Integración de revistas desde repositorios institucionales hacia Open Journal Systems

Gonzalo Luján Villarreal[1], Ezequiel Manzur[2], Dolores García[3], Marisa Raquel De Giusti[4]

[1] Dr. en Cs. Informáticas; Docente-Investigador UNLP; Subdirector CESGI, CIC. Correo: gonzalo@prebi.unlp.edu.ar
[2]Analista Programador Universitario; Pasante PREBI-SEDICI, UNLP. Correo: ezemanzur@sedici.unlp.edu.ar
[3]Lic. en Comunicación Social; pasante CESGI, CIC. Correo: dolores.garcia@sedici.unlp.edu.ar
[4] Dr. en Cs. Informáticas; Investigador CIC; Directora PREBI-SEDICI UNLP; Directora CESGI, CIC. Correo: marisa.degiusti@sedici.unlp.edu.ar

Resumen

Muchos repositorios institucionales alojan colecciones de revistas, que o bien nacieron en formato digital, o bien fueron digitalizadas para aprovechar la plataforma de preservación y difusión que naturalmente brindan los repositorios. Usualmente, las revistas envían su producción a los repositorios, pero existen situaciones en las que las revistas requieren recuperar desde los repositorios los artículos y sus metadatos, como ser, cuando una revista migra hacia el mundo digital y comienza a utilizar un sistema de gestión editorial. Esto sucede a menudo en la Universidad Nacional de La Plata, que cuenta con un repositorio institucional (SEDICI), basado en *DSpace*, que aloja recursos de toda la Universidad, y un portal de revistas que funciona sobre *OJS* y sirve como espacio para revistas de toda la institución. Dado que muchas revistas ya han migrado hacia dicho portal o lo están haciendo, se implementó un mecanismo para integrar los recursos del repositorio en el portal de revistas. Se desarrolló un software capaz de mapear metadatos exportados desde DSpace hacia la estructura de *OJS*, y a partir de allí generar archivos XML para importar tanto metadatos como documentos completos. Se generó una interfaz web para que usuarios no informáticos puedan generar los XML, y se estableció un flujo de trabajo con editores para organizar y agilizar la importación artículos. Esta herramienta ha sido utilizada por 6 revistas, que han servido para ajustar algoritmos, optimizar el flujo de trabajo y mejorar la interfaz para hacerla más simple y brindarle mayor robustez al proceso.

Palabras clave: OJS, DSpace, Interoperabilidad, revistas

Abstract

Many institutional repositories host journals collections, either born digital or digitalized to take advantage of the preservation and dissemination platform offered by repositories. Usually, journals send their production to repositories, but there are situations in which journals need to retrieve articles and metadata from the repositories, e.g. a journal migrating to the digital world making use of an editorial management system. Those situations are often seen at Universidad Nacional de La Plata because of its institutional repository (SEDICI), based on DSpace, which hosts resources from the entire University, and a journals portal which works on OJS and serves as a space for journals of the entire University. Since many journals have already migrated to this portal, and many others are in the process, a mechanism to integrate repository's resources into the journals portal was implemented. A software capable of mapping metadata exported from DSpace to the OJS structure was developed, which is able to generate XML data to import both metadata and complete documents. A web interface was generated so users with none to zero technical knowledge can generate XML files, and a workflow with editors to organize and expedite the importation of articles was established. This tool has been used by 6 journals, which have served to adjust algorithms, optimize the workflow, make a simpler interface and give the process greater sturdiness.

Keywords: OJS, DSpace, Interoperability, journals

Introducción

Un repositorio institucional es un espacio virtual en el que se depositan documentos digitales, y cuyo propósito es gestionar, organizar, almacenar, preservar y difundir la producción resultante de la actividades de una organización. (De Giusti, 2014, p.12). Los repositorios institucionales suelen brindar distintos mecanismos de interoperabilidad para exponer sus recursos y así permitir que puedan ser consumidos por otros sistemas: servicios de análisis de citas, sistemas de gestión de recursos de investigación (CRIS), agregadores de recursos, entre otros. En un escenario típico, los repositorios reciben objetos digitales directamente de los productores (autores de artículos, tesistas, editores, etc.) o por medio de intermediarios (desde la biblioteca, la oficina de títulos, la secretaría de investigación, etc), y sobre cada objeto recibido realizan los procesos correspondientes: control de calidad, verificación de licencias, catalogación, transformación, entre otros. Sin embargo, existen algunas situaciones en las que este escenario tradicional se invierte, y es el repositorio quien envía hacia otros sistemas los objetos digitales, para luego ser integrados en los circuitos de gestión. Un caso concreto se da con los repositorios institucionales que contienen revistas científicas, y que quieren comenzar a utilizar un sistema de gestión del ciclo editorial como *Open Journals System* (OJS).

Esta situación ha ocurrido en reiteradas ocasiones en La Universidad de La Plata, entre su repositorio institucional SEDICI (*Dspace*) y el Portal de Revistas (OJS 3). Muchas revistas que existen desde hace años (en algunos casos décadas) en el repositorio desean sumarse al Portal de Revistas, y para ello requieren algún mecanismo para importar todos los artículos y números a OJS. El software Dspace cuenta con varios mecanismos para exportar metadatos, como los archivos CSV y XML, y cuenta también con mecanismos para exponer metadatos, por ejemplo *OpenSearch* (Villarreal et al., 2017), OAI PMH o incluso campos META en los encabezados HTML. Asimismo, al momento de incorporar artículos ya publicados al sistema OJS, se puede hacer de forma manual (por ejemplo utilizando el plugin *QuickSubmit* de OJS), a través de comandos INSERT SQL o se puede utilizar el plugin de importación masiva de OJS, que utiliza el formato XML nativo para encapsular tanto los metadatos como los artículos en sus diferentes formatos (PDF, HTML, etc.).

El objetivo de este trabajo es presentar una herramienta que permite automatizar en gran parte el proceso de importación de revistas desde el repositorio hacia un portal de revistas en OJS a través del formato XML nativo, y proponer un flujo de trabajo para que los administradores de estos sistemas puedan realizar esta tarea de manera organizada y segmentada.

Materiales y metodología

Para realizar este trabajo se estudiaron los formatos mencionados para la exportación de metadatos en Dspace y los formatos de importación disponibles en OJS. Para realizar la exportación de metadatos se escogió el formato CSV por varios motivos: su simpleza a la hora de ser procesado (la mayoría de los lenguajes de programación provee soporte para interpretar archivos en formato CSV), porque ofrece la información mínima necesaria de las revistas para incorporarse a OJS, y porque permite agrupar en un único archivo todos los metadatos de todos números de una revista, lo que evita realizar múltiples exportaciones desde DSpace. Para la importación dentro de OJS, se seleccionó el formato XML nativo que utiliza el plugin de importación de artículos de OJS; si bien se evaluó la generación de consultas SQL personalizadas, este proceso puede ser muy complejo y propenso a errores (mientras que el plugin de importación funciona desde años y está en una etapa muy estable de su desarrollo). También se consideró la posibilidad de realizar la carga de artículos en OJS de forma manual, pero es evidente que este proceso requiere mucho tiempo por parte de los gestores y editores de las revistas, y es también propenso a errores originados a partir de fallas humanas. El uso de este plugin de importación es simple, el procesamiento de los mismos se realiza de manera muy eficiente, y se encuentra disponible en casi todas las instalaciones de OJS.

Una vez seleccionados los formatos de exportación e importación, se hizo necesario idear un método repetible y fácil de controlar para realizar la importación dentro de OJS. Esto sucede porque, por lo general, las importaciones implican

decenas de números y cientos de artículos, lo que requiere un trabajo metódico y cuidadoso para no pasar por alto ningún artículo, no cargar artículos repetidos y no mezclar metadatos entre artículos. En líneas generales, este flujo de trabajo consiste en una primera etapa de análisis del CSV exportado desde DSpace, segmentación en múltiples CSV de menor tamaño (uno por cada número de la revista) y generación de los documentos XML, y una segunda etapa en la cual se incorporan iterativamente los XML generados en la etapa anterior dentro de una instalación de OJS.

Procesamiento del CSV

En la etapa de procesamiento del CSV, se busca generar un mapeo al formato XML nativo capaz de ser importado en OJS. Para definir este mapeo se revisaron los metadatos de los artículos de revistas generados desde el repositorio, se relevaron los distintos esquemas disponibles en la exportación (típicamente, *Dublin Core*, aunque en ocasiones aparecen otros esquemas) y se analizaron requerimientos de normalización, mapeos y post-procesamiento de los metadatos. Además, se analizaron elementos disponibles en la exportación CSV que podrían aprovecharse para generar el XML nativo de OJS; por ejemplo, el metadato **dc.title[es]** de *Dublin Core* exportado en el CSV, que hace referencia al título en español del artículo, se presenta en el XML como el nodo title con el atributo de localización (*locale*) correspondiente. En este ejemplo, el mapeo en formato XML quedaría de la siguiente forma:

```
<title locale="es_ES"> Título en español </title>
```

De este modo, si el repositorio exporta metadatos en múltiples idiomas, al realizar el mapeo hacia XML se incorpora esta información, permitiendo así la incorporación de metadatos internacionalizados en OJS.

Al momento de realizar los mapeos, se observó que el repositorio SEDICI no exporta metadatos referidos a la sección de la revista a la que pertenece cada artículo. Sin embargo, este metadato es obligatorio en OJS: todos los artículos deben pertenecer a alguna sección en este sistema. Como solución, en todos los casos se mapean los artículos hacia una sección especial (llamada por lo general Importados o Imported), que se utiliza exclusivamente durante la etapa de migración. Esta sección es configurada desde OJS con varias limitaciones: los autores no pueden enviar artículos allí, no aparece en el índice de la revista y no se revisa por pares. De este modo, funciona como una sección interna, y su objetivo es alojar de manera temporal a los artículos recién importados, hasta tanto un editor los asigne a la sección correspondiente (reseñas de libros, artículos originales, resúmenes de tesis, etc.).

El formato XML nativo de OJS permite la incorporación de los artículos a texto completo como un elemento más del XML. Para ello, los archivos deben codificarse en base 64 (*IETF*, s/f) antes de ser adicionados al XML. De este modo,

al importar un XML a OJS, es posible importar tanto metadatos como documentos completos en formatos PDF, HTML, EPUB, etc. Sin embargo, los archivos CSV exportados por DSpace sólo incluyen metadatos, y no contienen ni los objetos digitales en sí si la URL de acceso a los mismos. Afortunadamente, dicha URL es posible de obtener a partir del documento HTML referenciado por el metadato URI de cada recurso (en el caso de SEDICI, dicho metadato será **sedici.identifier.uri**). Entonces, la aplicación aquí presentada, además de realizar el mapeo de metadatos, utiliza el metadato URI para recuperar la página web (documento HTML) de cada recurso en el repositorio, luego analiza dicho HTML con el objetivo de recuperar el enlace al documento PDF, alojado en el nodo META **citation_pdf_url** (ver Imagen 1). Una vez obtenido dicho enlace, esta aplicación descarga desde el repositorio el documento PDF a través de una conexión HTTP y, una vez finalizada la descarga, codifica el documento PDF obtenido y lo incorpora dentro del XML que está siendo generado.

Imagen 1: Representación del documento HTML de un artículo.

Importación en OJS

Los documentos XML obtenidos luego del mapeo deben ser importados uno a uno dentro de OJS. Este trabajo lo debe realizar un usuario con rol de editor o gestor, y luego de importar cada XML debe realizar una revisión manual de todos los artículos importados para:

- corregir posibles errores de mapeo, o completar datos faltantes (datos que el repositorio no poseía)
- ubicar los artçiulos en las secciones correspondientes
- publicar los números una vez revisados todos los artículos.

Cabe destacar que los documentos XML generados ya poseen la información sobre los números dentro de OJS. Esto evita al usuario la tarea de crear manualmente cada número, y de asignar los artículos importados a los

números correspondientes.

El desarrollo del software que da soporte a todo este proceso se realizó en dos etapas. En la primera, se creó una aplicación que funcionaba desde la línea de comandos, y cuyo funcionamiento era muy sencillo: dado un archivo CSV con los metadatos de la revista que se ingresaba como parámetro, sumado a algunos parámetros adicionales necesarios para procesar dicho archivo (por ejemplo la cantidad de registros a procesar o la sección de destino de los artículos), se procesaba y dividía el archivo CSV en varios archivos CSV de menor tamaño según el criterio ya explicado, y luego se procesaba a cada CSV con el objetivo de generar los archivos en formato XML correspondientes a cada número de la revista.

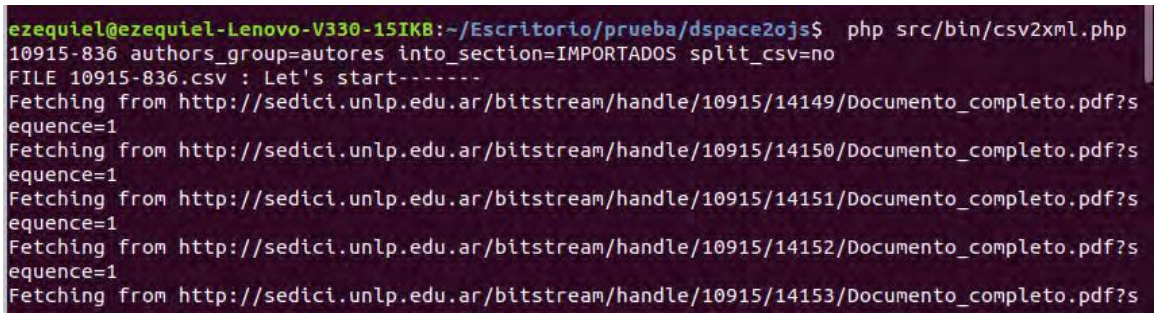


Imagen 2: Generación de archivos XML a partir de un único archivo CSV (10915-836.csv), utilizando la aplicación desde la línea de comandos.

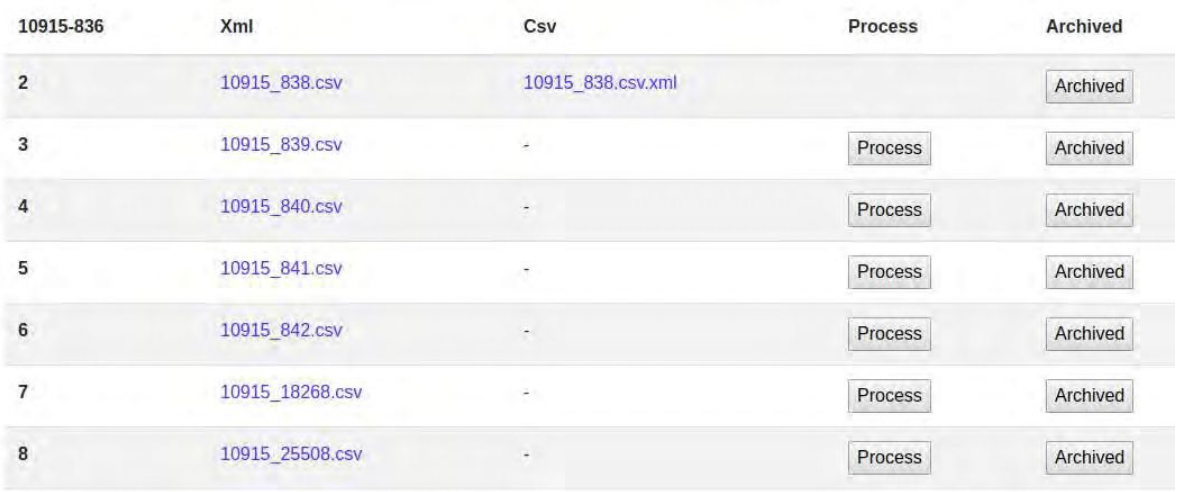
Como se mencionó previamente, la importación de una revista completa puede implicar cientos de artículos agrupados en decenas de números y volúmenes. Para organizar toda esta información, se agruparon los artículos en documentos XML por número. Esto fue posible gracias a la organización de recursos dentro del repositorio SEDICI, en el que cada revista posee una comunidad (Imagen 3), y cada número de las revistas es una subcomunidad dentro de la comunidad de la revista. La estructura de colecciones y comunidades de este repositorio es mucho más amplia, e incluye a las Unidades Académicas y colecciones especiales (*SEDICI*, s/f), pero esta estructura se gestiona de manera independiente a la jerarquía de comunidades de revistas, lo que permite asegurar que la información obtenida a partir de la jerarquía de revistas y números sólo contendrá elementos (artículos) de revistas.

Imagen 3: Subcomunidades de revistas en el repositorio SEDICI.

La información de las comunidades y subcomunidades se aprovechó también para generar los nombres de los archivos CSV: el CSV exportado por DSpace tiene por nombre el *handle* de la comunidad de la revista dentro del repositorio, y los archivos CSV y XML que representan un número específico toman por nombre el *handle* de la colección a la que pertenece cada uno, del CSV importado. Por ejemplo, a partir del *handle* de la revista *TE & ET* (hdl.handle.net/10915/836) el nombre del archivo exportado desde el repositorio resulta **10915-836.csv**, siendo la primera serie de números el prefijo del *handle* de SEDICI y la segunda, el identificador de la comunidad. Al momento de procesar este archivo, se obtiene el *handle* de cada número de la revista para nombrar a cada uno de los CSV derivados, manteniendo la estructura de prefijo de SEDICI y el identificador de colección que corresponde a cada número. El XML que se genera para cada número conserva el mismo nombre del CSV a partir del cual es generado, pero con la extensión xml. Entonces, a partir del ejemplo anterior, algunos de los CSV que representan a los números de la revista se pueden

nombrar como **10915_837.csv**, **10915_838.csv**, **10915_839.csv**, etc. Y el nombre de cada XML **10915_837.csv.xml**, **10915_838.csv.xml**, **10915_839.csv.xml**.

En la segunda etapa del desarrollo de esta aplicación se incorporó una interfaz de usuario web, que facilita el uso de la herramienta a personas sin conocimientos informáticos. Esta interfaz permite cargar el archivo CSV generado desde el repositorio y, una vez procesado en múltiples archivos CSV, permite generar uno a uno los XML correspondientes a cada CSV. Durante esta segunda etapa del desarrollo se agregaron también algunas funcionalidades nuevas, como la posibilidad de crear secciones de la revista para importar artículos, o generar automáticamente los números en OJS a partir de los metadatos del XML. Además, esta herramienta permite al usuario mantener un control sobre el proceso de migración, no solo mediante la generación a demanda de los documentos XML, sino también marcando (archivando) los archivos XML ya importados, o incluso brindando la posibilidad de procesar todos los archivos CSV juntos (muy útil en el caso de revistas de menor tamaño).



10915-836	Xml	Csv	Process	Archived
2	10915_838.csv	10915_838.csv.xml		Archived
3	10915_839.csv	-	Process	Archived
4	10915_840.csv	-	Process	Archived
5	10915_841.csv	-	Process	Archived
6	10915_842.csv	-	Process	Archived
7	10915_18268.csv	-	Process	Archived
8	10915_25508.csv	-	Process	Archived

Imagen 4: Interfaz de usuario del sistema.

Decisiones particulares sobre los mapeos de metadatos

Los dos sistemas con los que se trabaja aquí, OJS y DSpace, utilizan estructuras de metadatos muy diferentes; sumado a esto, las herramientas de importación y exportación utilizadas realizan sus propias adaptaciones sobre dichos metadatos: algunos metadatos se combinan en uno solo al ser exportados, otros metadatos no existen o no son exportados, etc. Asimismo, la estructura jerárquica del formato XML nativo de OJS, si bien es muy sencilla (son relativamente pocos elementos), posee algunas restricciones en cuanto a los valores posibles y en cuanto a la obligatoriedad de muchos de sus componentes. A continuación se presenta una vista simplificada del formato XML nativo de OJS, donde puede apreciarse la estructura jerárquica del mismo y los elementos que

considera:

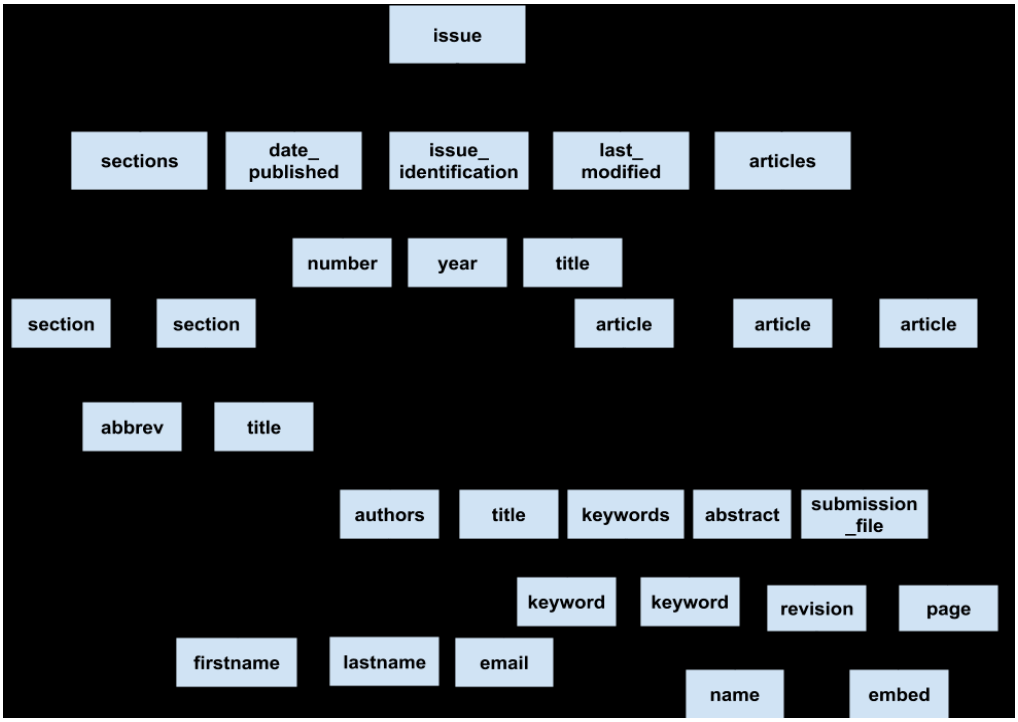


Imagen 5: Estructura jerárquica del XML nativo.

A diferencia del formato XML nativo, las exportaciones en CSV generadas por DSpace se componen de decenas de columnas con metadatos, que pueden variar según cada tipo de documento (artículo, tesis, libro, etc.) y que pueden pertenecer a más de un esquema de metadatos, como ser Dublin Core, Mods, elementos propios de DSpace o esquema de metadatos propio del repositorio. La Tabla 1 expone los mapeos realizados entre metadatos de ambos formatos.

Nodo XML		Metadato en el CSV	Descripción
issue_identification	number	-	El software le asigna un orden creciente a los números; pero puede no coincidir con el real. Este metadato debe revisarse y modificarse, de ser necesario.
	year	dc.date.issued	
	title	-	Se asigna un título relacionado con el número. Este metadato se debe corregir o eliminar.

		date_published	dc.date.issued	Este metadato lo exige el XML nativo, por eso se incorpora el mismo dato *year*
		last_modified		
section		abbrev	-	Este metadato lo exige el XML nativo, por eso se le asigna el mismo valor que el título.
		title	-	Nombre de la sección donde serán importados los artículos. Se ingresa desde el software.
article		title	dc.title	Metadato que se puede discriminar por idioma.
		abstract	dc.description.abstract	Metadato que se puede discriminar por idioma.
		keyword	sedici.subject.keyword	Las palabras clave se importan separadas por ";" y el software se ocupa de generar un nodo para cada una. Este metadato que se puede discriminar por idioma.
	author	firstname	sedici.creator.person	Los autores vienen separados por "||", el software lo procesa y obtiene nombre y apellido para cada autor
		lastname		
		email	-	Este metadato no es exportado desde SEDICI y es un campo obligatorio en el XML nativo, por lo que se crea un mail falso, y se corrige en OJS.
	revision	name	-	Nombre del formato del archivo (PDF)
		embed	sedici.identifier.uri	De este metadato se obtiene la URI al documento. El software descarga el documento, lo codifica en base 64 y lo incorpora en el XML.
		pages		

Tabla 1: Mapeos de metadatos entre el esquema XML nativo de OJS y los metadatos exportados desde DSpace.

Resultados y trabajos futuros

Este desarrollo fue utilizado con éxito en la migración de 6 revistas desde el repositorio SEDICI hacia Portal de Revistas de la Universidad Nacional de La Plata:

- Journal of Computer Science and Technology (*JCST)*
- *Tecnología en Educación y Educación en Tecnología (TE&ET)*
- *Económica*
- *Analecta Veterinaria*
- *Biología Acuática*
- *Anales JURSOC*

Es importante destacar que algunas de estas revistas (JCST y TE&ET) funcionan sobre un OJS externo, gestionado por la Facultad de Informática de la UNLP. El desarrollo aquí propuesto no está atado al OJS del Portal de Revistas de la UNLP, sino que genera documentos XML compatibles con cualquier sistema OJS (versión 3.0 o superior).

En líneas generales, se estima que en aproximadamente 10 o 15 minutos un editor puede importar, revisar y publicar un número completo de entre 10 y 12 artículos. Los tiempos para la generación del XML a importar son más variables, ya que entra en juego la velocidad de conexión con el repositorio y el tamaño de los archivos que se descargan. De todos modos, en las migraciones realizadas hasta ahora, los tiempos de generación de los XML oscilaron entre unos pocos segundos hasta unos 4 minutos para revistas cuyos números poseen muchos artículos y archivos PDF de gran tamaño. Por lo tanto, si se suman los tiempos de generación del XML, descarga desde la herramienta, carga y procesamiento en OJS, revisión y corrección de metadatos, y finalmente publicación del número, el tiempo total no supera los 20 minutos por número. Es evidente que la alternativa manual a este proceso, en la que el usuario debe descargar a mano cada PDF desde el repositorio y además copiar uno a uno los metadatos de cada artículo desde el repositorio hacia OJS, requerirá mucho más tiempo y será más propenso a errores y falla. Si bien el proceso para migrar una revista no está automatizado por completo (se debe importar los XML de a uno, revisar y completar los metadatos de los artículos y luego publicarlos), tanto el uso de la herramienta, como el flujo de trabajo propuesto reducen, de forma considerable, el tiempo para migrar revistas hacia el portal.

Otra ventaja de esta herramienta es el soporte que brinda a la hora de organizar el flujo de trabajo, en particular la segunda etapa en la que la misma aplicación genera una sección especial donde se importan los artículos -para evitar confusiones con los envíos en proceso editorial-, crea los números y permite gestionar el progreso de importación a través de la interfaz de usuario.

Más allá de las ventajas del uso de esta herramienta, a medida que más usuarios la utilizaron comenzaron a surgir algunas situaciones en las que se planteaba la necesidad de incorporar o de mejorar funcionalidades. En primera

medida, se observó la necesidad de considerar otros esquemas de metadatos más allá de Dublin Core; cabe recordar que, como se observa en la tabla 1, esta aplicación utiliza de momento sólo metadatos *Dublin Core* y del esquema propio de SEDICI. Si bien el diseño de la aplicación considera los mapeos entre metadatos de manera especial, ya que considera una sección interna de configuración de mapeos y transformaciones de metadatos, esto no ha sido puesto a prueba con otros repositorios y tampoco es evidente para los usuarios finales. El problema puntual se manifiesta cuando se requiere su uso desde otro repositorio, con sus propios metadatos, o cuando se deben incorporar metadatos que no existen, como puede ser el caso del correo electrónico del autor. Una posible solución para este último caso podría ser la incorporación de los metadatos de forma dinámica y configurable desde la aplicación, de modo que cada usuario pueda elegir con qué esquemas representar a cada uno del XML nativo.

En segunda medida, como se mencionó previamente, este desarrollo hace uso de la estructura jerárquica del repositorio SEDICI, valiéndose de las colecciones y comunidades que representan a las revistas y a sus números. Si bien esto resultó una gran ventaja en el contexto de la UNLP, esto no es posible de aplicar en repositorios que se encuentran estructurados de manera diferente. Por ejemplo, el repositorio CIC-Digital[1] utiliza las comunidades para agrupar recursos de los distintos centros de investigación, y carece de una comunidad o colección dedicada para las revistas. De todos modos, dado que los metadatos exportados por DSpace incluyen información sobre el número y el año de cada artículo, con algunos cambios en el análisis del CSV inicial podría realizarse la separación en múltiples CSV y a partir de allí generar los XML. Cabe destacar que esto requerirá también una etapa de filtrado, ya que en principio el CSV exportado podría incluir recursos de todo tipo, y no solo artículos de revistas propias.

Otra carencia de este desarrollo es la incorporación de artículos en más de un formato. Esto se debe al uso del elemento META **citation_pdf_url**, que es único por cada artículo, incluso si el artículo posee varias versiones (EPUB, HTML, XML JATS, etc.). Para resolver este problema, la herramienta podría obtener los enlaces a los documentos a partir del cuerpo del HTML del repositorio, en vez de utilizar el elemento META del encabezado.

Finalmente, sería deseable permitir al usuario cargar distintas fuentes de datos en vez de un archivo CSV exportado desde el repositorio. Resulta particularmente interesante la posibilidad de brindar soporte al formato XML utilizado por el protocolo OAI PHM. De este modo, el usuario podría especificar una interfaz de acceso OAI a la colección o comunidad dentro de un repositorio, que incluso podría seguir directrices de interoperabilidad estandarizadas (ej. *Driver* u *OpenAire*), cosechar desde allí los metadatos en formato XML, y tomar ese XML como insumo de entrada para generar el o los XML en formato nativo de OJS.

1 Consultar en https://digital.cic.gba.gob.ar/.

Referencia

1. Batch Metadata Editing, DuraSpace Wiki (en línea). https://wiki.duraspace.org/display/DSDOC5x/Batch+Metadata+Editing.
2. *IETF* (s/f). The Base16, Base32, and Base64 Data Encodings (en línea). https://tools.ietf.org/html/rfc4648.
3. De Giusti, M. R. (2014). Una metodología de evaluación de repositorios digitales para asegurar la preservación en el tiempo y el acceso a los contenidos (Doctoral dissertation, Facultad de Informática).
4. De Giusti, M., García, D., Manzur, E., Villarreal, G. L., & Folegotto, L. E. (2018). Importación de artículos en OJS desde Dspace. Recuperado de http://repositorio.pucp.edu.pe/index/handle/123456789/133195.
5. DSpace importing and exporting items via Simple Archive Format (en línea). https://wiki.duraspace.org/display/DSDOC5x/Importing+and+Exporting+Items+via+Simple+Archive+Format#ImportingandExportingItemsviaSimpleArchiveFormat-ExportingItems.
6. DSpace2OJS, espacio Github SEDICI UNLP (en línea). https://github.com/sedici/dspace2ojs.
7. OJS Data Import and Export (en línea). https://docs.pkp.sfu.ca/admin-guide/en/data-import-and-export.
8. Open Journal Systems and Dataverse Integration – Helping Journals to Upgrade Data Publication for Reusable Research (en línea). https://openscholar.mit.edu/sites/default/files/dept/files/the_code4lib_journal_-_open_journal_sys.data_publication_for_reusable_research.pdf.
9. *SEDICI* (s/f). Preguntas frecuentes (en línea). http://sedici.unlp.edu.ar/pages/FAQ#organizacion.
10. Villarreal, G. L., Manzur, E., Vila, M. M., De Giusti, M. R. (2017). Interoperabilidad con repositorios digitales: uso de OpenSearch en sitios web institucionales (en línea). En *Conferencia Internacional sobre Bibliotecas y Repositorios Digitales de América Latina (BIREDIAL-ISTEC'17) y Simposio Internacional de Biblioteca Digitales (SIBD'17) (Vol. 7).* Recuperado de http://sedici.unlp.edu.ar/handle/10915/63566.