see other*, las que usa handle.net)
- permita indicar un *timeout* de conexión máximo
- mantenga un historial de URL chequeadas durante la ejecución actual (como una cache muy simple)

Tarea 2: Control de metadatos conectados con Autoridades

DSPACE no brinda soporte para gestión de autoridades típicamente vinculadas a los documentos de un repositorio. Ej: tesauros, sistemas de clasificación, autores, instituciones, etc

Se asume que los sistemas de autoridades ya existen y se gestionan externamente por lo que se permite conectar/vincular los ítems con cualquier autoridad a través de extensiones denominadas ChoiceAuthority, que recuperan datos desde servicios complementarios para luego guardarlos en un metadato.

Estos servicios pueden ser:

- internos. Ej: Términos de un vocabulario interno (XML Controlled Vocabularies)
- externos al software. Ej: Materias de un sistema de clasificación
- externos al repositorio. Ej: Autores de un sistema institucional.

Tarea 2: Control de metadatos conectados con Autoridades

En el entorno de dspace, los autoridades sólo “existen” en los metadatos que las referencian.

Cada metadato puede mantener un vínculo con una autoridad a partir del guardado de su clave y texto representativo.

Por ejemplo:

dc.contributor.author=(“156442”, “Tim Berners-Lee”)

el metadato “dc.contributor.author” referencia al autor **Tim Berners-Lee existente en** la base de Autores de la biblioteca.

Tarea 2: inconsistencias en metadatos conectados con Autoridades

El vínculo es débil ya que no cumple con los principios de integridad referencial. Si la autoridad es modificada en el sistema de gestión de autoridades externo, el vínculo no se actualiza ya que Dspace no se entera y los datos quedan:

1. descoordinados: se referencia a una autoridad que ha cambiado su nombre pero en el repositorio se tiene el nombre viejo, o aún peor,
2. con referencias colgantes: se apunta a un autoridad que no existe más

Tarea 2: Solución propuesta. AuthoritySync

Se propone la creación de una curation task que

1. se ejecute periódicamente,
2. verifique la existencia de las autoridades apuntadas desde metadatos controlados, solicitando para cada clave su contraparte a los ChoiceAuthority
   - detecta autoridades eliminadas
   - podría corregir el dato local “desconectándolo”, es decir, eliminando la clave y dejando solo el texto.
3. contraste el texto de los metadatos controlados con el valor retornado por el ChoiceAuthority
   - detecta modificaciones en los textos
   - podría corregir el dato local, copiando el nuevo
4. genere un reporte con las discrepancias acciones

3. Accesibilidad de recursos

Las políticas de contenidos de los repositorios en relación a dónde mantienen los objeto digital son variables.

- la mayoría guarda los objetos completos en el repositorio, aunque
- algunos permiten que los objetos estén alojados en sistemas externos. Algunas causas son: falta de derechos sobre la obra, incapacidad de gestionar los datos. (ej. por tamaño excesivo), etc.
    - Los recursos externos pueden referenciarse a partir de una URL o a partir de identificador persistente que permita inferir una URL. Ej: doi, handle, etc.

Hay un factor común: el repositorio siempre debe dar

3. Accesibilidad de recursos
Solución propuesta: FileChecker

Corrobora para cada ítem que al menos un objeto digital sea accesible a partir de al menos:

- uno o más bitstreams públicamente accesibles, o
- un metadato (configurable) cuyo contenido sea
  - una **URL** al archivo alojado en un servidor externo. Por ejemplo en ”dc.identifier.uri”, o
  - un **identificador persistente** (diferente al del ítem actual). Por ejemplo: a partirde hdl:1822/24377→ generamos http://hdl.handle.net/1822/24377

El resultado de la ejecución es un reporte con los ítems que no cumplen las restricciones definidas de archivo o de enlace.

4. Metadatos requeridos

La obligatoriedad de un metadato está definida en base a:

1. requisitos globales, para cualquier clase de item. Ej: fecha, título, autor
2. requisitos según la tipología documental. Ej: ISBN para libros, ISSN para revistas.
3. otros criterios institucionales como licencias de uso, fuente de financiación, reglamentación local, etc.

Tarea 4: Metadatos obligatorios

La carga de documentos en DSpace está regida por una configuración única (input-forms.xml) que permite controlar los metadatos a partir de:

- formulario de carga con campos opcionales y obligatorios según el tipo de documento (desde 3.0+)
- tarea de curation *RequiredMetadata* que revisa los documentos y genera un reporte con todos los metadatos obligatorios globales ausentes.

Tarea 4: Metadatos obligatorios

El problema de la tarea provista es que no tiene en cuenta el cambio agregado en Dspace3 sobre obligatoriedad de campos según el tipo de documentos.

Solución simple:

- Extender la curation task de Dspace a fin de considerar los nuevos criterios definidos en base al valor del metadato dc.type.

Tarea 5: Validación del dominio de metadatos. Situación

- Cada metadato posee un dominio que determina cuáles son los valores que puede tomar o al menos qué sintaxis debe respetar según el tipo de dato. Ej:
  - dc.contributor.author debe ser un texto
  - dc.date debe ser una fecha en formato ISO8601
  - dc.language debe ser un ISO 3166 (en, es, pt, etc)
- Al igual que sucedía con el chequeo de metadatos requeridos, el control de dominio debe realizarse:
  1. inicialmente en la carga
  2. periódicamente sobre los recursos existentes

Tarea 5: Validación del dominio de metadatos. Qué hace DSpace?

- No permite asociar un dominio a un metadato y por tanto no brinda un mecanismo para validar su contenido.
- Permite configurar el formulario de carga con algunos controles específicos que “ayudan” a que los datos sean correctos:
  - selector de fecha
  - caja de texto
  - selector dentro de un vocabulario controlado

Problemas:

- los controles de carga poco estrictos representan un riesgo para la calidad de los registros.
- Si existen metadatos con valores inválidos el sistema no lo reporta

Tarea 5: Validación del dominio de metadatos. Propuesta: *DomainChecker*

Se propone el desarrollo de una tarea de curation configurable que corrobore que el valor de ciertos metadatos respeta el dominio correspondiente.

Se puede definir las reglas en:

1. la tarea de curation. Es simple, pero solo sería usable desde la tarea y no desde el resto del repositorio
2. el registro de metadatos. Tiene mayor alcance, pero es complejo de implementar y podría darse que en algún caso, el dominio del metadato no deba ser el mismo para todos los tipos de documentos
3. Configuración de carga (input-forms.xml): es mas simple y permite que se use desde el resto del sistema.

Tarea 5: Validación de metadatos según su dominio. Casos a soportar

- Tipos básicos: boolean, fechas, números, textos de una línea, multilínea.
- Tipos avanzados:
  - URL
  - HTML
  - Otras: LaTeX expression, geolocation, doi, hdl, expresión regular, barcode, etc.
- Vocabularios controlados: se debe validar que el valor del metadato sea permitido según la configuración del vocabulario controlado:
  - value-pair
  - autoridades

Tarea 6: Extracción de metadatos de preservación

Para asegurar la usabilidad de los recursos digitales en el tiempo, el repositorio debe:

- garantizar que los archivos no se pierdan,
- verificar los formatos de archivos y transformarlos para evitar su obsolescencia .

El análisis de formatos de recursos requiere contar con toda la información de los recursos , los objetos digitales y con el contexto necesario para acceder a cada objeto, dato que en general está dentro de los archivos en sí.

Estos datos suelen guardarse en **metadatos de preservación**

Tarea 6: Extracción de metadatos de preservación

La información asequible a partir de los archivos dependerá en gran medida de los formatos, pero en general incluye:

- plataforma,
- datos del software con el que se generó el archivo (nombre, versiones, etc),
- fechas.
- Ejemplo: un archivo PDF mantiene al menos tamaño físico de las páginas, software, y versión con el que se creó el archivo fuente y el PDF, fecha, autor y datos de la PC usada.

Tarea 6: Extracción de metadatos de preservación. Propuesta

Desarrollo de una tarea de curation que permita automatizar la generación de metadatos de preservación.

Para cada análisis, la tarea debería

1. realizar la extracción de datos de los archivos
2. almacenar metadatos de preservación asociados a cada bitstream.

Dspace registra de cada bitstream/archivo su formato , representado a por su mimeType.

Tarea 6: Extracción de metadatos de preservación. Propuesta

- Parte simple: Extracción de datos de los archivos.
  - Hay muchas aplicaciones y librerías estables.
  - Ej: Apache Tika soporta imágenes (JPEG, PNG, etc), audio y video (MPEG, AVI, etc), MS Office (DOC, XLS, etc), y muchos más.
- Parte compleja:Guardado de metadatos de preservación
  - DSpace no permite metadatos en bitstreams.
  - Opción 1: extender DSpace para que los soporte
    - Muy complejo
    - seguramente será agregado pronto por Dspace
  - Opción 2: crear un bitstream que almacene los datos en un Bundle de preservación oculto:
    - Es menos complejo
    - No es invasivo

Tarea 7: Generación de reportes en base a expresiones.

- Los administradores precisan detectar y analizar casos complejos en los ítems y sus metadatos. Por ejemplo:
  - registros con pocos metadatos
  - registros con metadatos faltantes (no obligatorios)
  - registros con múltiples valores de un mismo metadato
- El módulo de búsqueda de Dspace sólo considera metadatos públicos, es decir, no es posible
  - buscar por metadatos administrativos u ocultos
  - acceder a información de los recursos que no está en los metadato. Ej: última fecha de modificación.
- El módulo de curation de DSpace es muy complejo de utilizar para estos casos simples, porque requiere crear

Tarea 7: Generación de reportes en base a expresiones. Propuesta

Tarea de curation para selección y reporte de objetos DSpace en base a expresiones lógicas simples.

- Las expresiones permiten analizar y comparar los datos asociados a cada objeto:
  - item → datos del ítem, bundles, bitstreams y metadatos
  - colección → datos de la colección, comunidad padre
  - comunidad → datos de la comunidad, comunidades padre e hijas, colecciones hijas.
- Procedimiento: para cada objeto analizado (un Item, una Colección o una Comunidad):
  1. se almacena el objeto en un espacio común “ValueStack”
  2. se evalúa la expresión lógica sobre el ValueStack
  3. segun el caso, se incluye el objeto en el reporte final o no.

Tarea 7: Generación de reportes en base a expresiones. Propuesta

- las expresiones usan notación puntual para acceder al ValueStack y deben evaluar a verdadero o falso
- el ValueStack brinda acceso a los parámetros de configuración y a los datos del objeto (según el tipo)
- Alternativas de implementación:
  - expresiones algebraicas tradicionales
  - Criteria

Ejemplos:

- cantidad de bitstreams en el Bundle ORIGINAL de un ítem
  - item.bundle('ORIGINAL').bitstream().count() > 0
- cantidad de metadatos dc.title de un ítem
  - item.metadata('dc.title').count() > 0
- metadatos dependientes
  - if(item.metadata('metadato1').count() > 0,

Parte 2

(dejo esta diapo solo para separar)

Mecanismo actual de las CT

- Se ejecutan de manera secuencial sobre todos los items de un repositorio, comunidad o colección. Sin interrupción.
- Generan fuerte demanda sobre el servidor durante la ejecución.
- Si hay más de una tarea, hasta que no termina la previa no procede a la siguiente.

Esto justifica la presente propuesta para la selección de recursos y para la ejecución.

Estrategia de selección de recursos

- Por expresión lógica configurable: p.e. según el valor del metadato *type.*
- Por lotes incrementales para disminuir el impacto sobre el sistema.
- Cambio en la forma de ejecución secuencial a fin de obtener un avance a nivel de recursos ejecutando en paralelo varias tareas en lugar de una tras otra.

Estrategias de selección y ejecución de tareas de curation propuestas

- Actualmente la única estrategia de selección de recursos para la ejecución de *curation tasks* que ofrece DSpace se basa en la selección de un elemento dentro de la estructura jerárquica de comunidades y colecciones.
- A veces es necesario realizar revisiones periódicas y/o cambios masivos sobre un conjunto de ítems que cumplan con unas características, ejemplo: una validación sobre los archivos asociados a los ítems de tipo “artículo” dentro de una comunidad en particular (la comunidad puede contener múltiples tipos de documento).

Estrategias de selección y ejecución de tareas de curation propuestas

- En el caso expuesto y en otros es posible que los ítems de interés se encuentren dispersos en múltiples colecciones y comunidades, las cuales además pueden contener ítems con diferentes características.
- Cada repositorio establece la estructura de comunidades y colecciones acorde a sus necesidades, así como sus propios esquemas de metadatos.
- La ejecución de las *curation tasks*, procesa todos los ítems dentro de la comunidad o colección especificada (o incluso sobre todo el repositorio), debiendo incluir las restricciones de inclusión/exclusión de ítems como parte de la lógica de procesamiento de la tarea ⇒ dificultad de reuso de la CT sobre otros conjuntos de ítems.

Estrategias de selección y ejecución de tareas de curation propuestas

- Se propone considerar una nueva estrategia para la selección de los ítems que serán procesados por la *curation task*, tomando como criterio de selección una expresión lógica que permita realizar evaluaciones simples sobre las características de un ítem.
- Se plantea el uso de una expresión como un parámetro adicional al momento de la invocación, cuya evaluación sobre cada ítem dentro del conjunto preseleccionado de ítems (en base a la estrategia de selección original) determinará si un ítem debe procesarse o no.

Ejemplo:

dspace curate -e admin@dspace.org -i all -t fileChecker -f “item.metadata(‘dc.type’) = ‘Article’”

Estrategias de selección y ejecución de tareas de curation propuestas

- DSpace solo ofrece la posibilidad de realizar ejecuciones secuenciales sobre todo el conjunto de ítems a considerar. Esto es, cuando en una invocación se indican más de una tarea a ejecutar, cada tarea comienza su trabajo sólo cuando su predecesora ha terminado el propio; y una tarea se da por terminada cuando ha completado su procesamiento sobre todos los ítems considerados (puede ser toda una comunidad, colección o el repositorio completo).
- Problema: Cuando hay controles continuos y periódicos sobre todos los ítems del repositorio, cada ejecución puede representar una carga de procesamiento y un tiempo de ejecución importantes..
  - Ejemplo: Control de enlaces: ejecutada periódicamente sobre todo el conjunto de ítems, un enlace externo puede quedar fuera de servicio en cualquier momento.

Estrategias de selección y ejecución de tareas de curation propuestas

- Tareas como la previa requieren una estrategia de ejecución distinta a la propuesta por DSpace, ya que sus tiempos de ejecución, cargas de procesamiento o acceso a base de datos, etc., suelen ser valores significativos.
- Propuesta: estrategia de ejecución de curation tasks por bloques. Esto es, dado todo el conjunto de ítems a procesar, el mismo se divide en porciones equivalentes en cantidad de ítems, cada una de las cuales será procesada de forma independiente y en invocaciones independientes.

Futuro

DynamicCTask a partir de DpsaceEL