UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

**TFG del Grado en Ingeniería Informática**

# Integración de los datos del CENIEH en *ARIADNEplus*

Presentado por Gonzalo Cuesta Marín
en Universidad de Burgos — 19 de julio de 2020
Tutores: Dr. Carlos López Nozal
y D. Mario Juez Gil

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

D. Carlos López Nozal y D. Mario Juez Gil, profesores del Departamento de Ingeniería Informática, área de Lenguajes y Sistemas Informáticos.

Exponen:

Que el alumno D. Gonzalo Cuesta Marín, con DNI 71310247N, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado título de TFG.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 19 de julio de 2020

V$^{o}$. B$^{o}$. del Tutor:

D. Carlos López Nozal

V$^{o}$. B$^{o}$. del tutor:

D. Mario Juez Gil

## Resumen

El Centro Nacional de Investigación sobre la Evolución Humana (CENIEH), se ha incorporado recientemente al proyecto europeo *ARIANDEplus*. Este tiene como objetivo estimular la investigación en áreas relacionadas con la arqueología mediante la integración de diversas infraestructuras de datos arqueológicas situadas en Europa.

Son muchas las técnicas, herramientas y metodologías involucradas en el proceso de integración al que se ven sometidos los datos propuestos por cada uno de los participantes del proyecto. Entre ellas, podemos destacar la estandarización, mapeo y enriquecimiento de metadatos.

En este trabajo se expone el proceso que se ha llevado a cabo para la integración de los conjuntos de datos del CENIEH en *ARIANDEplus*.

Asímismo, se propone una infraestructura *software* que permite gestionar los conjuntos de datos involucrados en el proceso y provee herramientas que facilitan alguna de las tareas implicadas en el mismo.



## Descriptores

Integración de datos, estandarización e interoperabilidad de datos, mapeo de datos, enriquecimiento de datos, metadatos, sistema de gestión de contenidos.

## Abstract

The National Research Center for Human Evolution (CENIEH) has recently joined the European project *ARIADNEplus*. This project aims to promote archaeological research by integrating different archaeological databases set in Europe.

There are many methods (and techniques) involved in the integration of data originating from each participating center. Among them, we could highlight Data Standardization, Data Mapping and Data Enrichment.

This work tries to elaborate on the process followed to integrate the CENIEH data into *ARIADNEplus*.

We also propose a software-defined infrastructure that allows for the correct management of the included data and provides tools for facilitating some of the tasks involved in the process.



## Keywords

Data integration, data interoperability and standarization, data mapping, data enrichment, metadata, content management system.

# Índice general

# Índice de figuras

# Índice de tablas

# Introducción

El Centro Nacional de Investigación sobre la Evolución Humana, también conocido como CENIEH, se ha incorporado recientemente al proyecto europeo denominado *ARIADNEplus*.

*ARIADNEplus* proporciona una infraestructura de investigación orientada a la arqueología cuyo principal objetivo es apoyar la investigación, el aprendizaje y la enseñanza al permitir el libre acceso a recursos y servicios digitales. Lo consigue manteniendo un catálogo de conjuntos de datos digitales, promoviendo buenas prácticas en la gestión y uso de datos digitales arqueológicos, y apoyando el desarrollo de nuevos servicios innovadores para la arqueología.


Figura 1.1: Vista del catálogo oficial de *ARIADNEplus*.

El producto estrella de *ARIADNEplus* es su catálogo (*Figura* 1.1), lugar donde se exponen todos los conjuntos de datos ofrecidos por cada uno de los participantes del proyecto. Estos datos son meramente descriptivos, es decir,

no contienen el dato al que hacen referencia. Es por tanto competencia de los socios discernir quién tiene acceso a los datos reales y quién no. Podría decirse que el catálogo solo actúa como un simple escaparate que permite mostrar a los investigadores de todo el mundo información sobre el qué, cuándo y dónde de los datos propietarios de cada socio.

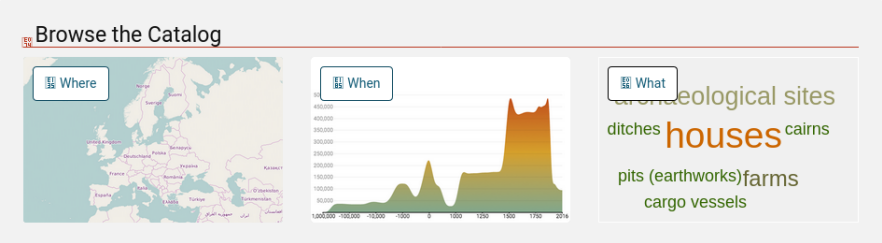


Figura 1.2: Filtros de búsqueda principales del catálogo.

Son estos tres conceptos, el qué, cuándo y dónde, los pilares de información sobre los que está construido el catálogo. Hacen referencia al tipo de dato (e.g. *fieldwork*), al espacio temporal en el que se enmarca (e.g. *Neolithic*) , y al lugar geográfico donde se ubica (e.g. *Sierra de Atapuerca*).

Con la realización de este proyecto se pretende llevar a cabo el proceso de integración de los conjuntos de datos del CENIEH en *ARIADNEplus*, de forma que estos sean visibles desde su catálogo oficial.

Además, para aplicar los conocimientos desarrollados durante toda la carrera, se propone una infraestructura *software* que permita gestionar estos conjuntos de datos y sirva como guía a los investigadores del CENIEH durante el mencionado proceso de integración.

# Objetivos del proyecto

En este apartado se indican, en primer lugar, los objetivos generales fijados durante el comienzo del proyecto. Seguido de estos, se describen los objetivos específicos, que se corresponden con los pasos previos que se han tomado para alcanzar las metas previamente fijadas.

## 2.1. Objetivos generales

- Integrar los conjuntos de datos propuestos por el CENIEH en *ARIADNEplus*.
- Proporcionar al CENIEH una infraestructura *software* que permita:
    - ◇ Gestionar sus metadatos en la integración con *ARIADNEplus*.
    - ◇ Transformar sus esquemas de metadatos a un esquema estandarizado compatible con *ARIADNEplus*.
    - ◇ Compartir los datos de forma que estos sean accesibles desde el exterior.
    - ◇ Facilitar la integración de los metadatos en *ARIADNEplus*.

## 2.2. Objetivos específicos

- Estudiar el proyecto *ARIADNEplus*, poniendo especial hincapié en el proceso de integración de los datos.
- Estudiar todos los conjuntos de datos involucrados en el proyecto.
- Diseñar e implementar un esquema de metadatos que satisfaga las necesidades de ambas partes, es decir, pueda ser transformado al modelo

objetivo (*AO-CAT*) y, además, tenga la capacidad de representar fehacientemente los conjuntos de datos propuestos por el CENIEH.

- Encontrar una aplicación *software* que cumpla con un mínimo de requisitos:
    - ◇ Sea *software* libre.
    - ◇ Tolere un esquema de metadatos compatible con *CIDOC-CRM* o alguna de sus variantes utilizadas por *ARIADNE-plus* como *ACDM* o *AO-CAT*.
    - ◇ Cuente con un sistema de importación y exportación de metadatos.
- Adaptar la aplicación seleccionada a las necesidades del proyecto a través del desarrollo de complementos (*plugins*).
- Estudio y uso de los componentes que aporta *Zend Framework* para el desarrollo de complementos (*plugins*).
- Aplicar la arquitectura MVC (*Model-View-Controller*) en el desarrollo de *plugins*.
- Estudio y uso de lenguajes empleados para el desarrollo web como *PHP*, *HTML*, *JavaScript*, *jQuery* y *CSS*.
- Utilizar bases de datos relacionales *MySQL* (*MariaDB*).
- Crear un entorno de desarrollo en *Google Cloud* haciendo uso de *Google Kubernetes Engine*.
- Trabajar con *Docker* para facilitar el despliegue de la infraestructura sobre los entornos de trabajo.
- Aplicar técnicas de integración continua a través de herramientas como *GitHub Actions* o *Codacy*.
- Aprender a utilizar el conjunto de herramientas alojadas en los entornos de investigación virtuales emphVRES de la infraestructura *D4Science*, en concreto, *Vocabulary Matching Tool* y *X3ML Mapping Tool*.
- Utilizar como sistema de documentación continua *Read the Docs*.

# Conceptos teóricos

A lo largo de este apartado se van a exponer los conceptos teóricos relacionados con las dos primeras fases en las que se divide el proyecto, que son investigación y desarrollo.

## 3.1. Conceptos teóricos relativos a la investigación

En esta sección se definen todos aquellos conceptos relacionados con la investigación previa al desarrollo e implementación de la infraestructura *software* propuesta.

### *ARIADNEplus*

*ARIADNEplus* [3] es la continuación del proyecto *ARIADNE* [2], el cual fue fundado por la Comisión Europea en febrero de 2013. Nació con el propósito de estimular la investigación en áreas relacionadas con la arqueología mediante la integración de diversas infraestructuras de datos arqueológicas situadas en Europa. Fruto de este proyecto surgió un catálogo *on-line* de metadatos referentes a conjuntos de datos que incluían informes no publicados, imágenes, mapas, bases de datos, y otros tipos de información arqueológica.

Este segundo proyecto forma parte del programa *H2020*, fundado también por la Comisión Europea. El proyecto se encuentra en desarrollo desde enero de 2019 y tiene previsto una duración total de 48 meses. A través de *ARIADNEplus*, se actualizarán y extenderán los datos del catálogo *on-line* anterior añadiendo a los mismos dimensión geográfica y temporal.

Además, se van a incorporar más organizaciones arqueológicas Europeas (entre ellas el CENIEH). También proveerá nuevos servicios en la nube para procesar y re-utilizar los datos incluidos en su portal.

### *CENIEH*

El Centro Nacional de Investigación sobre la Evolución Humana [5], tambien conocido como CENIEH, es una Infraestructura Científica y Técnica Singular (ICTS) abierta al uso de la comunidad científica y tecnológica, en la que se desarrollan investigaciones en el ámbito de la evolución humana durante el Neógeno superior y Cuaternario, promoviendo la sensibilización y transferencia de conocimientos a la sociedad e impulsando y apoyando la realización y colaboración en excavaciones de yacimientos de estos periodos, tanto españoles como de otros países.

Además, el CENIEH es responsable de la conservación, restauración, gestión y registro de las colecciones paleontológicas y arqueológicas procedentes de las excavaciones de Atapuerca y otros yacimientos tanto nacionales como internacionales de similares características.

### *CIR*

El CIR [7] (CENIEH Institutional Repository) es el repositorio bibliográfico institucional del CENIEH. Alberga toda la información fruto de la actividad investigadora desarrollada en el CENIEH como, por ejemplo, publicaciones científicas.

Toda la información que recoge está organizada en ítems que pertenecen a una colección, que a su vez forman parte de una comunidad. Cada ítem tiene asignado un conjunto de metadatos que describen al objeto digital que contiene. El esquema de metadatos utilizado por la plataforma se le conoce como *Dublin Core*.

### Metadatos

Los metadatos proporcionan la información mínima necesaria para identificar un recurso, pudiendo incluir información descriptiva sobre el contexto, calidad y condición o característica del dato [30]. Puede resultar algo complejo de entender ya que podemos reducir su definición a "son datos que describen otros datos".

Para aportar algo de claridad a esta definición se aplicará el concepto de "metadato" tomando como ejemplo una biblioteca. En este contexto, el conjunto de datos estaría formado por los libros y el conjunto de metadatos se correspondería con las fichas asociadas a cada libro. Este ejemplo de metadato está algo anticuado ya que se presenta de una forma física, no digital.

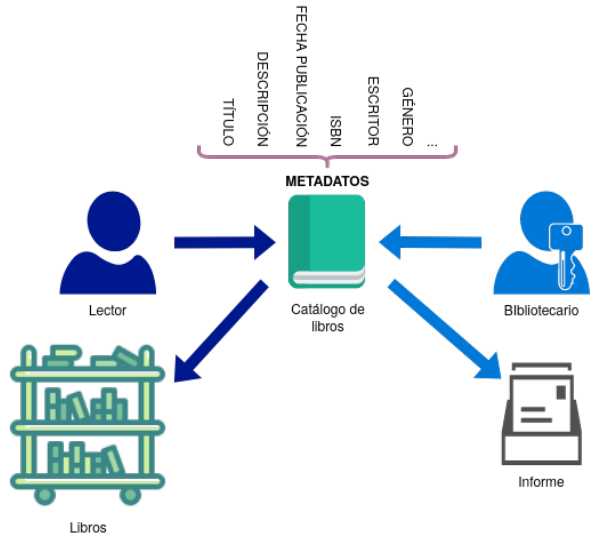


Figura 3.3: Ejemplo de metadatos.

En la actualidad, estas "fichas" se encuentran en formato digital a través de lenguajes de marcado como *XML* o *RDF*.

## Esquema de metadatos

Antes de introducir metadatos en cualquier catálogo, es necesario indicar como van a estar organizados. Para llevar a cabo esta tarea hay que definir antes un esquema de metadatos, también llamado modelo o estándar.

Cada esquema está formado por un conjunto de campos de diferentes tipos, los cuales siguen una estructura jerárquica en forma de árbol.

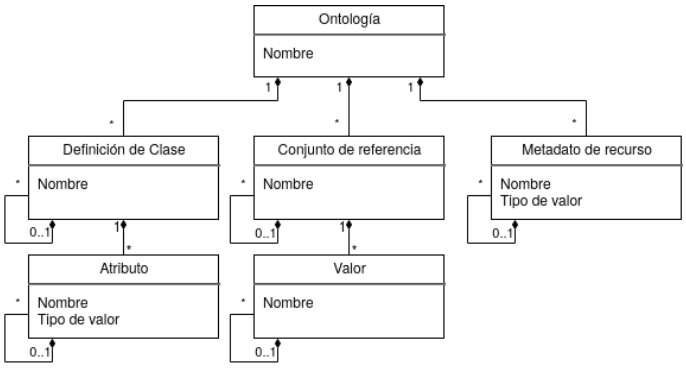


Figura 3.4: Estructura básica de un esquema de metadatos.

En la Figura 3.4, se muestra la **estructura básica** de cualquier esquema:

- **Ontología**: es la raíz del esquema. Su función es agrupar los demás campos en una única unidad temática. Puede tener tres tipos de descendientes: Clase, Referencia o Metadato.
- **Definición de Clase**: define una clase o subclase dentro de una ontología determinada, creando así una jerarquía de clases.
    - ◇ **Atributo**: define un atributo para una determinada clase existente en la ontología.
- **Conjunto de Referencia**: define un conjunto de valores que pueden ser instanciados en el Atributo de una Clase o en el Metadato de un recurso.
    - ◇ **Valor**: define el contenido de cada valor existente en un conjunto de referencia.
- **Metadato de Recurso**: define el metadato de un recurso determinado. Además, puede ser descendiente de otro metadato a modo de especificación.

Cuando se define un atributo o un metadato, se debe indicar, además, el tipo de contenido que va a adquirir, es decir, señalar qué se va a introducir. Algunos pueden ser texto plano, otros coordenadas, fechas, enlaces, etc.

### *CIDOC-CRM*

***CIDOC** **C**onceptual **R**eference **M**odel* [6] (CRM) es una ontología que ofrece definiciones y una estructura formal para describir conceptos implícitos y explícitos, así como las relaciones utilizadas en documentación sobre patrimonio cultural.

### *ACDM*

El ***A**RIADNE **C**atalogue **D**ata **M**odel* es el modelo de datos utilizado por el catálogo antiguo de *ARIADNE*. Sirve para describir los recursos arqueológicos publicados por los participantes del proyecto. El uso de *ACDM* posibilita el descubrimiento, acceso e integración de los citados recursos. Para formalizar este modelo, se ha utilizado como base la ontología *CIDOC CRM*, la cual se adapta correctamente al dominio arqueológico.

### *PEM*

*PEM* [18](***P**ARTHENOS **E**ntities **M**odel*) es el esquema de metadatos desarrollado en el proyecto *PARTHENOS* [28] que extiende al modelo *CIDOC-*

*CRM*. Está diseñado para ser lo suficientemente flexible como para mapear los diferentes tipos de esquemas de metadatos utilizados en todas las disciplinas académicas de manera uniforme.

### AO-Cat

La ontología *AO-Cat* [17] (***A**RIADNE **O**ntology - **Cat**alog*) deriva del modelo *ACDM*, empleado por el proyecto antiguo (*ARIADNE*) para modelar recursos arqueológicos, y del modelo *PEM*, utilizado para modelar cualquier recurso gestionado por una infraestructura de investigación.

Se podría decir que *AO-Cat* es una contracción del modelo *ACDM* impulsada por la conceptualización subyacente al *PEM*. Además, *AO-Cat* hereda del modelo *PEM* su estrecha relación con el modelo *CIDOC-CRM*, el cual sirve para representar cualquier aspecto relacionado con recursos arqueológicos.

*AO-Cat* es el **modelo utilizado por el catálogo actual de *ARIADNEplus*** y, por tanto, los metadatos de todos los socios del proyecto se tienen que transformar a este modelo.

## Mapeo de datos (*Data Mapping*)

El término "mapeo" puede utilizarse en múltiples contextos como, por ejemplo, en la cartografía, matemáticas, neurociencia, etc. En esta ocasión, se describirá el concepto relacionado con la informática, más específicamente con la gestión de datos.

El mapeo de datos consiste en crear asignaciones entre dos elementos que pertenecen a esquemas de datos distintos. En procesos como la integración o migración de datos es fundamental llevar a cabo este tipo de proceso debido a que, generalmente, el sistema al que se trasladan los datos no utiliza la misma estructura que el sistema de partida.

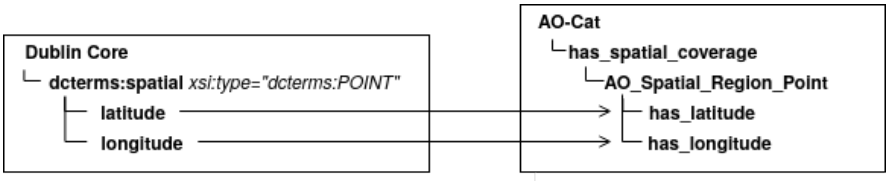


Figura 3.5: Ejemplo de definición de mapeo entre el esquema "Dublin Core" y el modelo "AO-Cat".

### Enriquecimiento de datos (*Data Enrichment*)

El enriquecimiento de datos es el proceso mediante el cual es posible mejorar la calidad de los datos sin necesidad de procesarlos. Durante este proceso, se fusionan los datos originales con datos de terceros provenientes de una fuente autorizada externa.

Para determinar la relación entre los datos originales y los externos se suele hacer uso de herramientas auxiliares que permiten establecer dichas relaciones.

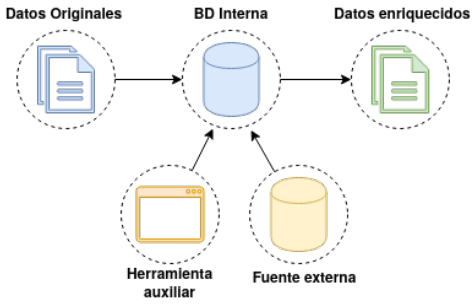


Figura 3.6: Proceso de enriquecimiento de datos.

### *D4Science* – Entornos de investigación virtuales

*D4Science* [11] es una organización que ofrece una infraestructura de datos basada en entornos de investigación virtuales (*VREs* [4]). En este tipo de entornos el usuario cuenta con un espacio de trabajo virtual que le da la posibilidad de acceder a datos y compartir los suyos propios. Además, también cuenta con herramientas y capacidad de cómputo para hacer uso de los datos en su proceso de investigación.

#### *ARIADNEplus Gateway*

*ARIADNEplus* cuenta con un portal en la plataforma *D4Science* denominado *ARIADNEplus Gateway* [10]. En él tiene implementados varios entornos virtuales de investigación. Cada uno de ellos ofrece una serie de servicios que facilitan el proceso de integración a los miembros del proyecto. Actualmente, cuenta con tres entornos virtuales, cada uno de los cuales tiene un fin específico:


Figura 3.7: Entornos virtuales de investigación en D4Science.

- *ARIADNEplus Aggregation Management*: entorno virtual donde los líderes del proyecto gestionan las importaciones de metadatos al catálogo. El acceso está restringido a los coordinadores del proyecto.
- *ARIADNEplus Mappings*: entorno virtual que da soporte a la conversión de metadatos (*mapping*) para su integración en *ARIADNEplus*.
- *ARIADNEplus Project*: entorno virtual que permite la colaboración y cooperación entre los beneficiarios del proyecto *ARIADNEplus*.

### *Workspace*

Otro de los servicios que ofrece *D4Science* es el *Workspace* [12]. La idea principal de esta herramienta es que los miembros de un determinado portal intercambien recursos digitales como, por ejemplo, documentos, imágenes, vídeos, etc.

En este espacio de trabajo los miembros de *ARIADNEplus* organizan y comparten recursos relacionados con el proyecto como, por ejemplos, guías, tutoriales, presentaciones, etc.

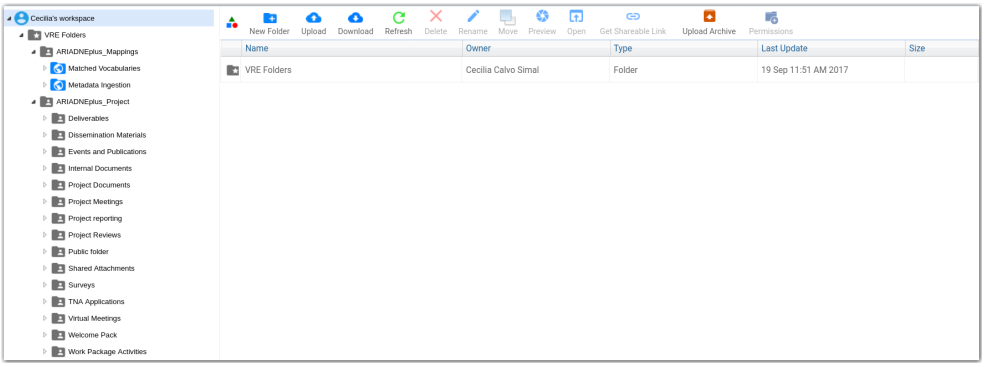


Figura 3.8: Espacio de trabajo (*Workspace*) del proyecto *ARIADNEPlus*.

Además, este mismo espacio se puede utilizar como medio de importación para el catálogo de *ARIADNEPlus*. Para tal fin, como podemos ver en el

imagen, existen dos carpetas públicas, *Matched Vocabularies* y *Metadata Ingestion*, en cuyo interior se aloja una carpeta para cada miembro del proyecto. La misión de cada carpeta es almacenar los ficheros de definición de mapeo de vocabulario (*.json*) y los ficheros con los metadatos (*.xml*). De esta manera, el coordinador puede acceder a los datos necesarios para llevar a cabo la importación sin necesidad de usar medios externos.

### *Getty AAT*

*Getty AAT* [19] es un vocabulario controlado y estructurado que se emplea para describir elementos de arte, arquitectura y material cultural. Está compuesto por términos generales como, por ejemplo, "Acueducto", pero no contiene nombres propios como "Acueducto de Segovia". Actualmente cuenta con alrededor de 55.000 conceptos registrados, incluyendo 131.000 términos, descripciones, citaciones bibliográficas, y otra información relacionada con las áreas previamente mencionadas.

Además cuenta con una interfaz *SPARQL* [20] que permite realizar consultas sobre los datos (*RDF*) almacenados en su base de datos mediante el lenguaje *SPARQL*.

### *PeriodO*

*PeriodO* [19] es un diccionario digital público donde se almacenan definiciones académicas de periodos históricos, histórico-artísticos y arqueológicos. Este proyecto es liderado por Adam Rabinowitz (Universidad de Texas, Austin) y Ryan Shaw (Universidad de Carolina del norte, Chapel Hill).

### Tecnología *GraphDB*

*ARIADNEplus* almacena todos los metadatos en un almacén de *RDF* (*triplestore*) basado en la tecnología *GraphDB* [27]. Este tipo de tecnología utiliza **bases de datos orientadas a grafos**. Estas se basan en un conjunto de objetos (vértices y aristas) que permiten representar datos interconectados junto a las relaciones existentes entre sí.

Cada grafo está compuesto por nodos o vértices, que se corresponden con los datos (objetos), y aristas o arcos, que serían las relaciones entre los datos.

La estructura de este tipo de bases de datos puede adoptar dos formas: *Labeled-Property Graph* (grafo de propiedades etiquetadas) o *Resource Description Framework* (marco de descripción de recursos, *RDF*).

*GraphDB* adopta la segunda estructura, que consiste en estructurar los grafos mediante *triples* y *quads*: los *triples* están compuestos por nodo-arco-nodo y los *quads* complementan a estos con información de contexto adicional, lo que facilita la división de los datos en grupos. Esta estrutucta es la ideal para almacenar ontologías como *AO-CAT*, de ahí que *ARIADNEplus* haya escogido esta tecnología.

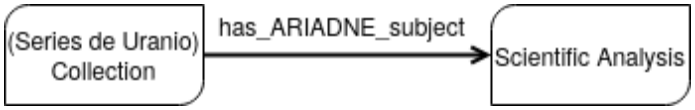


Figura 3.9: *GraphDB – Triple*.

En la Figura 3.9 se ha representado un *triple* que se correspondería con una parte del grafo asociado a la colección *CIR* almacenada en este tipo de base de datos. Vemos como se compone de dos nodos, uno para el sujeto (i.e. *CIR*) y otro para el objeto (i.e. *Scientific Analysis*), unidos por un arco, que sería el predicado (i.e. *has_ARIADNE_ subject*).

## 3.2. Conceptos teóricos relativos al desarrollo de la infraestructura

A continuación se definen aquellos conceptos relacionados con el desarrollo de la infraestructura.

### Sistema de gestión de contenidos (*CMS*)

Un sistema de gestión de contenidos o *CMS* [35] (***C**ontent **M**anagement **S**ystem*) es una aplicación *software*, generalmente de tipo *web*, que permite crear un entorno de trabajo para la creación y gestión de contenidos.

Este tipo de sistemas interactúan con una o varias bases de datos que almacenan el contenido sobre el que se realizan las operaciones de gestión. Además, suelen contar con sistemas que permiten adaptar la aplicación, tanto en diseño como en funcionalidad, de una forma sencilla.

La aplicación escogida para este proyecto (*Omeka Classic* [25]) se puede catalogar como *CMS*.

### LAMP

Las siglas *LAMP* [33] son utilizadas para describir infraestructuras *software* que hacen uso de cuatro herramientas específicas:

- ***L**inux* como sistema operativo.
- ***A**pache* como servidor web.
- ***M**ysql o **M**ariaDB* como gestor de base de datos.
- ***P**HP* como lenguaje de programación.

La aplicación *software* escogida requiere dicha infraestructura.

### Complementos (*Plugins*)

Los complementos, más conocidos como *plugins*, son aplicaciones que permiten ampliar la funcionalidad básica de un determinado producto software. Normalmente este tipo de aplicaciones son ejecutadas a través del *software* principal, interactuando con este a través de una determinada interfaz.

Mediante este tipo de aplicaciones se han conseguido añadir las funcionalidades requeridas por el proyecto en la aplicación escogida.

### *Hooking*

El término *hooking* [32] es empleado para referirse a todas aquellas técnicas utilizadas para modificar el comportamiento de un sistema operativo, aplicación u otro componente *software* interceptando llamadas de función, mensajes o eventos pasados entre componentes *software*. El código que maneja estos acontecimientos se le denomina *hook*.

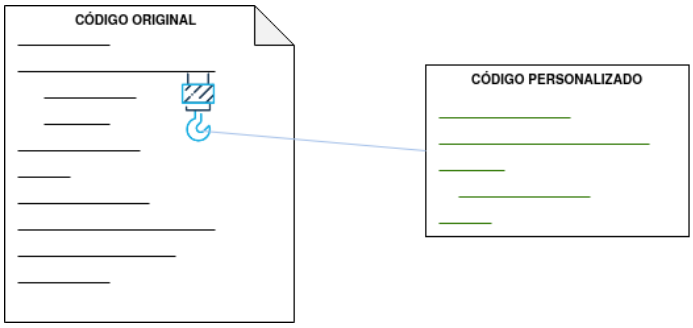


Figura 3.10: Ejemplo de *hook*.

Haciendo uso de estos *hooks* se ha conseguido modificar el comportamiento de la aplicación escogida.

## Prácticas ágiles

Durante la fase de desarrollo, se han adoptado una serie de prácticas ágiles que han contribuído favorablemente al desarrollo del *software*. A continuación, se explica en qué consiste cada una de ellas.

### Desarrollo iterativo e incremental

En un desarrollo iterativo e incremental el proyecto se va planificando en intervalos de tiempo constantes, cada uno de los cuales recibe el nombre de iteración. En todas las iteraciones se sigue un mismo procedimiento (de ahí el nombre de iterativo) para conseguir una funcionalidad determinada del producto que se pretende desarrollar.

En cada iteración, se van completando partes del producto final que son aptas para ser entregadas al cliente. Este goteo constante de entregas es el responsable de que a este procedimiento se le denomine incremental. Para que esto sea posible, se definen unos objetivos/requisitos al inicio de cada iteración que marcarán la evolución del proyecto. También se pueden plantear mejoras para requisitos que se entregaron en iteraciones anteriores.

### Pruebas unitarias

Las pruebas unitarias permiten comprobar el correcto funcionamiento de unidades de código fuente. Con el uso de este tipo de pruebas se pretende asegurar que cada unidad se comporta adecuadamente frente a distintas situaciones.

Resulta complicado determinar a qué nos referimos cuando decimos "unidad de código" ya que, por definición, se puede asociar este concepto tanto a una clase como a un método.

Habitualmente se desarrolla más de una prueba unitaria por unidad de código. El motivo radica en que una prueba unitaria sólo es capaz de comprobar el comportamiento de la unidad ante una única entrada. Lo ideal es comprobar su comportamiento ante todas aquellas entradas que tengan una probabilidad razonable de hacer que falle. El conjunto de pruebas que recoge todas estas entradas se le denomina *test suite*.

### Integración y Despliegue continuo (CI/CD)

La integración continua (*CI*) es una práctica utilizada en el desarrollo de *software* mediante la cual es posible automatizar operaciones tales como la

compilación o ejecución de pruebas. Aplicando esta metodologíam se consigue detectar fallos con mayor rapidez, mejorar la calidad del código y reducir el tiempo empleado en validar y publicar nuevas actualizaciones *software*.

El despliegue continuo (CD) se puede considerar como el siguiente paso a la integración continua, es decir, una vez automatizados los procesos de compilación y ejecución de pruebas, se procede a automatizar el despliegue del producto *software* que estemos desarrollando.

## 3.3. Otros conceptos

En este apartado se recogen todos aquellos conceptos que tienen cierta relevancia en el proyecto y no han sido expuestos en secciones anteriores.

### *Dublin Core*

*Dublin Core* es un esquema de metadatos elaborado por la *DCMI* [14], organización cuya misión principal es facilitar la compartición de recursos *on-line* por medio del desarrollo de un modelo de metadatos "base", capaz de proporcionar información descriptiva básica sobre cualquier recurso, sin importar el formato de origen, área de especialización u origen cultural.

Dispone de 15 elementos descriptivos, los cuales pueden ser repetidos, aparecer en cualquier orden y estar o no presentes (son opcionales).

### *Dublin Core Extended*

Dado que el modelo *Dublin Core* puede resultar algo escueto, se presenta como solución el esquema *Dublin Core Extended*, el cual cuenta con los elementos descriptivos del modelo original y, además, incluye una serie de elementos adicionales/complementarios [15] que satisfacen las necesidades que el modelo original no cubre.

Este modelo ha sido el propuesto para transformar todos los conjuntos de datos del *CENIEH* a un único modelo estándar.

### Interoperabilidad

La interoperabilidad es la capacidad que tiene un sistema o producto de compartir datos y posibilitar el intercambio de información y conocimiento entre ellos [21]. En lo que respecta a repositorios, se puede conseguir dicha capacidad haciendo uso de estándares como, por ejemplo, el protocolo *OAI-PMH*.

### Protocolo *OAI-PMH*

El protocolo *Open Archive Initiative-Protocol for Metadata Harvesting* (*OAI-PMH*) tiene como objetivo desarrollar y promover estándares de interoperabilidad que faciliten la difusión eficiente de contenidos en Internet. Permite transmitir metadatos entre diferentes tipos de infraestructuras *software* (repositorios, gestores, etc.) siempre y cuando éstos se codifiquen en *Dublin Core*.

Gracias a que la aplicación escogida ofrece este servicio, haciendo uso del mismo se han podido recolectar todos los metadatos existentes en el *CIR*. Además, *ARIADNEplus* permite importar metadatos en su catálogo haciendo uso de este protocolo, por lo que su implantación también abre otro posible camino de importación.

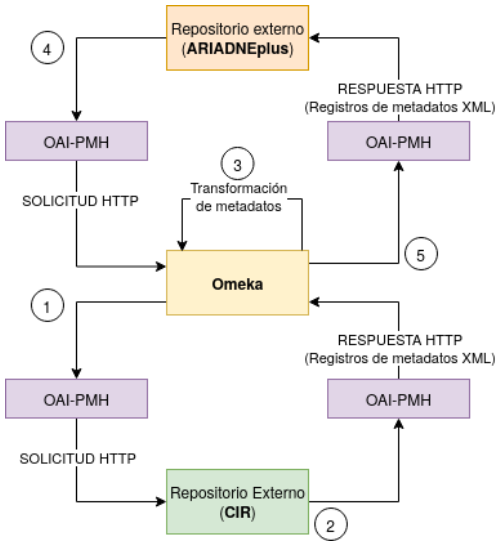


Figura 3.11: Ejemplo básico del protocolo *OAI-PMH*.

### Geolocalización

La geolocalización es la capacidad para obtener la ubicación geográfica real de un objeto [31]. Uno de los requisitos fundamentales del catálodo de *ARIADNEplus* es que todos los metadatos importados han de estar geolocalizados, es decir, tienen que tener, al menos, un elemento descriptivo que indique la ubicación actual del objeto. Nuestra plataforma cuenta con el elemento *Spatial Coverage* del modelo *Dublin Core Extended* para cubrir este requisito.

### WSG84

El **W**orld **G**eodetic **S**ystem **84** es un sistema de coordenadas geográficas usado mundialmente para localizar cualquier punto de la Tierra [37]. Uno de los requisitos del catálogo de *ARIADNEplus* es que todas aquellas localizaciones señaladas a través de coordenadas geográficas deben utilizar este sistema.

# Técnicas y herramientas

## 4.1. Metodologías

### Scrum

Scrum es un marco de trabajo donde se ejecutan procesos ágiles que contribuyen al desarrollo y mantenimiento de productos *software*. Por ello, está catalogado como una metodología ágil, la cual se caracteriza por trabajar con un ciclo de vida iterativo e incremental, donde se va liberando el producto *software* de forma periódica a través de sprints (iteraciones) [29].

## 4.2. Cliente de control de versiones

- Herramientas consideradas: Gitg, Smartgit, Gitkraken y GitHub Desktop.
- Herramienta elegida: Gitkraken.

*GitKraken* es un cliente para el control de versiones *Git* que nos permite realizar todas y cada una de las tareas propias de *Git* a través de una intuitiva y elegante interfaz gráfica. Además, incorpora funciones adicionales como *GitFlow*, que nos permite gestionar las diferentes ramificaciones del proyecto. De entre todas las opciones posibles es, en mi opinión, la más competente tanto en diseño como en funcionalidad.

## 4.3. *Hosting* del repositorio

- Herramientas consideradas: GitHub y GitLab.

- Herramienta elegida: GitHub.

*GitHub* es el servicio de *hosting* de *Git* más utilizado para albergar repositorios de código en la nube. El hecho de que cuente con una enorme comunidad de usuarios y que además ofrezca servicios exclusivos y gratuitos a estudiantes, lo convierte en la mejor opción posible. Alguno de estos servicios son: repositorios privados, cantidad ilimitada de colaboradores por repositorio y código propietario entre otros.

## 4.4. Gestor de contenidos

- Gestores de contenidos considerados: DSpace, Archimede, MyCoRe, Omeka Classic, Geonetwork y DCCD.
- Gestor elegido: Omeka Classic.

*Omeka Classic* es una plataforma de gestión de contenido libre, flexible y de código abierto. Su misión principal es la publicación de colecciones digitales provenientes de bibliotecas, museos o cualquier tipo de institución que pretenda difundir su patrimonio cultural, como es el caso del *CENIEH*. Los motivos principales por los que se ha decidido escoger esta aplicación son:

1. Se distribuye bajo una Licencia Pública General (*GNU*), con lo cual su distribución, uso y modificación es libre.
2. Utiliza un entorno *PHP-MySQL* (Fácil despliegue sobre el servidor)
3. Basado en estándares internacionalmente aceptados como *Dublin Core* o *W3C*.
4. Flexible, escalable y extensible.
    - *Zend framework* como arquitectura.
    - APIs documentadas.
    - Le respalda una gran comunidad de usuarios y desarrolladores.
5. Asistencia técnica gratuita gracias a la existencia de foros donde desarrolladores del proyecto oficial aportan soluciones.
6. Pensado para ser utilizado por usuarios no necesariamente expertos en el manejo de las TIC.

## 4.5. Servidor HTTP Apache

El servidor HTTP Apache es un servidor web HTTP gratuito, de código abierto y multi-plataforma (Unix, Linux, Windows y otras) que implementa el

protocolo HTTP/1.1 y HTTP2. Alguna de sus características más importantes son: su configuración e instalación es bastante sencilla, puede ser extendido o adaptado a través de módulos, incorpora funciones para la autentificación y validación de usuarios, y da soporte a varios lenguajes de programación como *PHP* y *Python*.

## 4.6. *MySQL*

MySQL es uno de los sistemas de gestión de base de datos más utilizados en la actualidad. Gestiona bases de datos relacionales, es decir, aquellas que siguen el modelo relacional. Es gratuito, multi-plataforma (disponible en Linux, Windows y Unix) y es utilizado con numerosos lenguajes de programación, siendo el más utilizado con diferencia PHP.

## 4.7. Librerías

### *Zend Framework*

Zend Framework (ZF) es un marco de trabajo de *PHP* desarrollado por la compañia *Zend Technologies*, principal responsable del mantenimiento del lenguaje de programación *PHP*. A través de *ZF* se pueden desarrollar aplicaciones web de una forma mucho más rápida y sencilla que utilizando PHP puro. Alguna de sus características son: aporta componentes que se pueden reutilizar en el código base de la aplicación, emplea el patrón de diseño *MVP* (con las ventajas que eso conlleva), y permite escalar la aplicación web con el concepto de módulos (*plugins*).

### *PHPUnit*

PHPUnit es un *framework* utilizado para desarrollar pruebas unitarias sobre aplicaciones basadas en *PHP*. Es un proyecto *open-source* creado por Sebastian Bergmann cuyo repositorio oficial se encuentra en GitHub.

### *ZipStream*

ZipStream es una librería de PHP que permite transmitir archivos zip de forma dinámica, sin necesidad de ocupar espacio en el servidor.

### *Leaflet*

Leaflet es una librería *open-source* de *JavaScript* utilizada para crear mapas interactivos sobre aplicaciones web. Sorprende lo ligera que es (39kB) para la cantidad de servicios que ofrece. Además de las funciones presentes en su versión original, se pueden añadir nuevas funciones gracias a los *plugins* desarrollados por la comunidad.

### *Leaflet Draw*

Leaflet draw es una extensión de la librería *Leaflet* que provee herramientas para dibujar y editar figuras sobre los mapas creados por esta librería. Permite delimitar regiones con múltiples formas (rectangular, circular, etc.), bastante útil cuando un simple marcador no es suficiente para señalar una determinada ubicación.

### *jQuery*

jQuery es una de las librerías más conocidas de *JavaScript*. Aporta multitud de funciones que facilitan el desarrollo de aplicaciones web enriquecidas del lado del cliente. Estas permiten llevar a cabo servicios tales como la manipulación de documentos HTML, el manejo de eventos, animaciones, ajax, etc.

### *Notify*

Notify es una extensión de la librería *jQuery* que proporciona notificaciones simples altamente personalizables. Es un buen medio para mantener informado al usuario ante determinadas acciones o eventos.

### *Sweet alert 2*

Sweet alert 2 es una extensión de la librería *jQuery* que permite crear *popups* (ventanas emergentes) de alerta personalizados. Es una buena alternativa a los *popups* de alerta que *JavaScript* ofrece por defecto ya que se pueden añadir nuevas funciones que mejoran la calidad de información mostrada al cliente.

## 4.8. Patrón de diseño

### MVP (Modelo-Vista-Controlador)

MVP es un patrón de diseño muy utilizado en el desarrollo de aplicaciones web. Permite organizar el código en tres secciones diferentes: Modelo, Vista y Controlador. La primera se encarga del manejo de los datos y la lógica de negocio, la segunda está relacionada con el diseño y la presentación, y la tercera interactúa con las dos primeras.

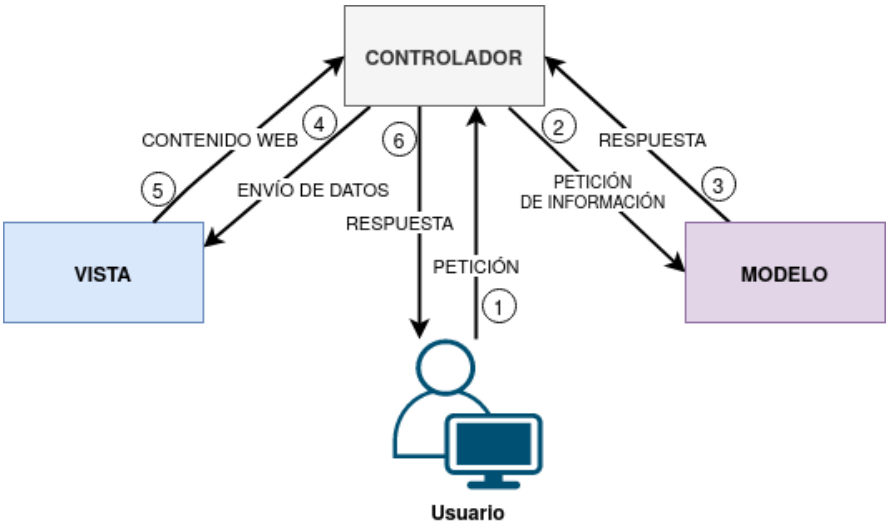


Figura 4.12: Diagrama que muestra la relación entre Modelo, Vista y Controlador.

- **Modelo**: modifica, gestiona y actualiza los datos de la aplicación. En el caso de contar con una única base de datos, sería la capa donde se encuentra el código relacionado con las consultas, búsquedas, filtros y actualizaciones.
- **Vista**: muestra al usuario final la interfaz gráfica de la aplicación, es decir, las páginas, ventanas, formularios, etc. En términos de programación se correspondería con el *frontend*.
- **Controlador**: gestiona, atiende y procesa las peticiones realizadas por parte de los usuarios. A través de esta capa se comunican el modelo y la vista. Como vemos en la `mvc`, el controlador solicita los datos necesarios al modelo, se manipulan acorde a la petición del usuario y se entregan a la vista de forma que el usuario pueda visualizar los resultados esperados.

## 4.9. Entorno de desarrollo integrado (IDE)

***PHP*** | ***CSS*** | ***JavaScript*** | ***XML***

- Herramientas consideradas: NetBeans, Atom, Eclipse, Zend Studio y Komodo.
- Herramienta elegida: NetBeans.

*NetBeans* es un entorno de desarrollo muy completo escrito en Java. Contiene una gran cantidad de funcionalidades y da soporte a todos y cada uno de los lenguajes de programación utilizados en el desarrollo de la infraestructura *software*. Además, se pueden instalar complementos que permiten extender su compatibilidad con otros marcos de trabajo como *Zend Framework*.

**LaTeX**

- Herramientas consideradas: TeXstudio y Texmaker.
- Herramienta elegida: Texmaker.

*Texmaker* es un editor libre y gratuito para LaTeX distribuido bajo la licencia GPL. Incluye múltiples herramientas necesarias para elaborar documentos tanto con LaTeX como BibText o Metapost. También incorpora funciones adicionales como la corrección ortográfica, el auto-completado y plegado de código o un visor de pdf compatible con SyncTeX y con modo de visualización continua. Además, es multi-plataforma, disponible tanto en UNIX como en MacOS y Windows.

## 4.10. Generador de documentación

- Herramientas consideradas: Sphinx y Mkdocs.
- Herramienta elegida: Sphinx.

He decidido utilizar el generador de documentación *Sphinx* ya que es mucho más completo que *MkDocs*. Además de soportar el lenguaje de marcado ligero *Markdown* es compatible con *reStructuredText*. Esta compatibilidad hace que sea posible usar ambos lenguajes en un mismo proyecto *Sphinx*. Además, con el uso del conversor Pandoc, toda la documentación generada a partir de ambos lenguajes se puede exportar a multitud de formatos, entre los que se encuentra LaTeX.

*Markdown* es un lenguaje muy conocido debido a que es utilizado en plataformas como *GitHub* o *StackOverflow*. Fue creado para generar contenido de una manera sencilla de escribir y fácil de leer. Permite además convertir el texto marcado en documentos *XHTML*.

*reStructuredText* presenta también una sintaxis sencilla y de fácil lectura. La principal ventaja respecto a *Markdown* es que permite elaborar expresiones más complejas sin el uso de librerías/aplicaciones externas.

LaTeX es el estándar de facto para la publicación de documentos científicos. Permite la creación de documentos con una alta calidad tipográfica. Utiliza TeX como motor a la hora de darle formato a los documentos.

## 4.11. Herramientas de integración continua

### Compilación y Despliegue

- Herramientas consideradas: GitHub Actions, Travis CI y Jenkins.
- Herramienta elegida: GitHub Actions.

Para aplicar la integración continua al proyecto se ha decidido utilizar *GitHub Actions*. El principal motivo de esta elección es que todas sus funciones se encuentran integradas en la propia interfaz de *GitHub*, lo que facilita en gran medida su uso. Además, permite reutilizar código elaborado por otros usuarios de la comunidad en los flujos de trabajo (*workflows*) personales.

*Workflows* del proyecto: GitHub Actions.

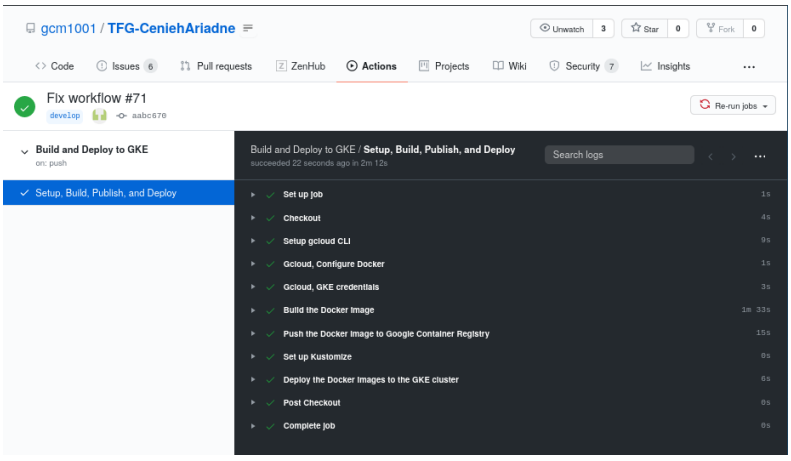


Figura 4.13: Ejecución de uno de los flujos de trabajo del proyecto.

### Calidad del código

- Herramientas consideradas: Codacy, Codecov y CodeClimate.
- Herramienta elegida: Codacy.

La opción escogida ha sido *Codacy* ya que, de entre las tres propuestas, es la que está más enfocada a la revisión de código automatizada, que es lo se estaba buscando. Da soporte a todos los lenguajes que se han utilizado en el proyecto ( *PHP*, *HTML*, *JavaScript* y *CSS* ). Además, el proceso de configuración no se hace nada pesado gracias a que se puede llevar a cabo desde su propia interfaz gráfica. Entre sus configuraciones más utilizadas están la exclusión de ficheros, la activación/desactivación de patrones de código, la selección de ramas y la gestión de integraciones.

*Dashboard* del proyecto: Codacy.

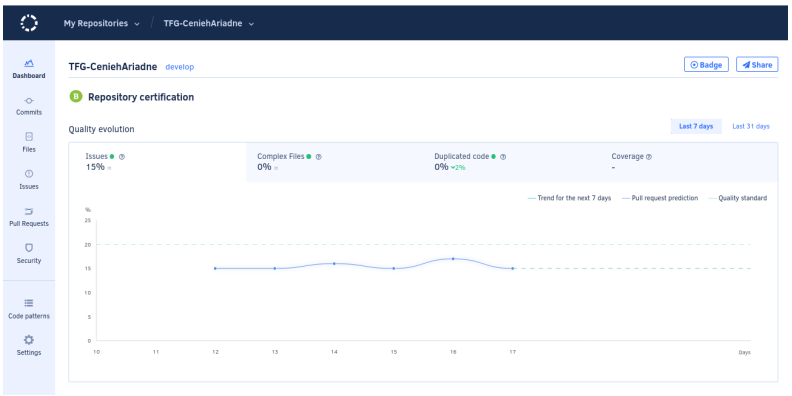


Figura 4.14: Panel principal de Codacy asociado al proyecto.

### Documentación continua

Read the Docs es una plataforma web que facilita la tarea de documentar productos *software* automatizando la compilación, versionado y hospedaje de los ficheros generados por la herramienta de documentación *Sphinx*. El proceso es muy sencillo, basta con alojar la documentación *Sphinx* en un repositorio, realizar un *commit* sobre este y, automáticamente, se actualizan los cambios en la documentación alojada en *readthedocs.org*. Presenta múltiples formatos de exportación y permite configurar múltiples aspectos (traducciones, variables de entorno, reglas de automatización, etc.). Todos estos servicios se ofrecen de forma gratuita.

*Host* de la documentación del proyecto: ReadTheDocs.

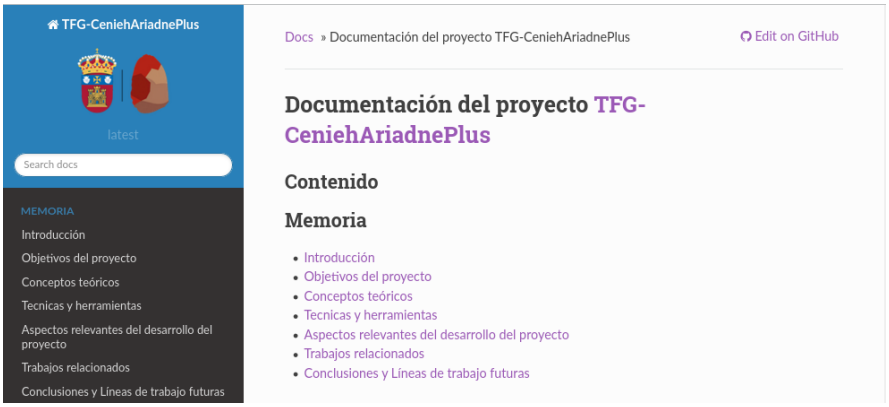


Figura 4.15: Página principal de la documentación del proyecto.

## 4.12. Herramienta para crear diagramas

- Herramientas consideradas: Draw - LibreOffice, SmartDraw y Draw.io.
- Herramienta elegida: Draw.io.

*Draw.io* es una herramienta gratuita de diseño que permite crear y compartir diagramas *on-line*, es decir, sin necesidad de instalar programa alguno. Presenta una interfaz elegante y fácil de utilizar desde la cual podemos hacer uso de sus múltiples funciones como, por ejemplo, importar imágenes, añadir objetos UML, exportar e importar proyectos en diversos formatos, etc.

## 4.13. Herramientas de comunicación

### *Microsoft Teams*

A través de Microsoft Teams se han llevado a cado las reuniones de cada *sprint*. *Teams* viene integrado en el paquete de *Microsoft Office 365*, por lo que es un servicio que puede ser adquirido por el personal de la UBU. Ofrece una gran cantidad de funcionalidades relacionadas con la comunicación como, por ejemplo, la creación de chats personalizados (individuales/grupales, públicos/privados, etc.), compartición de pantalla, integración de aplicaciones externas (Stream, Excel, etc.) o introducción de efectos de cámara (efectos de fondo, filtros, etc.).

### *Zoom*

Zoom es la herramienta de comunicación con la que se han llevado a cabo las reuniones tanto con el CENIEH como con *ARIADNEplus*. Al igual que la herramienta *Microsoft Teams*, permite realizar videollamadas y reuniones virtuales con multitud de funcionalidades extra.

## 4.14. Otras herramientas

### *Docker*

La tecnología Docker permite desplegar una aplicación distribuida y empaquetarla junto a todas sus dependencias y librerías en un uno o varios "objetos" denominados contenedores o *containers*. Estos pueden ser ejecutados en cualquier servidor Linux, aumentando así la flexibilidad y portabilidad de nuestra aplicación.

Imagen *Docker* del proyecto: DockerHub

### *Google Cloud*

Google Cloud es una plataforma creada por la compañía *Google* desde la que puedes acceder a numerosos servicios relacionados con el desarrollo web. Alguno de sus servicios son: *Cloud Computing*, *Networking*, *Data Storage*, *Data Analytics*, *Machine learning*, etc.

### *GKE – Google Kubernetes Engine*

Google Kubernetes Engine (GKE) proporciona un entorno desde donde puedes implementar, administrar y escalar aplicaciones en contenedores mediante la infraestructura de *Google*. El entorno de GKE consta de varias máquinas (en particular, instancias de *Compute Engine*) que se agrupan para formar un clúster.

Aplicación del proyecto desplegada en el clúster: ubucenh.es

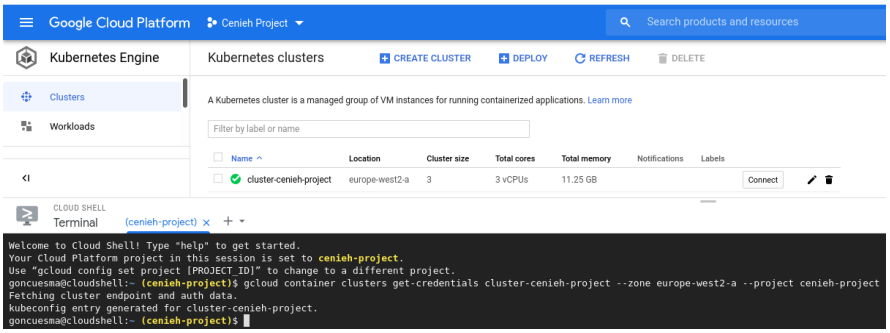


Figura 4.16: Vista del panel de administración del clúster en *Google Cloud*.

### *Kubernetes*

Kubernetes es una plataforma *open-source* que permite automatizar los procesos relacionados con la implementación, administración y escalabilidad de contenedores. He decidido utilizar este orquestador (*orchestrator*) para desplegar mi aplicación en la nube (*Google Cloud*) por la gran cantidad de ventajas que ofrece como, por ejemplo, autoreparación de contenedores, utilización de *secrets* o despliegues y rollbacks automáticos.

### *Kustomize*

Kustomize es una herramienta que permite operar sobre objetos de la plataforma *Kubernetes* a través de un archivo de personalización. Facilita en gran medida la creación de entornos que requieren más de un contenedor.

# Aspectos relevantes del desarrollo del proyecto

A continuación se van a mostrar los aspectos más relevantes del desarrollo del proyecto, justificando las decisiones tomadas y dando visibilidad a los problemas encontrados.

## 5.1. Ciclo de vida del proyecto

En Mayo de 2019, tuvo lugar una reunión entre las diferentes partes involucradas, UBU y CENIEH, donde se sentaron las bases del proyecto. A esta le siguieron muchas otras donde se fueron detallando distintos aspectos del trabajo a realizar. Además, dos de los miembros de la UBU participaron en un *workshop* que tuvo lugar en Pisa, Italia, donde se adquirieron nuevos conocimientos acerca del proyecto *ARIADNEplus* y se estableció un primer contacto entre la UBU y los demás participantes del proyecto.

El 01 de febrero de 2020 se inició este trabajo con el objetivo de llevar a cabo el proceso de integración de los datos del *CENIEH* en *ARIADNEplus*. Adicionalmente, se propuso implantar en el*CENIEH* una infraestructura *software* mediante la cual sus investigadores fueran capaces de llevar a cabo las tareas de integración por sí solos.

El proyecto tiene una duración total de 6 meses, lo que significa que el 01 de agosto de 2020 dará por concluido el periodo de colaboración entre el CENIEH y la UBU.

Las fases en las que se puede dividir el proyecto son:

- **Investigación**: en esta fase se realiza un estudio previo del proyecto *ARIADNEplus*, así como de los conjuntos de datos del CENIEH que

están involucrados en el proceso de integración.

- **Desarrollo**: a lo largo de esta fase se ejecutan todas las tareas relacionadas con el diseño, desarrollo e implementación de la infraestructura *software*.
- **Integración**: en esta fase se integran los datos del CENIEH en *ARIADNEplus* haciendo uso tanto de la infraestructura *software* implementada como de los servicios ofrecidos por *ARIADNEplus*.

## 5.2. Investigación

Durante las primeras semanas de trabajo, se llevó a cabo un estudio exhaustivo del proyecto *ARIADNEplus*, especialmente del **proceso de integración** al que cada miembro debía someter sus datos. A medida que se iban aprendiendo nuevos aspectos, se comprobaba su compatibilidad con los datos propuestos por el *CENIEH*, anotando en todo momento los problemas que pudieran surgir.

### Proceso de integración

*ARIADNEplus* no agrega ni mueve datos de los sistemas de información de los miembros del proyecto, solo añade **metadatos** a los conjuntos de datos que son mantenidos y gestionados por cada miembro. Este tipo de información debe adoptar un **esquema o modelo** para poder ser representado a través de catálogos o repositorios. En el caso de *ARIADNEplus*, cuentan ya con un esquema denominado **AO-Cat** (**A**RIADNE **O**ntology **- Cat**alog), diseñado exclusivamente para el proyecto.

Cada miembro del proyecto, incluído el *CENIEH*, cuenta desde un principio con sus propias colecciones de metadatos. Esto puede considerarse un problema ya que, como se ha comentado anteriormente, este tipo de información se representa a través de un esquema, el cual será distinto en cada uno de los miembros. Ante esta situación, al no coincidir el esquema de origen (e.g. esquema sedimentos, esquema ratón pérez, esquema litoteca...) con el de destino (AO-Cat), sería imposible poder representar los datos del origen en el catálogo.

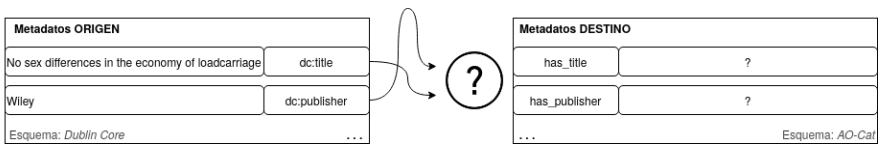


Figura 5.17: Conflicto entre esquemas de metadatos distintos.

Además, un esquema está formado por un conjunto de elementos, cada uno de los cuales está sujeto a unas determinadas **reglas**. Por ejemplo, podría especificarse el tipo de dato que almacena (*string*, *date*, etc.) o considerarse como imprescindible (no nulo). Este hecho aumenta aún más la complejidad del proceso de integración ya que cada miembro cuenta con sus propios elementos y con sus propias reglas.

Para dar solución a todos estos problemas, *ARIADNEplus* propone dividir el proceso de integración en 6 fases.

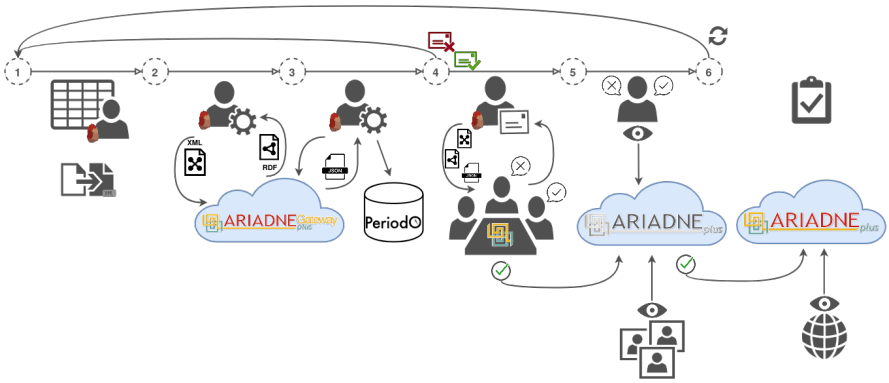


Figura 5.18: Representación gráfica de las fases en las que se divide el proceso de integración.

1. **Confirmación**: se confirman las colecciones de metadatos que serán agregadas y, además, se indica a qué categoria de datos de *ARIADNEplus* pertenecen.
2. **Transformación**: una vez estén listos los metadatos de origen, se genera un fichero de definición de mapeo que permita transformar el esquema de metadatos de origen al esquema objetivo (*AO-Cat*).
3. **Enriquecimiento**: se mejora la calidad de los metadatos sometiendo los datos a un proceso de enriquecimiento utilizando como fuentes externas el vocabulario *Getty AAT* y *PeriodO*.
4. **Importación**: habiendo completado las tres fases anteriores, se ejecuta el proceso de importación de los metadatos.
5. **Simulación de publicación**: con los metadatos de origen ya importados en la base de datos, se realiza una simulación de publicación.
6. **Publicación**: si los resultados obtenidos en la fase 5 son favorables, se lleva a cabo la publicación de los metadatos en el catálogo oficial.

### Confirmación

En esta fase se confirman qué colecciones de datos serán integradas en el proyecto. Además, se debe indicar a qué categoría de datos de *ARIADNE* pertenecen.

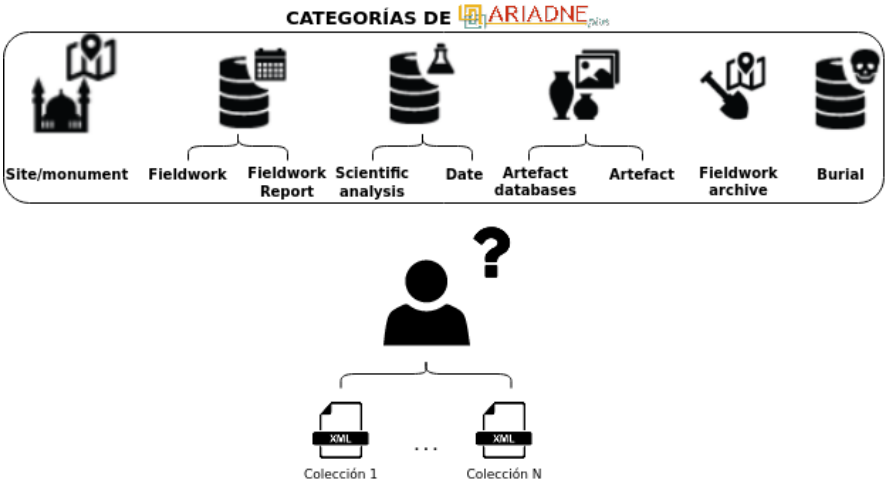


Figura 5.19: Fase de confirmación.

Afortunadamente, los datos propuestos por el *CENIEH* se obtuvieron con suma facilidad, sin ningún tipo de reticencia por su parte.

En la siguiente tabla se indican las principales características de cada una de las colecciones de datos confirmadas para la integración con *ARIADNEplus*. Además, se especifica la categoría *ARIADNE* a la que corresponden.

| Colección | Núm. Registros | Docs asociados | Campos | Formato | Categoría ARIADNE |
|---|---|---|---|---|---|
| Anatomía Comparada | 571 | Sí | SIGNA_CENIEH, Clase, Orden, Familia Género, Especie, Sigla de campo, Elemento, Sexo, Adulto Localidad, Municipio, Provincia, Pais, Tipo de objeto | CSV | Scientific analysis |
| Litoteca | 99 | Sí | Afloramiento, Sigla, Localización, Datum, X, Y, Z, Acceso, Tipo de Afloramiento, Tipo de roca, Depositante, Muestra física, Lámina delgada, Laboratorio geología CENIEH, Fotografías, Otros datos, Topografía | CSV | Scientific analysis |
| Ratón Pérez | 1323 | Sí | Sigla, Individuo, Sexo, Edad, Pieza, Superior/inferior, Lado, Conservación, Consolidado, Pegado, Observaciones, Localización, Fecha MicroCT, Archivo mCT, Proyecto Amira, localización, No Imágenes, kv/mA, Vxl, Size, Filter, Fotos mCT | CSV | Scientific analysis |
| Sedimentos | 7695 | No | ReferenciaBolsa, ReferenciaCaja, Yacimiento, Nivel, Cuadro, Z, Situacion, FechaRecogida, FechaAlmacen, FechaProcesando | CSV | Scientific analysis |
| CIR | 1853 | Sí | Dublin Core terms | CSV | Scientific analysis & Fieldwork Reports |

Tabla 5.1: Colecciones de metadatos propuestas por el *CENIEH* para la integración con *ARIADNEplus*.

### Transformación

Para evitar el problema mostrado en la Figura 5.17, *ARIADNEplus* pone a disposición de sus miembros la **herramienta *X3ML Mapping Tool***, disponible en el entorno virtual *ARIADNEplus Mappings* del portal *ARIADNEplus Gateway* de *D4Science*. Está compuesta por un conjunto de microservicios, de código abierto, que siguen el modelo de referencia *SYNERGY* [16] para la transmisión y agregación de datos.

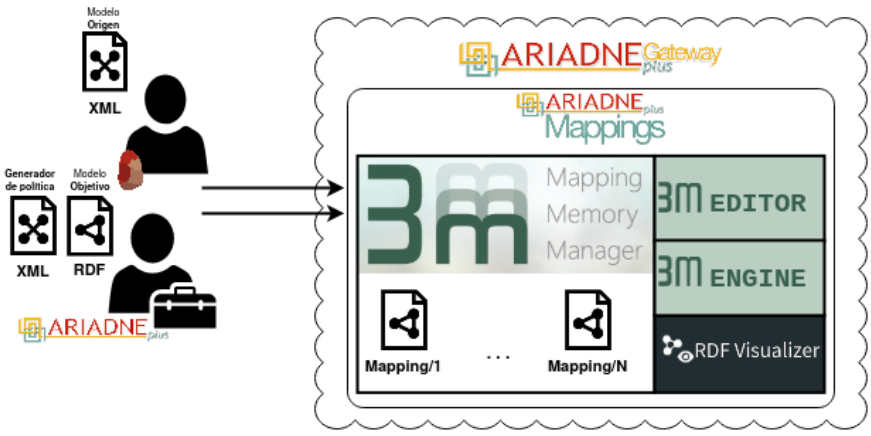


Figura 5.20: Fase de transformación.

Los componentes clave de este servicio son:

- ***3M – Mapping Memory Manager***: herramienta utilizada para la gestión de archivos de definición de mapeo. Proporciona una serie de acciones administrativas que ayudan a los proveedores de datos a administrar sus archivos de definición de mapeo.


Figura 5.21: Vista de la herramienta *Mapping Memory Manager - 3M*.

- ***3M Editor***: provee la interfaz que permite crear asignaciones entre los elementos del esquema de metadatos a mapear y el esquema objetivo.

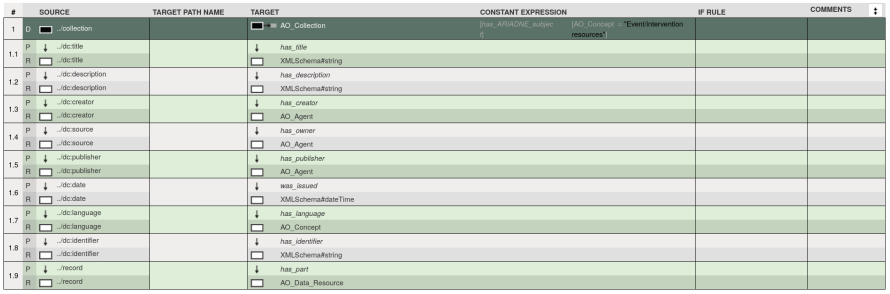


Figura 5.22: Vista de la herramienta *3M Editor*.

- ***X3ML Engine***: ejecuta la transformación de los elementos de origen al formato de destino. Tomando como entrada los metadatos de origen (en formato *XML*), la descripción de las asignaciones existentes en el fichero de definición de mapeo y el archivo que contiene las políticas para la generación de *URIs*, es responsable de transformar el documento original en un documento *RDF* válido que corresponda al archivo *XML* de entrada con las asignaciones y políticas indicadas.
- ***RDF visualizer***: permite, de una forma rápida, inspeccionar los documentos transformados.

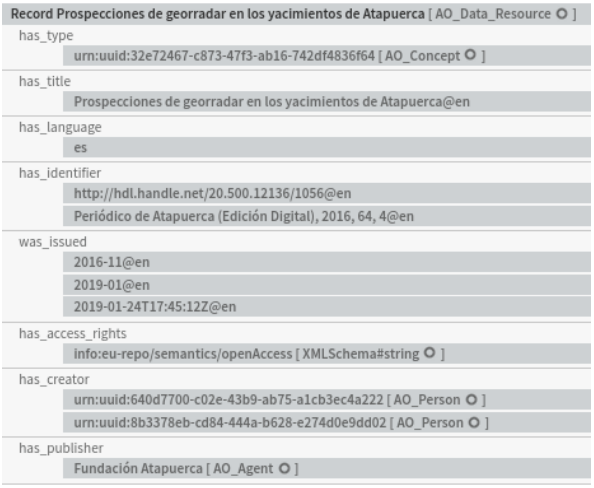


Figura 5.23: Vista de la herramienta *RDF visualizer*.

Esta herramienta toma un **papel decisivo** en el proceso de integración ya que permite transformar el modelo de origen al esquema de metadatos utilizado en *ARIADNEplus* (*AO-CAT*).

A continuación se van a describir los **principales retos** a los que nos hemos tenido que enfrentar durante esta segunda fase:

- Todos los conjuntos de datos propuestos por el CENIEH están en formato *CSV*. Esto supone un problema ya que ***ARIADNEplus* solo trabaja con ficheros *XML***, es decir, no cuenta con ningún método de importación que tolere archivos *CSV*.
- **Los conjuntos de datos del CENIEH**, a excepción de la colección del *CIR*, están dispuestos de forma irregular, es decir, **no siguen ningún esquema estandarizado**. Esto implica que para cada conjunto de datos, se necesita hacer un fichero de definición de mapeo distinto, lo que no es para nada eficiente.
- En el esquema objetivo, los **elementos** pueden ser opcionales u **obligatorios**. Los elementos opcionales no suponen ningún problema ya que pueden quedar vacíos, sin embargo, los elementos obligatorios requieren la existencia de un elemento en el modelo de origen que pueda sustituirlo, es decir, que tenga el mismo significado. Esta regla supone un reto para el CENIEH ya que muchos de estos elementos obligatorios no cuentan con un elemento apto en las colecciones de datos propuestas.
- **El contenido** almacenado en cada elemento del esquema objetivo **ha de tener un formato específico**. Por ejemplo, el contenido del elemento *has_language*, responsable de indicar el idioma en el que está dispuesto el objeto al que referencia, debe cumplir con el estándar ISO639-1 o

ISO639-2. Por tanto, el elemento asignado en el origen debe seguir el mismo formato.

### Enriquecimiento

En ocasiones, los metadatos por si solos no son lo suficientemente precisos o claros como para describir una determinada característica del objeto al que se refieren. En el caso de la arqueología, existen multitud de conceptos con un alto grado de complejidad que necesitan ser explicados en detalle. Por este motivo, *ARIADNEplus* propone enriquecer los metadatos haciendo uso del vocabulario *Getty AAT* y del cliente *PeriodO*.

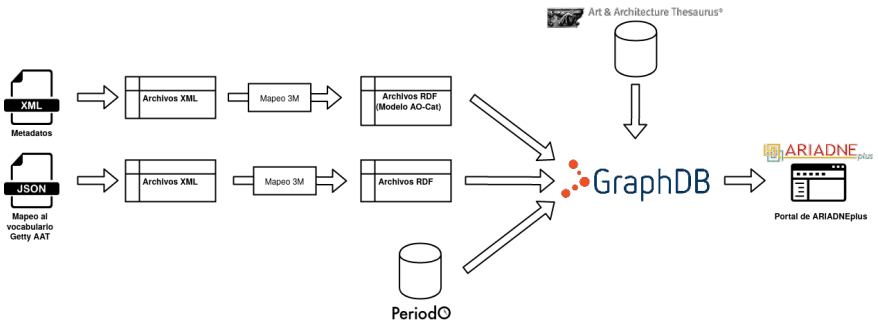


Figura 5.24: Enriquecimiento de metadatos.

En la Figura 5.24 se muestra el flujo de datos del proceso de enriquecimiento de metadatos. Por una parte, vemos un archivo .json, el cual se obtiene a través de la **herramienta *Vocabulary Matching Tool***. Esta es otra de las herramientas que se pueden encontrar en el entorno de investigación virtual *ARIADNEplus Mappings*. Permite mapear el vocabulario utilizado en el documento de origen al vocabulario *Getty AAT*.

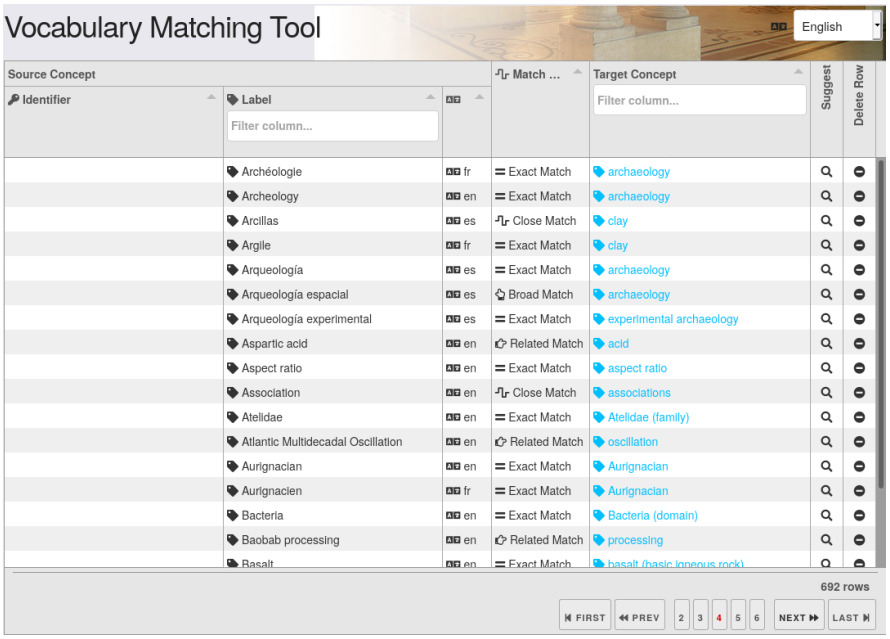


Figura 5.25: Vista de la herramienta *Vocabulary Matching Tool*.

El archivo generado por esta herramienta (*.json*) define las relaciones entre los términos del vocabulario de origen y los términos del vocabulario *Getty AAT*. Desde el catálogo de *ARIADNEplus*, todos aquellos términos que tengan una asociación definida, serán hiperenlaces que apunten al término *Getty AAT* asociado.

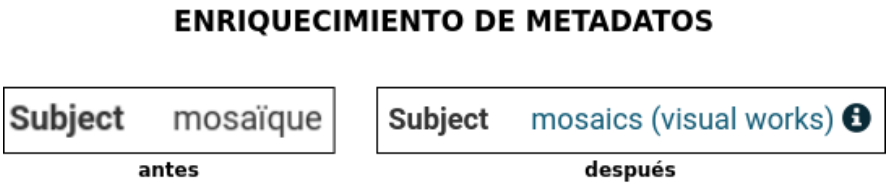


Figura 5.26: Vista de un metadato (*Subject*) antes y depués de ser enriquecido.

Además, en la Figura 5.24 vemos representada la base de datos donde **PeriodO** almacena sus registros. Para aportar información adicional a los periodos existentes en nuestros datos, debemos publicar en el cliente de *PeriodO* nuestra propia colección de periodos. De esta forma, *ARIADNEplus* podrá recoger desde la BD de *periodO* nuestra colección para, posteriormente, establecer una relación entre los periodos de un lado y de otro. Al igual que con el vocabulario, todos los periodos que tengan una asociación definida, serán hiperenlaces que apunten al objeto de *periodO*.


Figura 5.27: Vista de un metadato (*Dating*) antes y depués de ser enriquecido.

En esta fase se encontraron varios **inconvenientes**:

- Muchos de los términos existentes en los conjuntos de datos del CENIEH no están presentes en el vocabulario *Getty AAT*. Por este motivo, solo se pudo enriquecer una pequeña parte del conjunto total.
- Para poder publicar la colección en *periodO*, se requería determinar la autoridad de los periodos, es decir, indicar de donde procedían. Desde el CENIEH no me pudieron facilitar ese dato ya que lo desconocían. Por este motivo, no se pudo llevar a cabo la publicación y por ende no se enriquecerieron los periodos.

### Importación

El sistema de importación de *ARIADNEplus*, conocido como *ARIADNEplus Aggregator*, se basa en el kit de herramientas de *software* D-Net (implementado y mantenido por *ISTI-CNR* [22]), que proporciona funciones integradas que permiten recopilar conjuntos de metadatos a través de múltiples métodos. Está disponible en el portal *ARIADNEplus Gateway*, sin embargo, su acceso está restringido a los coordinadores del proyecto.

Las principales opciones de importación son:

1. ***OAI-PMH*** [23]: es un protocolo estándar para el intercambio de metadatos. A través de este método *ARIADNEplus* puede recolectar los metadatos almacenados en un repositorio de forma remota.
2. ***SFTP*** [36] : es un protocolo de transmisión de ficheros. Esta opción es algo engorrosa ya que debe existir un archivo XML por recurso, es decir, no puedes agrupar varios registros en un mismo fichero XML. Los socios son responsables del servidor SFTP. Se admiten modos de autentificación.
3. ***FTP(S)*** [34]: es otro protocolo de transferencia de ficheros. Presenta las mismas características de importación que SFTP.
4. **Workspace**: se pueden subir directamente los metadatos en el *workspace* de *D4Sciente* (*ARIADNEplus Gateway*). Cada socio tiene su propia carpeta donde puede ir almacenando los documentos *XML* (metadatos) que desee importar.

Dado que este sistema es inaccesible para la mayoría de los miembros (incluido el *CENIEH*), se debe escoger una de esas opciones y comunicársela al coordinador responsable. Este nos indicará como proceder y, una vez realizada la importación, se deben facilitar tres datos:

1. Qué ficheros (*.xml*) de los importados se desean publicar.
2. Cuál es el identificador del fichero de definición de mapeo (e.g. *Mapping/621*) almacenado en la herramienta *X3ML Mapping* que transformará el esquema de metadatos presente en tus ficheros al esquema *AO-Cat*.
3. Opcionalmente, el enlace a tu colección de *periodO* y/o el fichero de definición de mapeo (*.json*) del vocabulario.

Los conjuntos de datos del *CENIEH* están almacenados de forma local, exceptuando el *CIR*. Por ello, de entre todas las opciones posibles, la única forma válida de importar metadatos sería a través del *Workspace*.

### Simulación de publicación

Una vez establecida la comunicación con el coordinador responsable del proceso de importación, se debe esperar a su respuesta. Dependiendo del contenido de esta, se pueden tomar dos caminos:

1. Nos indican que todas las partes del proceso (metadatos, mapeo, enriquecimiento) son correctos. En tal caso, los metadatos propuestos estarían ya disponibles desde el portal fantasma de *ARIADNEplus*. Este es idéntico al original con la única diferencia de que sólo tienen acceso los miembros del proyecto.
2. Nos indican que alguna parte del proceso no es correcta. Ante esta situación, se debe volver hacia atrás en el proceso de integración para solventar los conflictos señalados por el coordinador.

### Publicación

Si en la fase previa se ha obtenido una respuesta satisfactoria, el miembro que inició el proceso de integración sería ya capaz de observar el resultado final sobre el catálogo fantasma.

Una vez visualizado el resultado, deberá comunicarse de nuevo con el responsable de la importación para indicarle sus impresiones. Se pueden dar dos situaciones:

1. El resultado es favorable. Ante esta situación el coordinador lleva a cabo la publicación de los datos en el portal real.
2. No se esperaba el resultado obtenido. En tal caso, se deben mantener las conversaciones hasta llegar a una solución.

En el caso de que todo haya salido según lo planeado, el proceso de integración para los conjuntos de datos publicados habría concluído. Existe la posibilidad de reactivar este proceso en el caso de que se deseen actualizar o añadir datos, sin embargo, hay que tener en cuenta que cualquier cambio en la estructura de los datos supondría tener que volver a realizar el proceso desde cero (primera fase).

## 5.3. Desarrollo

Recordemos que en la fase anterior se anotaron todos los aspectos relevantes del proceso de integración, incluyendo además los problemas de incompatibilidad encontrados entre dicho proceso y los datos propuestos por el CENIEH.

Es en esta fase cuando se aplican las competencias y los conocimientos adquiridos a lo largo del grado con el objetivo de desarrollar una infraestructura *software* que sea capaz de guiar a los operarios del CENIEH en el proceso de integración y, además, resuelva los problemas mencionados en la fase anterior.

### *Omeka* como aplicación principal

Desarrollar desde cero una infraestructura *software* que cumpliera con todos los requisitos propuestos no era viable debido a la limitación temporal del proyecto. Por este motivo, se decidió utilizar *software* de terceros que cumpliera con un mínimo de **requisitos**:

- Permitir la **gestión de metadatos**: los archivos de información involucrados son metadatos, por tanto, se necesita un sistema que permita realizar todo tipo de tareas de gestión sobre este tipo de datos.
- Disponer de **herramientas de importación y exportación**: los datos de origen necesitarán ser importados a la plataforma para realizar sobre ellos las operaciones oportunas. Una vez gestionados, deberán ser exportados para someterlos al proceso de integración.
- Ser **software libre**: este requisito era fundamental ya que, para poder adaptar la infraestructura a las necesidades del proyecto, se debe tener total libertad a la hora de ejecutar, copiar, distribuir, estudiar, modificar y mejorar el *software*.

Se consideraron varios productos *software* para acabar escogiendo Omeka Classic. Una de las características que hacen de la aplicación una magnífica plataforma para el proyecto es su **escalabilidad**. Gracias a su sistema de **complementos** o *plugins* cualquier programador tiene la posibilidad de adaptarla a sus necesidades individuales sin tener que modificar el código base de la aplicación. También cuenta con un sistema de temas para personalizar la estética del área pública de la aplicación (*frontend*).

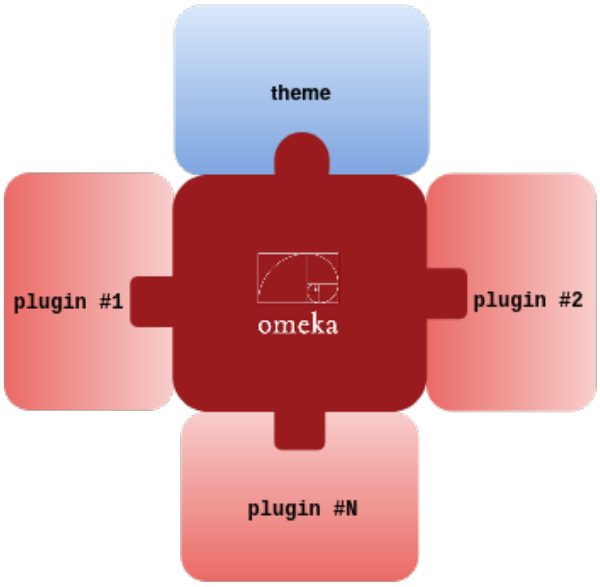


Figura 5.28: Complementos y temas en *Omeka*.

Actualmente, *Omeka* cuenta con una gran cantidad de *plugins* disponibles, tanto en su página oficial como en GitHub. Esto es posible gracias a la extensa comunidad de usuarios que le respalda.

Parte de estos *plugins* se han podido reutilizar para adaptar la infraestructura a las necesidades del proyecto, sin embargo, se han tenido que desarrollar nuevos *plugins* para cubrir requisitos específicos. Además, se han llevado a cabo modificaciones sobre alguno de los *plugins* externos utilizados.

Por este motivo, la mayoría de las tareas realizadas en esta fase están directamente relacionadas con la creación y modificación de *plugins* para *Omeka*.

### Complementos o *plugins*

Los complementos o *plugins* son capaces de añadir nuevas funcionalidades a *Omeka* gracias a que esta tiene implementado un sistema de ganchos o *hooks*. Estos nos permiten acoplar código en puntos específicos del flujo de ejecución de la aplicación, evitando así tener que alterar el código base de esta.

Dentro de la aplicación se pueden encontrar dos tipos distintos de *hooks*: ***hooks* de acción** y **filtros (*filters*)**.

### *Hooks* de acción.

Este tipo de *hook* permite añadir la ejecución de funciones externas en puntos de ejecución específicos.

Por ejemplo, en el caso de que se quiera introducir un formulario en una página de *Omeka*, se debería utilizar el *hook* alojado en dicha página para ejecutar la función encargada de imprimir el código HTML del formulario. En este ejemplo, la función no retornaría nada ya que se limita a imprimir código, y es que en este tipo de *hooks* la función no tiene por qué devolver nada.

En los archivos de *Omeka* se pueden localizar estos *hooks* buscando la función *fire_plugin_hook()*. Una vez encontrada, desde el *plugin* que estamos desarrollando, haciendo uso de la interfaz *Omeka_Plugin_AbstractPlugin*, bastaría con añadir este *hook* a la lista *_hooks* e instanciar el método correspondiente, el cual siempre tiene la nomenclatura *hook<NombreDelHook>()*.

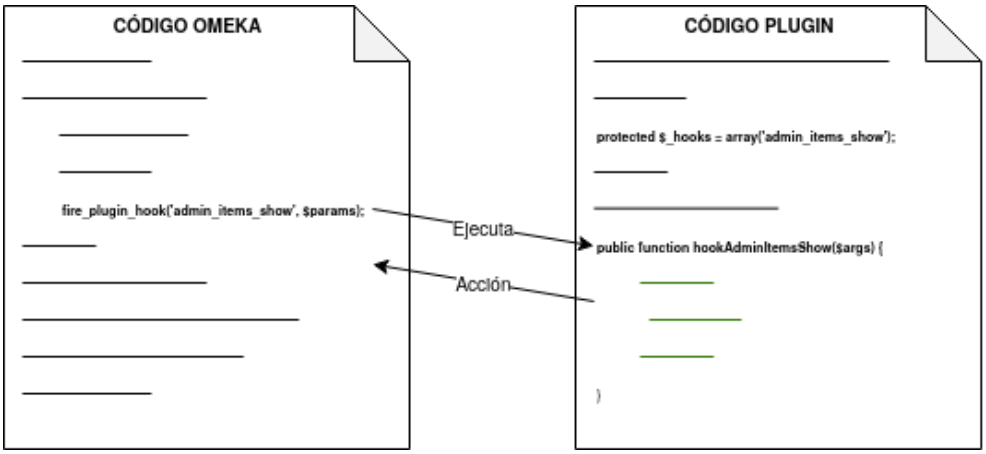


Figura 5.29: Ejemplo de *action hook*.

En el ejemplo vemos como *fire_plugin_hook()* tiene dos parámetros de entrada, el primero indica el nombre del *hook* y el segundo almacena los argumentos de entrada que tendrá la función que almacena la acción.

**Filtros (*Filters*).**

Los filtros permiten, al igual que los *hooks* de acción, ejecutar funciones externas en puntos específicos de la aplicación. Sin embargo, el objetivo de estos es algo distinto ya que no pretenden modificar código sino alterar los datos de una determinada variable.

Las funciones deben tener un parámetro de entrada y otro de salida de forma que, desde el interior de la función, se procesa el valor de entrada y se devuelve el valor resultante.

En los archivos de *Omeka* se pueden localizar estos *filters* buscando la función *apply_filters()*. Una vez encontrada, existen dos formas de usar ese filtro:

1. Utilizando la interfaz *Omeka_ hooksPlugin_ AbstractPlugin* es posible utilizar el filtro añadiendo su nombre a la lista *_filters*. A continuación, se añadiría el método público con el nombre *filter* seguido del nombre del filtro.

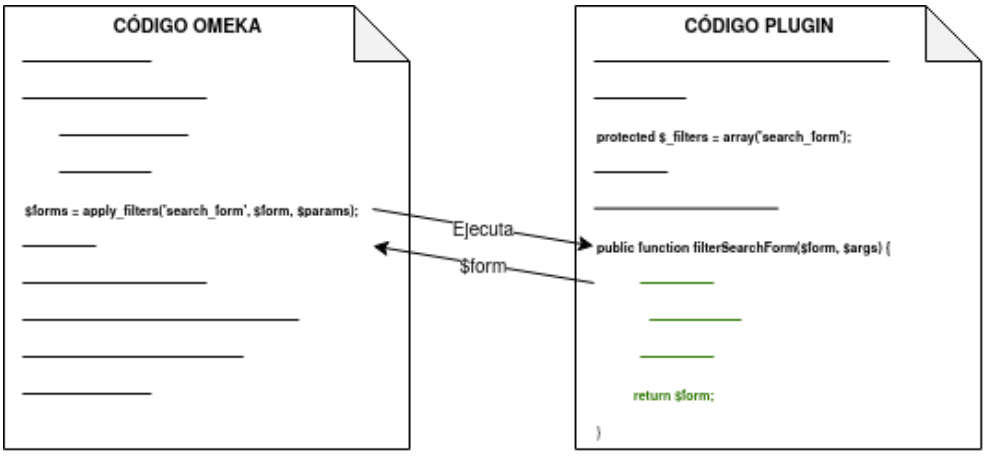


Figura 5.30: Ejemplo de *filter hook*.

2. Utilizando el método *add_filter()*, se puede utilizar el filtro pasando como primer parámetro el nombre del filtro implicado y como segundo parámetro la función que se ejecutará. En este caso, el nombre de la función es personalizable. Además, se puede pasar un tercer parámetro para indicar la prioridad de nuestro *hook*, es decir, si existiera más de un *plugin* utilizando ese mismo filtro, se ejecutaría la función de cada uno en función de su prioridad, de mayor a menor prioridad. Por defecto, todos los *filtros* de cada *plugin* tienen una prioridad de 10, por lo que el orden

de ejecución se determina por la fecha de instalación, de más antiguos a más nuevos.

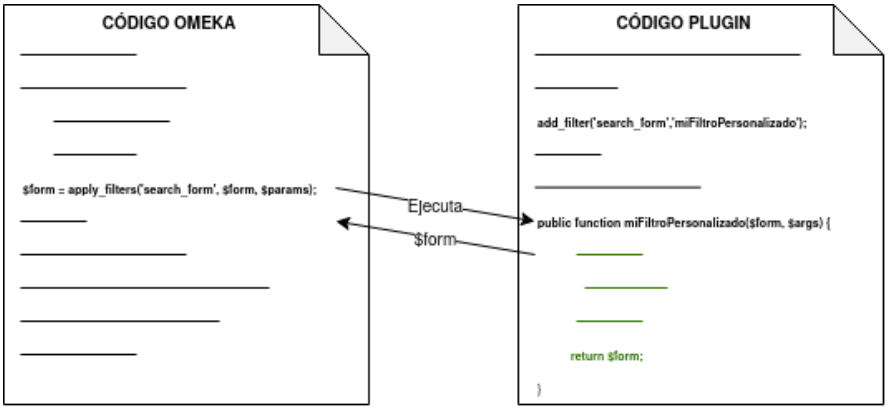


Figura 5.31: Segundo ejemplo de *filter hook*.

## Entornos de trabajo

Durante la fase de desarrollo, se ha trabajado sobre dos entornos visibles en el repositorio alojado en *GitHub*:

- **Entorno de desarrollo**: se actualiza al cometer cambios sobre la rama *develop*. Permite llevar un seguimiento diario del estado de la aplicación durante el desarrollo de la misma.
- **Entorno de producción**: se actualiza al cometer cambios sobre la rama *main*. En su interior se puede encontrar una versión estable de la aplicación. El intervalo de tiempo de actualización gira entorno a las dos semanas.

## Despliegue de la infraestructura

Para llevar a cabo el despliegue de la infraestructura se ha utilizado la herramienta *GitHub Actions*. Dependiendo del entorno de trabajo, se ha procedido de una manera u otra:

### Servidor de desarrollo

A través de la herramienta *GitHub Actions* se ha automatizado el despliegue de la infraestructura sobre el servidor de desarrollo. A esta técnica se la conoce como despliegue continuo.

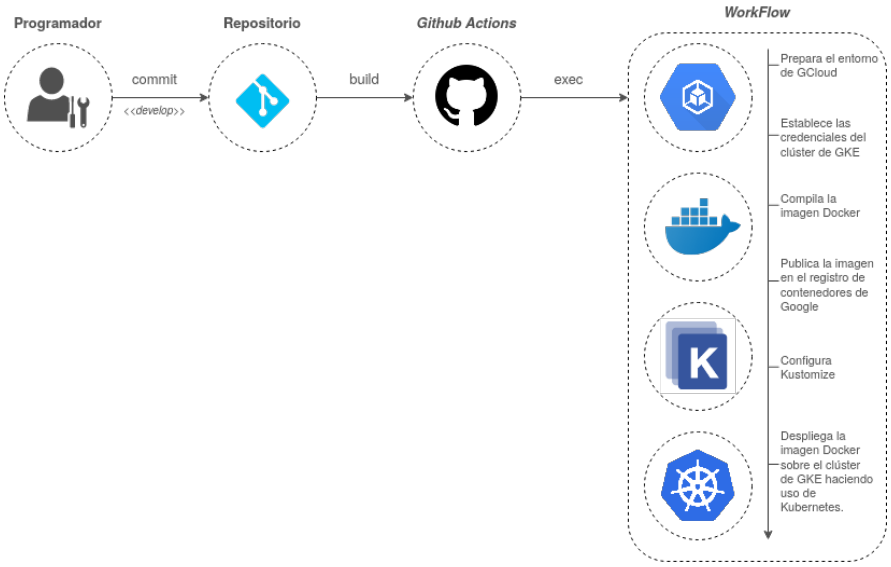


Figura 5.32: Despliegue continuo de la aplicación.

En la *Figura* 5.32 vemos representado el proceso mediante el cual se lleva a cabo el despliegue. Con el *workflow* configurado y alojado en la ruta *.github/workflows* de mi repositorio en GitHub, cuando ejecuto un *push* sobre la rama *develop*, si los cambios cometidos afectan a cualquier carpeta que no sea la de */docs*, se ejecutan las acciones correspondientes al despliegue de mi aplicación, las cuales se pueden apreciar en la imagen.

### Servidor de producción

Sobre el servidor de producción no se ha podido automatizar el despliegue debido a que el acceso a este era privado, es decir, no se podía establecer comunicación desde el exterior sin previa conexión al VPN del CENIEH y el posterior acceso vía *ssh* al servidor.

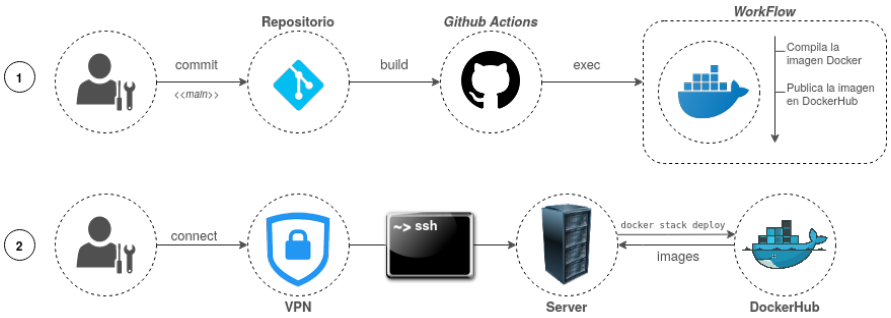


Figura 5.33: Despliegue "semi-continuo" de la aplicación.

Como solución a este inconveniente, se automatizó por separado la compilación y publicación de la imagen *Docker* asociada a nuestra aplicación. De esta manera, cada vez que se cometía un cambio sobre la rama *main*, se ejecutaba dicho proceso, actualizando la imagen publicada en el repositorio de *DockerHub*. Finalizado el proceso, se accedía al servidor de producción y se desplegaba manualmente la infraestructura. Durante el despliegue, se recogían las imágenes desde *DockerHub*, incluyendo la imagen actualizada de nuestra aplicación.

# Trabajos relacionados

Algunos socios del proyecto *ARIADNEplus* han adoptado una solución muy similar a la propuesta en el presente proyecto, es decir, han hecho uso de aplicaciones *software* de terceros para la gestión de sus metadatos y las han adaptado según sus necesidades. A continuación, se muestran aquellos casos que guardan una mayor relación con el proyecto.

## 6.1. Casos similares

### *Fasti Online*

Fasti Online [26] es un proyecto liderado por la Asociación Internacional de Arqueología Clásica (AIAC) [1] y el *Center for the Study of Ancient Italy* (CSAI) [9] de la Universidad de Texas, Austin [24]. Su principal objetivo es proporcionar una infraestructura *software* que permita almacenar, gestionar y publicar registros relacionados con la arqueología.

Para tal fin, han utilizado como base la aplicación *software* denominada ARK. Esta es una aplicación web que provee servicios como la gestión, compartición y transformación (mapeo) de metadatos. Además, la aplicación es de código abierto, lo que significa que es personalizable y extensible.

La incorporación de *Fasti Online* al proyecto *ARIADNE* y, posteriormente, al proyecto *ARIADNEplus*, ha impulsado la implementación de nuevas funcionalidades sobre la aplicación *ARK* como, por ejemplo, la integración de datos espaciales, nuevos mecanismos de búsqueda y otros servicios web como, por ejemplo, el protocolo *OAI-PMH*.

### *CONICET*

El Consejo Nacional de Investigaciones Científicas y Técnicas [8] (*CONICET*) es el principal organismo dedicado a la promoción de la ciencia y la tecnología en Argentina. Este, al igual que el *CENIEH*, es una de las nuevas incorporaciones al proyecto *ARIADNEplus* y, como tal, han tenido que adaptarse para satisfacer los requisitos del proyecto.

La solución planteada por este organismo es muy similar a la del presente proyecto. Están desarrollado una infraestructura *software* que permita a los operarios del *CONICET* gestionar y publicar sus conjuntos de datos adoptando un esquema de metadatos compatible con *ARIADNEPlus*. La aplicación *software* que han decidido adaptar ha sido Dspace 5.5 . Se puede acceder a su infraestructura desde el siguiente enlace .

### *DANS*

*DANS* [13] (*Data Archiving and Networked Services*) es una institución de los Paises Bajos cuya misión principal es proporcionar las herramientas necesarias a investigadores para hacer que sus datos sean accesibles, interoperables y reutilizables.

Esta organización es responsable del desarrollo y mantenimiento del repositorio digital DCCD. Este entró en funcionamiento en 2011 y es considerado como la principal red de metadatos arqueológicos/históricos existente en Europa. Dentro del *DCCD*, laboratorios belgas, daneses, holandeses, alemanes, letones, polacos y españoles publican contenido fruto de la investigación de, entre otros: sitios arqueológicos (incluidos paisajes antiguos), construcciones, pinturas, esculturas e instrumentos musicales.

Esta organización participó en el proyecto *ARIADNE* y, actualmente, forma parte del proyecto *ARIADNEplus*. Con el objetivo de mejorar la integración europea de datos dendrocronológicos ofrecen, de forma gratuita, la misma solución *software* empleada en su proyecto *DCCD*, la cual es compatible con el proyecto *ARIADNE*. Está disponible en Github .

## 6.2. Comparativa entre soluciones *software*

| Caraterísticas | Omeka | ARK | DSpace | DCCD |
|---|---|---|---|---|
| Tipo de aplicación | Web | Web | Web | Web |
| Lenguaje de programación principal | PHP | PHP | Java | Java |
| Gestión de metadatos | ✓ | ✓ | ✓ | ✓ |
| Importación masiva de metadatos | ✓ | ✓ | ✓ | ✓ |
| Exportación masiva de metadatos | ✓ | ✓ | ✓ | ✓ |
| Edición masiva de metadatos | ✓ | × | × | × |
| Múltiples formatos de localización | ✓ | × | × | × |
| Cobertura temporal | × | ✓ | × | ✓ |
| Protocolo *OAI-PMH* | ✓ | ✓ | ✓ | × |
| Soporte para *ARIADNEplus* | ✓ | × | × | × |
| Transformación de metadatos | ✓ | ✓ | × | × |
| Sistema de usuarios | ✓ | ✓ | ✓ | ✓ |
| Almacenamiento de ficheros | ✓ | ✓ | ✓ | × |
| Asistencia técnica gratuita | ✓ | × | × | × |
| Interfaz pública | ✓ | ✓ | ✓ | ✓ |
| Interfaz intuitiva | ✓ | × | ✓ | × |
| Sistema de *plugins* | ✓ | × | ✓ (*) | × |
| Sistema de plantillas | ✓ | × | × | × |
| Comunidad de usuarios activa | ✓ | × | ✓ | × |
| Manuales de documentación detallados | ✓ | × | × | × |
| Última actualización | 2020 | 2018 | 2020 | 2015 |

Tabla 6.2: Comparativa de las características de las aplicaciones propuestas por cada socio.

*(*) Servicio de pago.*

Basándonos en el contenido de la Tabla 6.2, se listarán los puntos fuertes y débiles que presenta la aplicación del proyecto frente a las propuestas de los otros socios.

### Puntos fuertes

- Gran parte de la configuración de la aplicación puede realizarse desde la interfaz gráfica, sin necesidad de modificar ficheros internos que requieran un mínimo de conocimiento de la estructura interna de la aplicación, como pasa en aplicaciones como *ARK* o *DCCD*. Esto facilita en gran medida las labores de configuración de la aplicación.

- Al requerir una infraestructura *LAMP* para su despliegue, la instalación de la aplicación es relativamente sencilla en comparación con las otras aplicaciones. Además, gracias al presente proyecto, es posible instalar la aplicación a través de tecnologías como *Docker* o *Kubernetes*, facilitando aún más su despliegue.
- De entre todas las soluciones mostradas es, sin duda, la más sencilla y segura de adaptar y personalizar. Esto es gracias al sistema de complementos (*plugins*) y plantillas (*themes*) que incorpora.
- Gracias a las labores de desarrollo llevadas a cabo en el presente proyecto, dispone de herramientas de apoyo para la integración de conjuntos de datos en *ARIADNEplus*.
- La comunidad de usuarios con la que cuenta *Omeka Classic* es superior a la de sus competidores. Muchos usuarios comparten sus propios desarrollos, tanto complementos como plantillas, de forma que estos pueden ser reutilizados o incluso mejorados por otros usuarios. Además, existe un foro desde donde los expertos de *Omeka*, incluídos los líderes del proyecto, brindan soporte técnico gratuito a otros usuarios de la aplicación.
- La documentación disponible es, tanto para usuarios como para desarrolladores, la más clara y detallada de todas las aplicaciones mostradas.
- Actualmente el proyecto *Omeka* continúa en desarrollo, es decir, siguen saliendo nuevas actualizaciones con mejoras y funcionalidades nuevas para la aplicación. Sin embargo, otros proyectos como *ARK* o *DCCD* están obsoletos.

### Puntos débiles

- Actualmente, no dispone de ningún mecanismo que identifique aquellos metadatos cuyo contenido sea un periodo temporal (e.g. "1190 BCE") y los procese de tal forma que estos sean mostrados dentro de una línea temporal y a su vez puedan ser un criterio aislado de búsqueda.
- No posee las ventajas que proporciona el lenguaje de programación *Java* utilizado tanto en *DSpace* como en *DCCD*. Este es más rápido y presenta un mejor rendimiento al ser un lenguaje compilado. Además, posee una estructura más ordenada y es mucho más seguro que *PHP*.

# Conclusiones y Líneas de trabajo futuras

En este último apartado, se exponen las conclusiones extraídas tras un breve análisis objetivo del trabajo realizado. Además, se proponen nuevas perspectivas para las posibles líneas de trabajo futuras.

## 7.1. Conclusiones

A continuación se listan las conclusiones más relevantes que se han obtenido tras finalizar el proyecto.

- En cuanto a los objetivos generales del proyecto, se han cumplido ambos. Los operarios del *CENIEH* cuentan con una aplicación que les facilita el proceso de integración de sus datos en *ARIADNEplus* y, además, se ha conseguido integrar una de las colecciones propuestas.
- Durante la fase de investigación del proyecto, se han aprendido multitud de técnicas y conocimientos nuevos relacionados con la creación e implementación de metadatos así como de las herramientas y aplicaciones utilizadas para su gestión y distribución.
- El ser parte de un proyecto internacional como *ARIADNEplus* me ha permitido conocer nuevos métodos de trabajo como, por ejemplo, la utilización de entornos de investigación virtuales (*VREs*). Gracias a estos he podido comunicarme con los demás socios del proyecto, utilizar servicios y herramientas comunes, y compartir recursos digitales de todo tipo.
- El elevado número de participantes del proyecto y la complejidad del mismo han permitido comprobar la necesidad de una continua comunicación y la dificultad a la hora de obtener los requisitos.

- En la parte de desarrollo del proyecto se han aplicado muchos de los conocimientos adquiridos durante el Grado de Ingeniería Informática relacionados fundamentalmente con Bases de Datos, Diseño y Desarrollo Avanzado de Aplicaciones Web basadas en componentes y aplicando buenas prácticas de agilidad de integración y despliegue continuo. Asímismo, se han utilizado otras técnicas de desarrollo que han requerido un estudio especial como *PHP*, *Zend Framework*, *Hooking*, etc.
- El desarrollo basado en componentes web para la adaptación de la aplicación propuesta ha supuesto una experiencia totalmente nueva que me ha permitido conocer cómo funcionan este tipo de aplicaciones.
- En el proyecto se han aplicado técnicas de integración continua que hasta el momento desconocía. Estas han permitido agilizar muchas de las tareas involucradas en el desarrollo del proyecto, afectando positivamente a la calidad del código producido y a la depuración de errores.

## 7.2. Líneas de trabajo futuras

Se pueden tomar dos caminos distintos para mejorar la aplicación propuesta:

1. Desarrollar nuevos complementos (*plugins*) que añadan nuevas funcionalidades.
2. Extender la funcionalidad de los complementos propuestos en este proyecto.

A continuación se exponen las funcionalidades que pueden resultar interesantes añadir en la plataforma.

- Dar soporte a los periodos temporales que pudieran aparecer dentro del metadato "*Temporal Coverage*". Por ejemplo:
    - ◇ Representar gráficamente a todos los periodos dentro de una linea temporal.
    - ◇ Sugerir periodos temporales del cliente *PeriodO* a la hora de rellenar el metadato "Temporal Coverage". De esta manera, se podrá enriquecer dicho metadato en la fase de integración correspondiente.

En cuanto a las posibles mejoras de los complementos:

- Complemento *ARIADNEplus Tracking*: introducir nuevas funciones en alguna de las fases de los *tickets*.

  - ◇ *Fase 1*: Poder editar los ítems desde la misma ventana, sin necesidad de desplazarse al gestor de ítems.
  - ◇ *Fase 3*: Previsualizar la colección de *periodO* indicada por el usuario y poder adjuntar el fichero de definición de mapeo desde la misma ventana.
  - ◇ *Fase 4*: Previsualizar los ítems publicados en el portal fantasma de *ARIADNEplus* a partir del enlace *SPARQL*.

- Complemento *Geolocation*: introducir localizaciones con áreas poligonales (hasta ahora solo se pueden simples o rectangulares).

- Complemento *Bulk Metadata Editor*: introducir nuevas acciones de edición como, por ejemplo, poder asignar más de dos valores a un mismo metadato.

- Complemento *OAI-PMH Harvester*: poder programar recolecciones de metadatos en determinados intervalos de tiempo.

- Complemento *AutoDublinCore*: dado que la localización de los datos es imprescindible, crear un sistema que en caso de que el metadato que se encarga de ello ("*Spatial Coverage*") se encuentre vacío, busque en el contenido de los demás metadatos (e.g. *Title*, *Description*, etc.) una localización y, en caso de encontrarla, actualizar el contenido del metadato con dicha localización.

- Traducir todos los complementos desarrollados en este proyecto a otros idiomas.

# Bibliografía


[1] AIAC. Associazione internazionale di archeologia classica, 2020. [Online; Accedido 17-Junio-2020].

[2] ARIADNE. Ariadne project eu — foundation., 2020. [Online; Accedido 09-Junio-2020].

[3] ARIADNEplus. Ariadne infrastructure, 2020. [Online; Accedido 09-Junio-2020].

[4] Leonardo Candela, Donatella Castelli, and Pasquale Pagano. Virtual research environments: An overview and a research agenda. *Data Science Journal*, advpub, 2013.

[5] CENIEH. Sobre el cenieh, 2020. [Online; Accedido 09-Junio-2020].

[6] CIDOC. Cidoc crm — conceptual reference model, 2020. [Online; Accedido 09-Junio-2020].

[7] CIR. Cenieh institutional repository, 2020. [Online; Accedido 09-Junio-2020].

[8] CONICET. Consejo nacional de investigaciones científicas y técnicas., 2020. [Online; Accedido 17-Junio-2020].

[9] CSAI. Center for the study of ancient italy, 2020. [Online; Accedido 17-Junio-2020].

[10] D4Science. Ariadneplus gateway, 2020. [Online; Accedido 10-Junio-2020].

[11] D4Science. D4science infraestructure, 2020. [Online; Accedido 10-Junio-2020].

[12] D4Science. Workspace, 2020. [Online; Accedido 10-Junio-2020].

[13] DANS. Data archiving and networked services., 2020. [Online; Accedido 17-Junio-2020].

[14] DCMI. *Dublin Core Metadata Initiative*, 2020. [Online; Accedido 12-Junio-2020].

[15] DCMI. *Terms*, 2020. [Online; Accedido 12-Junio-2020].

[16] Martin Doerr, Achille Felicetti, Gerald de Jong, Konstantina Konsolaki, Barry Norton, Dominic Oldman, Maria Theodoridou, and Thomas Wikman. THE SYNERGY REFERENCE MODEL OF DATA PROVISION AND AGGREGATION. pages 1–65, 2016.

[17] Achille Felicetti, Carlo Meghini, Julian Richards, and Maria Theodoridou. Towards the AO-Cat Ontology. pages 1–8, 2020.

[18] FORTH-ICS. Parthenos entities: Research infrastructure model. 2020. [Online; Accedido 09-Junio-2020].

[19] The Getty Research Institute. Art & architecture thesaurus, 2020. [Online; Accedido 10-Junio-2020].

[20] The Getty Research Institute. Sparql endpoint, 2020. [Online; Accedido 10-Junio-2020].

[21] ISO and IEC. *ISO/IEC 9126-1:2001, Software Engineering – Product Quality, Part 1: Quality Model*. 2001.

[22] ISTI-CNR. Istituto di scienza e tecnologie dell'informazione, 2020. [Online; Accedido 14-Junio-2020].

[23] OAI-PMH. Open archives initiative protocol for metadata harvesting, 2020. [Online; Accedido 14-Junio-2020].

[24] University of Texas at Austin, 2020. [Online; Accedido 17-Junio-2020].

[25] Omeka. Omeka classic, 2020. [Online; Accedido 12-Junio-2020].

[26] Fasti Online, 2020. [Online; Accedido 17-Junio-2020].

[27] Ontotext. Graphdb technology, 2020. [Online; Accedido 10-Junio-2020].

[28] PARTHENOS Project, 2020. [Online; Accedido 10-Junio-2020].

[29] Ken Schwaber and Jeff Sutherland. The scrum guide, 2020. [Online; Accedido 12-Junio-2020].

[30] José Antonio Senso and Antonio de la Rosa Piñero. El concepto de metadato. Algo más que descripción de recursos electrónicos. *Ci. Inf.*, 32(2):95–106, 2003.

[31] Wikipedia. Geolocalización — wikipedia, the free encyclopedia, 2020. [Online; Accedido 12-Junio-2020].

[32] Wikipedia. Hooking — wikipedia, the free encyclopedia, 2020. [Online; Accedido 12-Junio-2020].

[33] Wikipedia. Lamp — wikipedia, the free encyclopedia, 2020. [Online; Accedido 12-Junio-2020].

[34] Wikipedia. Protocolo de transferencia de archivos — wikipedia, the free encyclopedia, 2020. [Online; Accedido 14-Junio-2020].

[35] Wikipedia. Sistema de gestión de contenidos — wikipedia, the free encyclopedia, 2020. [Online; Accedido 12-Junio-2020].

[36] Wikipedia. Ssh file transfer protocol — wikipedia, the free encyclopedia, 2020. [Online; Accedido 14-Junio-2020].

[37] Wikipedia. Wsg84 — wikipedia, the free encyclopedia, 2020. [Online; Accedido 12-Junio-2020].