

FONDO EUROPEO DE DESARROLLO REGIONAL (FEDER)

Una manera de hacer Europa

Servicios Web de ASIO

Álvaro Palacios y Esteban Sota

RIAM Intelearning Lab – GNOSS

alvaropalacios@gnoss.com y estebansota@gnoss.com



HĒRCULES

Introducción

- ☐ Resumen funcional de Hércules ASIO
- ☐ Descripción de los componentes de Hércules ASIO
- ☐ Infraestructura Ontológica y Arquitectura Semántica
- ☐ Descripción de los componentes
- ☐ Reutilización de los datos
- ☐ SPARQL
- ☐ Casos prácticos



FONDO EUROPEO DE DESARROLLO REGIONAL (FEDER)

Una manera de hacer Europa

Hércules ASIO. Resumen funcional



Resumen funcional de Hércules ASIO

- Crear un grafo de conocimiento para los proyectos de investigación que se realizan en las universidades pertenecientes a la CRUE
- Identificar unívocamente a cada entidad, sin generar duplicidades
- Generar un sistema de carga
- Crear un sistema automatizado de integración de nuevas fuentes de datos, basado en OAI-PMH
- Tener un sistema de descubrimiento y reconciliación de entidades
- Crear un sistema de validación de entidades, que verifique que éstas siempre cumplen la ontología ROH.

Resumen funcional de Hércules ASIO

- Crear un benchmark de bases de datos RDF (RDF Stores), que permita establecer a cada universidad sus propios criterios para decidir que RDF Store elegir
- SPARQL Endpoint agnóstico del sistema RDF que se use.
- Ofrecer un interfaz Web que permita a los usuarios navegar por el grafo de conocimiento (Linked Data Server)
- Crear un interfaz Web que permita administrar todas las funcionalidades desarrolladas
- Crear una interfaz por la cuál se puedan añadir nuevas páginas de manera dinámica, consultando cualquiera de los API del ASIO

Hércules ASIO. Infraestructura Ontológica y Arquitectura Semántica

Hércules ASIO. Infraestructura Ontológica

Definición del modelo ontológico del grafo de conocimiento de la investigación universitaria.

Creación una Red de Ontologías que pueda ser usada para describir con fidelidad y alta granularidad los datos del dominio GI: **Red de Ontologías Hércules (ROH)**.

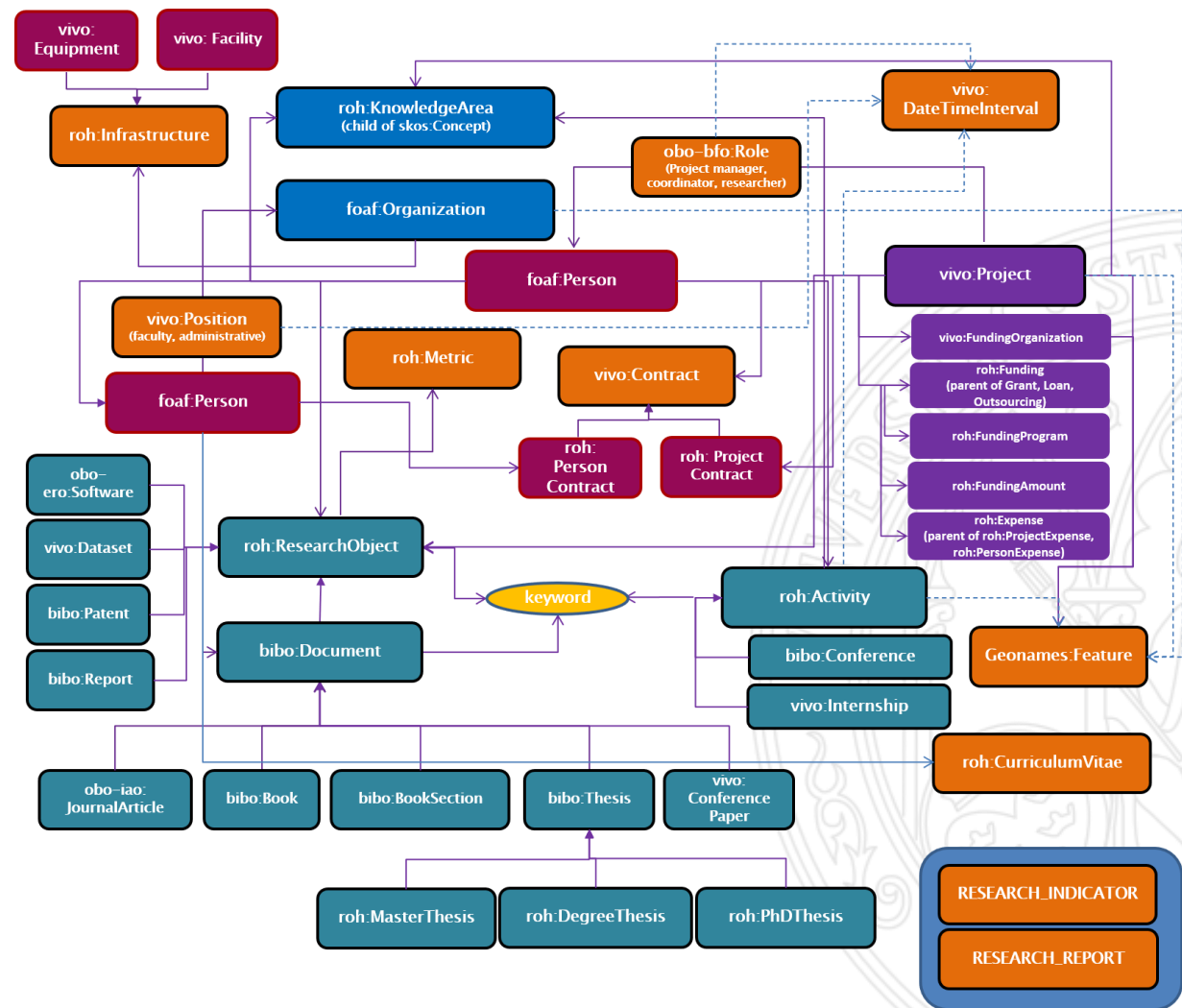
Hércules ASIO. Infraestructura Ontológica

Especificación de los requerimientos a cumplir por la ROH:

- ❑ **Análisis de escenarios:** mapa del conocimiento nacional, cuadros de mando flexibles, búsqueda de socios, selección de grupos de investigación, etc.
- ❑ **Análisis de funcionalidades del SGI:** proyectos, convocatorias y ayudas, producción científica, CV, contratos y patentes, ética, etc.
- ❑ **Análisis de entidades:** taxonomías, atributos, relaciones y fuentes de datos por cada entidad identificada, como investigador/a, proyecto, fuente de financiación, etc.
- ❑ **Requisitos funcionales y no funcionales de ASIO:** principios de Open Data, principios FAIR, URIs persistentes, multilingüismo, interoperabilidad con otras ontologías, integración con fuentes de información existentes, enlace con grafos de conocimiento externos (linked data), mantenibilidad y licencia Open Source.

ROH. Red de Ontologías Hércules

- Descripción formal sobre un modelo de dominio concreto. En este caso, Investigadores, proyectos de investigación, etc.



Hércules ASIO. Arquitectura Semántica. Tecnología

En el desarrollo de ASIO hemos utilizado las siguientes tecnologías y técnicas de desarrollo:

- ☐ **.Net Core**. Un framework informático administrado, gratuito y de código abierto para los sistemas operativos Windows, Linux y macOS, que sustituye a .NET Framework aportando capacidades multiplataforma. Desarrollado principalmente por Microsoft bajo la Licencia MIT.
- ☐ Arquitectura **SOA (Service Oriented Architecture)**.
 - ☐ Favorece la mantenibilidad y la reutilización.
 - ☐ APIs desarrollados siguiendo el principio “Eat Your Own Dog Food”. ASIO no llama a librerías, sino a los APIs desarrollados e instalados.
 - ☐ Ofrece un modo sencillo de reutilización de componentes de ASIO.
- ☐ Los componentes de ASIO también se pueden reutilizar como librerías en otros desarrollos, pero no es lo que recomendaríamos.

Hércules ASIO. Service Oriented Architecture SOA

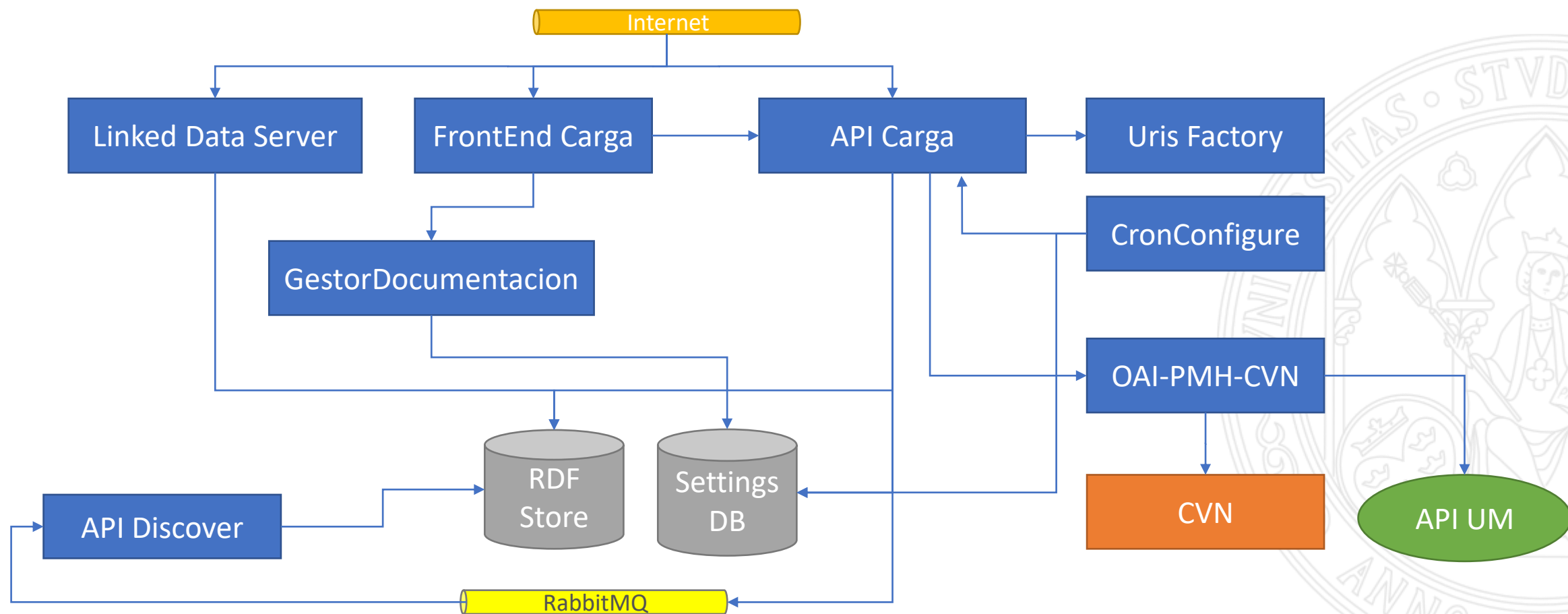
Es un patrón de arquitectura de diseño de software en el que los componentes proporcionan servicio a otros componentes a través de un protocolo de comunicación, habitualmente una red y especialmente a través de HTTP y HTTPS.

https://en.wikipedia.org/wiki/Service-orientation_design_principles

Hércules ASIO. Descripción de los componentes



Arquitectura ASIO



Aplicaciones ASIO: API CARGA

Es un servicio web que contienen 4 controladores, utilizados cada uno de ellos para su propio propósito:

- etlController: Contiene los procesos ETL (Extract, Transform and Load) necesarios para la carga de datos.
- repositoryController: Contiene los procesos necesarios para la gestión de los repositorios OAI-PMH (creación, modificación, eliminación...).
- syncController: Contiene los procesos necesarios para la ejecución de las sincronizaciones.
- ValidationController: Contiene los procesos necesarios para la gestión de las validaciones (creación, modificación, eliminación...). La carpeta Validaciones contiene información sobre los shapes SHACL definidos para validar.

Aplicaciones ASIO: API DISCOVER

El API Descubrimiento es accesible como servicio Web y ofrece unas funciones que son parte del proceso de carga. Estas funciones se dividen en 3 grupos:

- Reconciliación de entidades, que evita la duplicación de entidades y puede añadir datos desde otros nodos ASIO, a través de las entidades incorporadas en el nodo Unidata.
- Descubrimiento de enlaces, que genera enlaces hacia datasets externos, puede incorporar datos en ASIO y ofrece información de ayuda en la reconciliación de entidades.
- Detección de equivalencias, que genera equivalencias semánticas.

Aplicaciones ASIO: CRON CONFIGURE

Es un api para la gestión y configuración del programado de tareas, tanto de ejecución recurrente como ejecución única sobre los repositorios configurados.

Este api contiene 3 controladores:

- **JobController:** Con los métodos disponibles en este controlador se pueden crear una sincronización, obtener las tareas y volver a ejecutar una tarea ya ejecutada.
- **RecurringJobController:** Mediante los métodos de este controlador se puede crear una tarea recurrente especificando el patrón de repetición, borrar las tareas recurrentes y obtener dichas tareas, así como obtener las tareas que se han ejecutado a partir de una tarea recurrente.
- **ScheduledJobController:** Este controlador sirve para crear tareas que se ejecuten en una determinada fecha, obtener este tipo de tareas y eliminarlas

Aplicaciones ASIO: CRON CONFIGURE

Mediante estos controladores se pueden crear los siguientes tipos de tareas:

- **Tarea recurrente:** Una tarea se le denomina recurrente cuando tiene una repetición o recurrencia a través de un patrón. Por ejemplo que se ejecute todos los lunes a las 8 de la mañana.
- **Tarea programada:** Una tarea se le denomina programada cuando está configurada para ejecutarse en un momento en el futuro.
- **Tarea/ tarea de única ejecución:** Una tarea se le denomina de ejecución única cuando se ejecuta una sola vez en el momento de su creación o cuando se ha ejecutado ya, con esto último lo que se quiere decir que una tarea recurrente cada vez que se ejecuta crea tareas de única ejecución al igual que una tarea programada cuando pasa a ser ejecutada ejecuta una tarea de ejecución única.

Aplicaciones ASIO: FRONT END CARGA

- Este módulo constituye el interfaz Web de administración de las cargas de datos en la plataforma Hércules ASIO.
- Esta aplicación web realizada mediante el patrón MVC (modelo-vista-controlador) esta formado por diferentes controladores que se comunican con los diferentes APIs creados en este proyecto.
- Estos controladores dividen su ámbito en la gestión de errores, de los repositorios, de los shapes, de las tareas, la gestión de la factoria de uris, etc.

Aplicaciones ASIO: FRONT END CARGA


- **ErrorController:** Gestiona los errores para devolver una página 404 o 500 según la excepción que se genera.
- **JobController:** Controlador que gestiona las operaciones (listado, obtención de tareas, ejecución, ...) que se realizan en la web con cualquier tipo de tarea.
- **RepositoryConfigController:** Controlador encargado de gestionar las operaciones con los repositorios. Creación, modificación, eliminación y listado.
- **ShapeConfigController:** Encargado de gestionar las operaciones con los shapes. Creación, modificación, eliminación y listado.

Aplicaciones ASIO: FRONT END CARGA


- **UrisFactoryController:** Este controlador es el encargado de gestionar todas las tareas que se pueden realizar con el API de UrisFactory, tanto la obtención de uris como las operaciones con el fichero de configuración de las uris.
- **CMSController:** Controlador encargado de gestionar las páginas creadas por los usuarios.
- **TokenController:** Encargado de obtener los tokens de acceso para los diferentes apis.
- **CheckSystemController:** Se encarga de verificar que el api cron y el api carga funcionan correctamente.

Hércules ASIO; GESTOR DOCUMENTACIÓN

ASIO cuenta con una gestión de web documental que permite publicar páginas webs que informen del nodo ASIO de la universidad, con contenido estático y dinámico, obtenido éste último del API de consulta o del SPARQL Endpoint.



GOBIERNO DE ESPAÑA



MINISTERIO DE CIENCIA E INNOVACIÓN


FONDO DE DESARROLLO REGIONAL (FEDER)

HÉRCULES


Información sobre Hércules Grafo de Conocimiento Administración

ARQUITECTURA SEMÁNTICA E INFRAEST

Hércules ASIO - Arquitectura Semántica Infraestructura Onto



HÉRCULES
Semántica de Datos de Investigación de Universidades



crue
Universidades Españolas


Proyecto Hércules ASIO - Backend SGI


El objetivo del Proyecto ASIO es adquirir un servicio de I+D que consiste en el desarrollo de soluciones innovadoras para la Universidad de Murcia en relación al reto de Arquitectura Semántica e Infraestructura Ontológica. En concreto, se pretende desarrollar e incorporar soluciones que superen las actualmente disponibles en el mercado ...[ver más](#)

Grafo de


El proyecto ASIO cuenta con un grafo de 47 triples, generados desde el SGI de la Universidad de Murcia. ...[ver más](#)

El grafo de conocimiento de Hércules ASIO





GOBIERNO DE ESPAÑA



MINISTERIO DE CIENCIA E INNOVACIÓN

FONDO DE DESARROLLO REGIONAL (FEDER)

HÉRCULES

Información sobre Hércules Grafo de Conocimiento Administración

Estadísticas generales Resumen del Grafo ASIO-SGI Entidades del Grafo ASIO-SGI **Datos del Grafo ASIO-SGI**

Datos del Grafo de Conocimiento ASIO-SGI

Los datos del grafo pueden consultarse mediante un SPARQL Endpoint, mientras que sus entidades y relaciones están accesibles como datos enlazados y abiertos (Linked Open Data). Esto convierte los datos de ASIO SGI en un conjunto de datos FAIR.

Consultas SPARQL

SPARQL (SPARQL Protocol and RDF Query Language) es el lenguaje de consulta para RDF y también el protocolo con el que se envían las consultas y se reciben los resultados.


Hércules ASIO dispone de un SPARQL Endpoint público, que permite realizar consultas SPARQL sobre los datos del grafo ASIO-SGI y que se encuentra disponible en la siguiente URL:

<http://155.54.239.204:8890/sparql>

Se indican a continuación algunos ejemplos de consultas:

- Obtener los 100 investigadores con más atributos (más datos).
- Obtener los atributos de un investigador.

Linked Data Server



Knowledge Graph

Linked Data Server es un componente de ASIO desarrollado en Net Core que proporciona el servicio de datos enlazados de Hércules ASIO.

Linked Data Server proporciona acceso a los datos de las entidades del grafo para personas y máquinas, que pueden obtener la representación RDF de la entidad si la petición solicita "application/rdf+xml", y cumple la especificación LDP: Hércules Backend ASIO. Evaluación de cumplimiento Linked Data Platform (LDP)

Se puede comprobar el funcionamiento del servidor mediante los

Aplicaciones ASIO: GESTOR DOCUMENTACION

Este módulo se usa para cargar páginas HTML y servir las a través de la web a modo de un gestor de contenidos, en el que los usuarios pueden subir páginas html con una ruta determinada y luego acceder a esas páginas a través de la web. Contiene un único controlador con cuatro acciones:

- Get Page: Que devuelve la lista de páginas sin el html.
- Get Page: Que devuelve una página dada su ruta.
- Post Page: Carga una nueva página o modifica una página existente
- Delete Page: Elimina una página dando su identificador

Aplicaciones ASIO: GESTOR DOCUMENTACION. Ejemplo

```
@model ApiCargaWebInterface.ViewModels.CmsDataViewModel
@using CsvHelper
@using System.Globalization
@using System.IO
@using System.Text
@{
    Layout = "_Layout";
    ViewData["BodyClass"] = "fichaRecurso";
    \tstring result = Model.Results.FirstOrDefault();
    byte[] byteArray = Encoding.UTF8.GetBytes(result);
    MemoryStream stream = new MemoryStream(byteArray);
    var csvReader = new CsvReader(new StreamReader(stream), CultureInfo.InvariantCulture);
    var records = csvReader.GetRecords<PruebaSparql>();
}
```

```
@*<% sparql
select ?type count(?s) as ?count
where
{
    ?s rdf:type ?type
}
group by ?type
/%>*<
```

```
<div class="row">
    <div class="col col-12 col-lg-12 col-contenido">
        <div class="maxCol">
```

Aplicaciones ASIO: GESTOR DOCUMENTACION. Ejemplo

```
<h1>Número de entidades por tipo</h1>
<div class="contenido">
  <div class="grupo grupo-descripcion">
    <div class="items tabla">
      <div class="cabecera">
        <div class="columna">
          <p>Type</p>
        </div>
        <div class="columna">
          <p>Count</p>
        </div>
      </div>
      @foreach (var fila in records)
      {
        <div class="fila">
          <div class="columna principal">
            <p>@fila.type</p>
          </div>
          <div class="columna">
            <p>@fila.count</p>
          </div>
        </div>
      }
    </div>
  </div>
</div>
```

Aplicaciones ASIO: IDENTITY SERVER HERCULES

Este módulo es el encargado de la securización mediante tokens para los APIs que forman el proyecto, valida los tokens de seguridad y proporciona nuevos tokens de acceso.

Aplicaciones ASIO: URIs Factory

Este módulo es el encargado de unificar las URIs de todas las entidades existentes en ASIO. Este API contiene dos controladores.

- Factory: contiene el siguiente método para la generación de URIs
- Schema: ofrece métodos para administrar la configuración de URIs

Aplicaciones ASIO: URIs Factory. Ejemplo

Quiero una URI para un **Investigador** cuyo identificador es **1234**

Su URI sería: <http://graph.um.es/res/person/1234>

Aplicaciones ASIO: Linked Data Server

LINKED DATA SERVER es un componente de ASIO desarrollado en .Net Core que proporciona el servicio de datos enlazados de Hércules ASIO, cumpliendo la recomendación Linked Data Platform (LDP)

En la ejecución del proyecto se ha optado por el desarrollo de un componente propio.

Ejemplo: <http://graph.um.es/res/person/c305b739-a36d-45db-ae87-186600b3cbde>

Aplicaciones ASIO: OAI PMH CVN

OAI PMH CVN es un servicio web basado en OAI-PMH que sirve los datos de los currículums de los investigadores de la Universidad de Murcia en formato RDF y dublin core.

Para ello, este servicio hará uso de dos servicios externos:

- API CVN UM: Api que proveerá de los currículums en formato XML CVN.
- CVN: Servidor HTTP que ofrece un API para convertir XML CVN a RDF ROH.

Aplicaciones ASIO: CVN

Servidor Web que ofrece un API web para convertir XML CVN a tripletas ROH

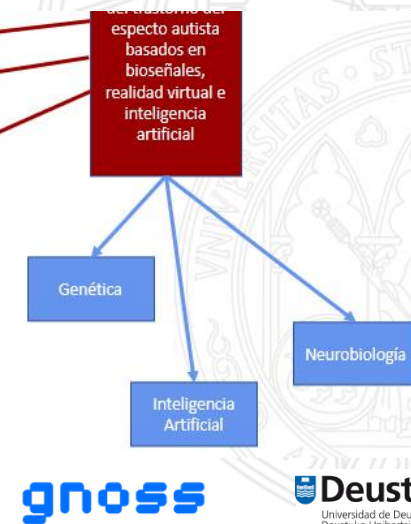
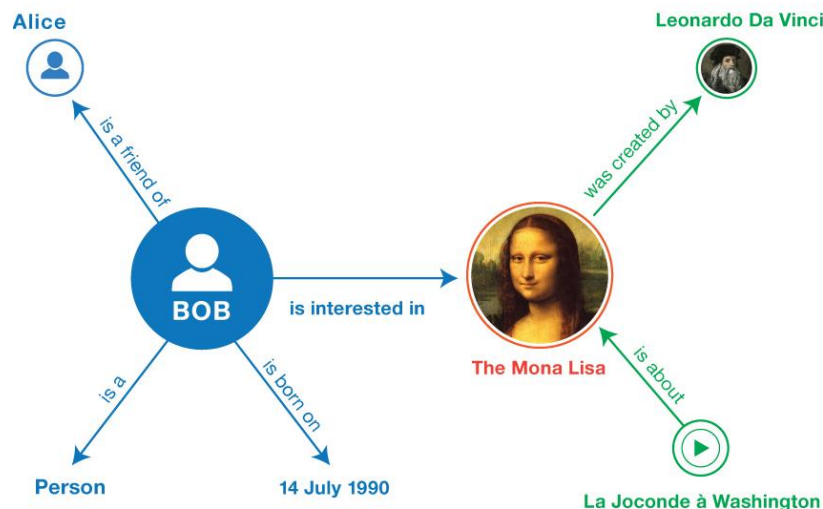
Hércules ASIO. Reutilización de los datos



RDF. Resource Description Framework

RDF identifica conceptos usando identificadores Web (URIs), y describe recursos con propiedades y valores de las mismas. Definiciones:

- ❑ Un Recurso es cualquier cosa que puede tener un URI, como por ejemplo "http://graph.um.es/res/**person**/abfdf688-dc94-4506-a9bd-e89f85db0475"
- ❑ Una Propiedad es un Recurso que tiene un nombre, como "**author**" o "**webpage**"
- ❑ Un Valor de propiedad es el valor de una Propiedad, tal como "**Diego López de Ipiña**" o "**http://www.w3schools.com**" (un valor de propiedad puede corresponder a un recurso).



Hércules ASIO. Reutilización de los datos

ASIO proporciona dos interfaces de reutilización de los datos del grafo de conocimiento modelados por la Red de Ontologías Hércules ROH:

- ❑ **Linked Data Server.** Es el componente que permite la publicación de los datos cargados en el RDF Store como datos abiertos y enlazados (linked open data).
- ❑ **SPARQL Endpoint.** Permite a usuarios y administradores consultar los datos del grafo de conocimiento almacenados en el RDF Store, mediante el lenguaje y protocolo de consultas SPARQL.

Reutilización: Conocer ROH + SPARQL Queries + Linked Data Server

Hércules ASIO. Arquitectura Semántica. Linked Data Server

Linked Data en el RDF St las siguientes

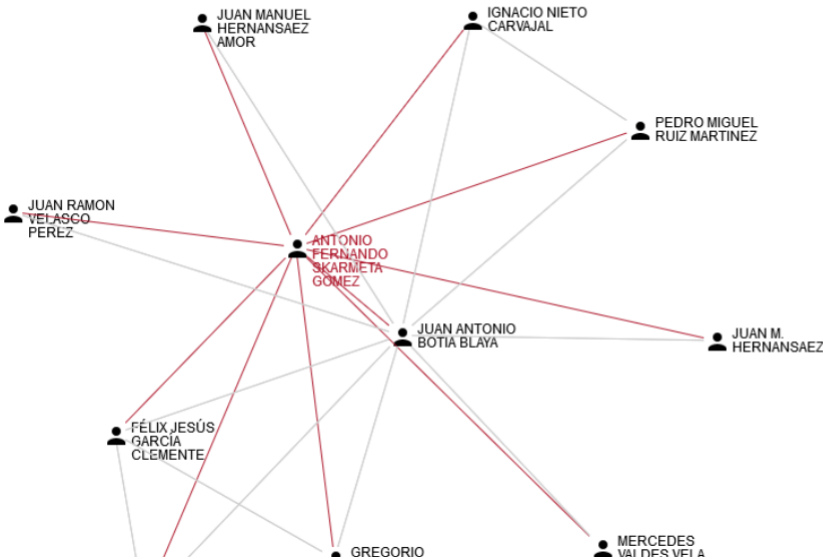
- ☐ Proporcion la presencia
- ☐ Proporcion
- ☐ Cumple l
- ☐ Cuenta c

http://graph.um.es/res/article/bae9443b-9d51-43c8-843b-ef56c664b79c	Towards semantic web-based management of security services	http://purl.org/ron/mirror/bibo#AcademicArticle
http://graph.um.es/res/article/6bcd4117-6a6a-4661-b165-5a7b9858aa50	Towards semantic-aware management of security services in GT4	http://purl.org/roh/mirror/bibo#AcademicArticle
http://graph.um.es/res/article/6e10fef0-ae84-43f9-bfc8-b500c16e6316	Toward a Framework for the Specification of Hybrid Fuzzy Modeling	http://purl.org/roh/mirror/bibo#AcademicArticle

Mostrando página 1 de 3

Anterior 1 2 3

Coautores

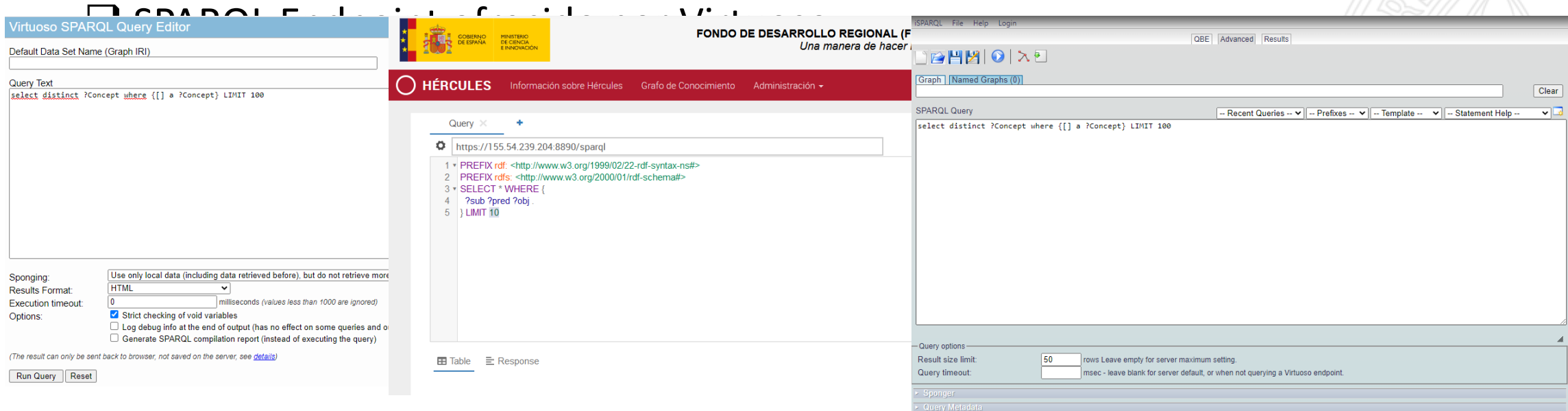


se cargados
vicio tiene
e incluyen
máquinas.
nombre.

Hércules. SPARQL Endpoint

ASIO cuenta con un SPARQL Endpoint de acceso público, que:

- ☐ Permite consultas de lectura sobre los datos públicos del SGI.
- ☐ Se pueden configurar limitaciones de uso para evitar saturaciones y ataques.
- ☐ Disponible en varias URLs (pendiente SSL):



The screenshot displays the Virtuooso SPARQL Query Editor interface. The main window shows a SPARQL query: `select distinct ?Concept where {[?Concept]} LIMIT 100`. The interface includes a sidebar with navigation links: "HÉRCULES", "Información sobre Hércules", "Grafo de Conocimiento", and "Administración". The bottom section shows query options, including "Result size limit" set to 50 and "Query timeout" set to 1000 milliseconds. The interface also features a "Run Query" button and a "Reset" button.

Hércules. SPARQL Endpoint

ASIO cuenta con un SPARQL Endpoint de acceso público, que es accesible:

- ☐ Por personas, en un interfaz web (como hemos visto en la diapo anterior)
- ☐ Por máquinas, mediante peticiones HTTP de tipo GET o POST. Por ejemplo, con una petición GET con un comando cURL:

```
curl
```

```
"http://server/sparql?query=SELECT%20*%20WHERE%20%7B%3Fs%20%3Fp%20%3Fo%7D%20LIMIT%208"
```

La consulta enviada es:

```
SELECT * WHERE { ?s ?p ?o } LIMIT 8
```


FONDO EUROPEO DE DESARROLLO REGIONAL (FEDER)

Una manera de hacer Europa

Hércules ASIO. SPARQL



SPARQL. SPARQL Protocol and RDF Query Language

Es un Protocolo de consulta y lenguaje de interrogación para grafos RDF, normalizado inicialmente por el RDF Data Access Working Group del World Wide Web Consortium (W3C) y por el SPARQL Working Group en su version 1.1

- ❑ SPARQL (<https://www.w3.org/TR/sparql11-overview/>) permite la consulta de grafos RDF a través de un lenguaje sencillo
- ❑ SPARQL es idóneo para extraer y consultar información mantenida por aplicaciones, servicios o repositorios ad-hoc de terceras partes expresados en RDF
- ❑ Elementos destacados:
 - ❑ **Lenguaje de consultas** para seleccionar o actualizar datos.
 - ❑ Mecanismo para **transmitir una consulta** a un servicio de procesamiento de consultas remoto
 - ❑ Formato **XML, CSV, TSV y JSON** para obtención de los resultados

SPARQL. Ejemplo de consulta SPARQL

Obtención de los triples de una investigadora concreta:

```
select * from <http://graph.um.es/graph/um cvn>  
where { <http://graph.um.es/res/person/f6544839-0e1e-4e4b-bc5d-70199198a50f> ?p ?o }
```

Todos los datos

Grafo origen de los triples

URI de la investigadora de la que se quieren recuperar los datos

SPARQL. Ejemplo de consulta SPARQL

Obtención de las entidades de las que un investigador concreto es objeto:

```
select ?s ?rdfType from <http://graph.um.es/graph/um_cvn>
where {
  ?s ?p <http://graph.um.es/res/person/f6544839-0e1e-4e4b-bc5d-70199198a50f>.
  ?s rdf:type ?rdfType.
}
```

Devuelve los URIs y tipos de las entidades de las que un investigador es objeto. Por ejemplo, listas de autores en las que está el investigador

Grafo origen de los triples

URI del investigador como objeto de otras entidades

Petición del tipo de la entidad de la que el investigador es objeto

FONDO EUROPEO DE DESARROLLO REGIONAL (FEDER)

Una manera de hacer Europa

Hércules ASIO. Casos



0. Obtención del Token de autorización de llamadas a los APIs

El token se puede obtener y proporcionar desde el interfaz de administración de ASIO:

https://herc-as-front-desata.atica.um.es/carga-web/Token/get?token_Type=0

En la ejecución de los casos vamos a usar éste (duración temporal):

Bearer

eyJhbGciOiJIUzI1NiIsImtpZCI6IkhRWk03VklzOWViLV8tWFRDOGFNMHciLCJ0eXAiOiJhdCtqd3QifQ.eyJ1YmYiOiJlY2MTI0MjQ2MzgsImV4cCI6MTYxMjUxMTAzOCwiaXNzIjoiaHR0cDovL2hlcmtYXMTnJvbnQtZGVzYS5hdGJlYS51bS5lczo1MTA4liwiYXVkljoiYXBpQ2FyZ2EiLCJjbGllbnRfaWQiOiJlYXZWIiLCJzY29wZSI6WyJhcGlDYXJnYSJdfQ.MhGkb6Y550Aalgz5BN88jRXkWQubeCPd_RzdbyBb0dY5u7xy5GpJrUHEg10I__CPothhiy mzmClvl0B3FrPIBIONzelySec9ywnsWcqE3mG3q0OmZrWtAgkc-nvDDxYDFPiOGI5PCku5hDr-hjjlv-AIECSdcSKC7muUHGkbA9kpD5y6p5BnAJ5R4HBtmnwV8Thm803Q_RxrQhdX4WZQQDh2otvUyuBexJnpJxjpziS9dGGO9xEJO4cjf3x5CVbkp SABayNwSL2FXe81bs2t5R12jLDIx-drKpb1cd7gAksXZELj1rzso24ESIfBc1C2ZBLphYRardU0fCPjPBPXQ

Accedemos al API en:

<http://herc-as-front-desata.atica.um.es/carga/swagger/index.html>

Documentación de la jornada:

<https://github.com/HerculesCRUE/GnossDeustoBackend/tree/master/Formacion/Taller%20Servicios%20Web>

1. Interacción con los repositorios OAI-PMH configurados

La especificación OAI-PMH se puede consultar en:

<https://openarchives.org/OAI/openarchivesprotocol.html>

Disponemos de un repositorio OAI-PMH con 8 CVs simulados de la Universidad de Murcia. Este repositorio tiene el identificador:

5efac0ad-ec4e-467d-bbf5-ce3f64edb46a

También hay configurado un repositorio que devuelve datos en formato XML simulando el comportamiento de actualización de datos de Hércules SGI. Este repositorio tiene el identificador:

14e66ce5-7bbf-44f4-8ad4-e4019f34edad

La manipulación de los repositorios se hace desde la sección 'repository' del API Carga

1.1. Repositorios disponibles

Para obtener los repositorios OAI-PMH configurados se llama a /etl-config/repository

Por ejemplo, con la siguiente instrucción:

```
curl -X GET "http://herc-as-front-desata.um.es/carga/etl-config/repository" -H "accept: text/plain" -H "Authorization: <autorización>"
```

El repositorio que nos interesa tiene el identificador:

5efac0ad-ec4e-467d-bbf5-ce3f64edb46a

Para interactuar con los repositorios configurados se puede usar el API Carga, que hará de proxy hacia los repositorios configurados.

1.2. MetadataFormat disponibles

Para obtener los 'MetadataFormat' configurados en el repositorio se llama a etl/ListMetadataFormats

Por ejemplo, con la siguiente instrucción:

```
curl -X GET "http://herc-as-front-desat.atica.um.es/carga/etl/ListMetadataFormats/5efac0ad-ec4e-467d-bbf5-ce3f64edb46a" -H "accept: */*" -H "Authorization: <autorización>"
```

Obtenemos que el 'MetadataFormat' disponible es 'rdf'

1.3. SetSpec disponibles

Para obtener los 'SetSpec' configurados en el repositorio se llama a etl/ListSets

Por ejemplo, la siguiente instrucción:

```
curl -X GET "http://herc-as-front-desata.um.es/carga/etl/ListSets/5efac0ad-ec4e-467d-bbf5-ce3f64edb46a" -H "accept: */*" -H "Authorization: <autorización>"
```

Obtenemos que el 'SetSpec' disponible es 'cvn'

1.4. Listar los identificadores de los *records* disponibles

Para listar los identificadores de los '*records*' se llama a etl/ListIdentifiers

Por ejemplo, la siguiente instrucción:

```
curl -X GET "http://herc-as-front-desata.um.es/carga/etl/ListIdentifiers/5efac0ad-ec4e-467d-bbf5-ce3f64edb46a?metadataPrefix=rdf" -H "accept: */*" -H "Authorization: <autorización>"
```

Obtenemos un listado con los identificadores disponibles junto con su 'SetSpec'

1.5. Obtener un *record*

Para obtener un *record* en particular se llama a etl/GetRecord

Por ejemplo, la siguiente instrucción:

```
curl -X GET "http://herc-as-front-desa.atika.um.es/carga/etl/GetRecord/5efac0ad-ec4e-467d-bbf5-ce3f64edb46a?identifier=1&metadataPrefix=rdf" -H "accept: */*" -H "Authorization: <autorización>"
```

Obtiene un *record* con todos sus metadatos.

2. Interacción con los validaciones SHACL configuradas

La especificación de SHACL se puede consultar en:

<https://www.w3.org/TR/shacl/>

Disponemos de varias validaciones SHACL para el repositorio:

5efac0ad-ec4e-467d-bbf5-ce3f64edb46a

La manipulación de los validaciones se hace desde la sección 'Validation' del API Carga

2.1. Validaciones disponibles

Para obtener las validaciones SHACL configurados se llama a etl-config/Validation

Por ejemplo, la siguiente instrucción:

```
curl -X GET "http://herc-as-front-desa.atika.um.es/carga/etl-config/Validation" -H "accept: text/plain" -H "Authorization: Bearer <autorización>"
```

Obtiene el listado con las validaciones SHACL configuradas.

Para ejecutar las validaciones SHACL se usa el API Carga.

2.2. Obtener una validación SHACL en particular

Para obtener una validación se llama a etl-config/Validation/{identifier}

Por ejemplo, la siguiente instrucción:

```
curl -X GET "http://herc-as-front-desa.atika.um.es/carga/etl-config/Validation/5a932857-940d-4ee9-93cf-5d2a57509c1a" -H "accept: text/plain" -H "Authorization: Bearer <autorización>"
```

Obtiene la validación con el identificador '5a932857-940d-4ee9-93cf-5d2a57509c1a', que será la validación que se va a incumplir en el siguiente ejemplo.

2.3. Aplicar validación sobre un RDF con las configuraciones del repositorio 5efac0ad-ec4e-467d-bbf5-ce3f64edb46a

Para ejecutar la validación de un RDF con las configuraciones de un repositorio se llama a etl/data-validate

Por ejemplo, con la instrucción:

```
curl -X POST "http://herc-as-front-desa.atika.um.es/carga/etl/data-validate?repositoryIdentifier=5efac0ad-ec4e-467d-bbf5-ce3f64edb46a" -H "accept: */*" -H "Authorization: <autorización>" -H "Content-Type: multipart/form-data" -F "rdfFile=@Ejemplo error validación.xml;type=text/xml"
```

Se puede probar con los ficheros 'Ejemplo error validación.xml' y 'Ejemplo SIN error validación.xml'

<https://github.com/HerculesCRUE/GnossDeustoBackend/tree/master/Formacion/Taller%20Servicios%20Web>

3. Ejecución de shapes arbitrarios sobre RDF

Se puede ejecutar un shape arbitrario sobre un RDF, llamando a etl/data-validate-personalized

Por ejemplo, con la instrucción:

```
curl -X POST "http://herc-as-front-desa.atika.um.es/carga/etl/data-validate-personalized" -H  
"accept: */*" -H "Authorization: <autorización>" -H "Content-Type: multipart/form-data" -F  
"rdfFile=@Ejemplo SIN error validación.xml;type=text/xml" -F  
"validationFile=@ShapePersonSample.xml;type=text/xml"
```

Se puede probar con los siguientes ficheros: 'Ejemplo error validación.xml' 'Ejemplo SIN error validación.xml' junto con el Shape 'ShapePersonSample.xml'

<https://github.com/HerculesCRUE/GnossDeustoBackend/tree/master/Formacion/Taller%20Servicios%20Web>

4. Ejecución de descubrimiento

El proceso de descubrimiento se encarga de:

Reconciliar las entidades con las entidades que ya están cargadas en el sistema

Descubrir enlaces hacia APIs externos (ORCID, DBLP, PUBMED...)

Detectar equivalencias en Unidata



4.1. Aplicar descubrimiento

Se puede ejecutar el proceso de descubrimiento sobre un RDF, llamando a etl/data-discover

Por ejemplo, con la instrucción:

```
curl -X POST "http://herc-as-front-desa.atika.um.es/carga/etl/data-discover" -H "accept: */*" -  
H "Authorization: <autorización>" -H "Content-Type: multipart/form-data" -F  
"rdfFile=@Ejemplo SIN error validación.xml;type=text/xml"
```

Este es un proceso ASÍNCRONO que puede resultar lento debido a que las peticiones que se hacen a los APIs externos. Dado que lo probaréis varias personas simultáneamente es mejor que NO utilicéis un RDF completo de los obtenidos en el OAI-PMH de CVN.

Este método devuelve un identificador para preguntar por el estado del descubrimiento.

Se puede probar con el siguiente fichero: 'Ejemplo SIN error validación.xml'

<https://github.com/HerculesCRUE/GnossDeustoBackend/tree/master/Formacion/Taller%20Servicios%20Web>

4.2. Obtener descubrimiento

Se puede obtener el estado del proceso de descubrimiento de un RDF, llamando a etl/data-discover-state

Por ejemplo, con la instrucción:

```
curl -X GET "http://herc-as-front-desa.atika.um.es/carga/etl/data-discover-state/8da903b6-73e6-4f5f-83bd-70c1a69f7031" -H "accept: text/plain" -H "Authorization: <autorización>"
```

Con este método obtenemos el resultado del proceso de descubrimiento:

id: identificador del item (el que se le pasa por parámetro)

status: estado del descubrimiento (Pending, Error, Processed)

rdf: rdf original

discoverRdf: rdf tras el descubrimiento

error: error, en el caso de que se haya producido.

discoverReport: informe del proceso de descubrimiento

DissambiguationProblems: problemas de desambiguación, en caso de que se hayan producido.

5. Obtención de datos del Linked Data Server y SPARQL

En el servicio Linked Data Server se pueden visualizar los datos cargados en la RDF Store y también solicitar los datos desde un programa para obtener el RDF de una entidad.



5.1. Obtención de datos de una publicación en Linked Data Server

Desde la siguiente instrucción se puede obtener el RDF de una publicación:

```
curl -X GET "http://graph.um.es/res/article/168a8caa-e151-4648-8f7c-e8f89ad69fb5" -H  
"accept: application/rdf+xml"
```



5.2. Obtención de datos de los publicadores de la publicación anterior en Linked Data Server

Usando una petición similar a la anterior, pero pidiendo los URIs de los autores de la publicación anterior, se pueden obtener sus datos RDF

5.3. De uno de los autores, obtener sus publicaciones en SPARQL Endpoint

Dirección del SPARQL Endpoint:

<http://155.54.239.204:8890/sparql>

Obtener primero todas sus publicaciones, luego sus artículos académicos y luego sus libros.
Probar con otros dos autores.

Se pueden encontrar ejemplos de las consultas en:

<https://github.com/HerculesCRUE/GnossDeustoBackend/tree/master/Formacion/Taller%20Servicios%20Web>

5.3. De uno de los autores, obtener sus publicaciones en SPARQL Endpoint

Las consultas anteriores se pueden hacer con un GET a la URL del Endpoint:

`http://155.54.239.204:8890/sparql?default-graph-uri=&query=<QUERY>&should-sponge=&format=<FORMATO>&timeout=0&debug=on&run=+Run+Query+`

Sustituyendo <QUERY> por la query correspondiente codificada en URL y <FORMATO> por el formato deseado, por ejemplo:

"text/html": HTML

"text/x-html+tr": HTML (Basic Browsing Links)

"application/vnd.ms-excel": Spreadsheet

"application/sparql-results+xml": XML

"application/sparql-results+json": JSON

"application/javascript": Javascript

"text/turtle": Turtle

"application/rdf+xml": RDF/XML

"text/plain": N-Triples

"text/csv": CSV

"text/tab-separated-values": TSV

5.4. De uno de los autores, obtener aquellos autores con los que es coautor en SPARQL Endpoint

Probar con tres de los autores anteriores ejecutando la consulta del mismo modo que en el ejemplo anterior.

Se pueden encontrar un ejemplo de las consulta en:

<https://github.com/HerculesCRUE/GnossDeustoBackend/tree/master/Formacion/Taller%20Servicios%20Web>

FONDO EUROPEO DE DESARROLLO REGIONAL (FEDER)

Una manera de hacer Europa

¡Gracias!



HERCULES

