

Estructura del repositorio ASIO en GitHub

Juan Valer y Javier Ruiz

RIAM Intelearning Lab – GNOSS

juanvaler@gnoss.com y javierruiz@gnoss.com



HĒRCULES

Introducción

- ☐ Organización del repositorio: Aplicaciones, Documentación y empaquetados
- ☐ Arquitectura de aplicaciones
- ☐ GitHub Actions, Codecov y SonarQube
- ☐ Taller de mantenimiento

Introducción GitHub

- Es una plataforma de desarrollo colaborativo usando el sistema de control de versiones Git.
- En un repositorio de GitHub se pueden albergar uno o varios proyectos



Requisitos funcionales

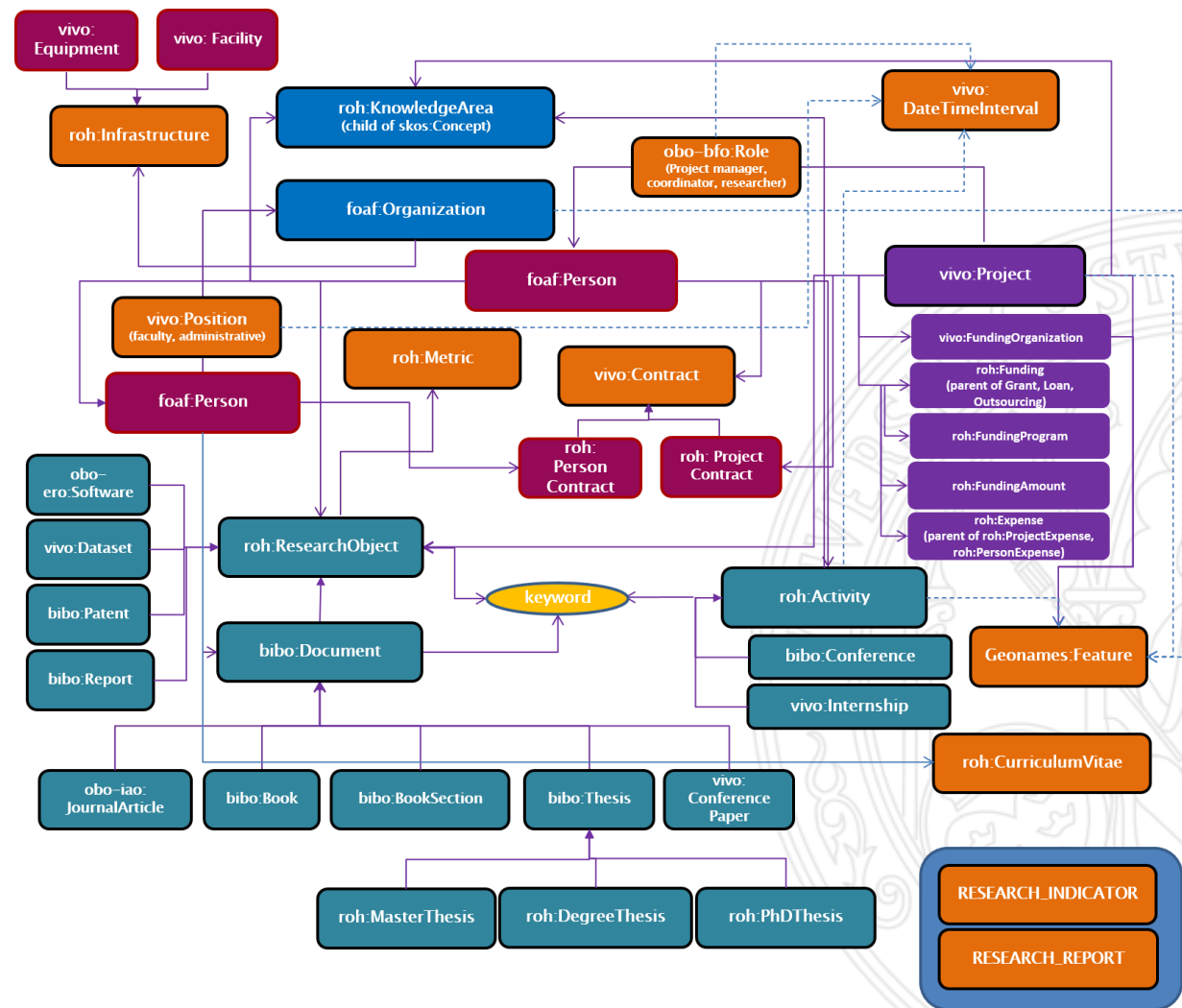
- Crear un grafo de conocimiento para los proyectos de investigación que se realizan en las universidades pertenecientes a la CRUE
- Identificar unívocamente a cada entidad, sin generar duplicidades
- Tener un sistema de descubrimiento y reconciliación de entidades
- Crear un sistema automatizado de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

Requisitos funcionales

- 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
- Por lo tanto, el sistema debe ser totalmente agnóstico del sistema RDF que se use.
- Crear una interfaz por la cuál se puedan añadir nuevas páginas de manera dinámica, consultando cualquiera de los API del ASIO
- Generar una carga inicial, con los datos de la Universidad de Murcia
- Crear un sistema de validación de entidades, que verifique que éstas siempre cumplen la ontología ROH.

Definiciones: Ontología

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



Definiciones: OWL

- Lenguaje de definición de ontologías para la Web (Ontology Web Language). Basado en XML

```
<owl:ObjectProperty rdf:about="http://purl.org/roh#hasKnowledgeArea">
  <owl:inverseOf rdf:resource="http://purl.org/roh#knowledgeAreaOf"/>
  <rdfs:range rdf:resource="http://www.w3.org/2004/02/skos/core#Concept"/>
  <rdfs:comment xml:lang="en">Some entity related to any Concept.</rdfs:comment>
  <rdfs:label xml:lang="en">has knowledge area</rdfs:label>
  <rdfs:label xml:lang="es">tiene area de conocimiento</rdfs:label>
</owl:ObjectProperty>
<!-- http://purl.org/roh#hasMetric -->
<owl:ObjectProperty rdf:about="http://purl.org/roh#hasMetric">
  <owl:inverseOf rdf:resource="http://purl.org/roh#metricOf"/>
  <rdfs:domain rdf:resource="http://purl.org/roh#ResearchObject"/>
  <rdfs:range rdf:resource="http://purl.org/roh#Metric"/>
  <rdfs:comment xml:lang="en">A Metric which quantifies a Research Object.</rdfs:comment>
  <rdfs:label xml:lang="en">has metric</rdfs:label>
  <rdfs:label xml:lang="es">tiene métrica</rdfs:label>
</owl:ObjectProperty>
```

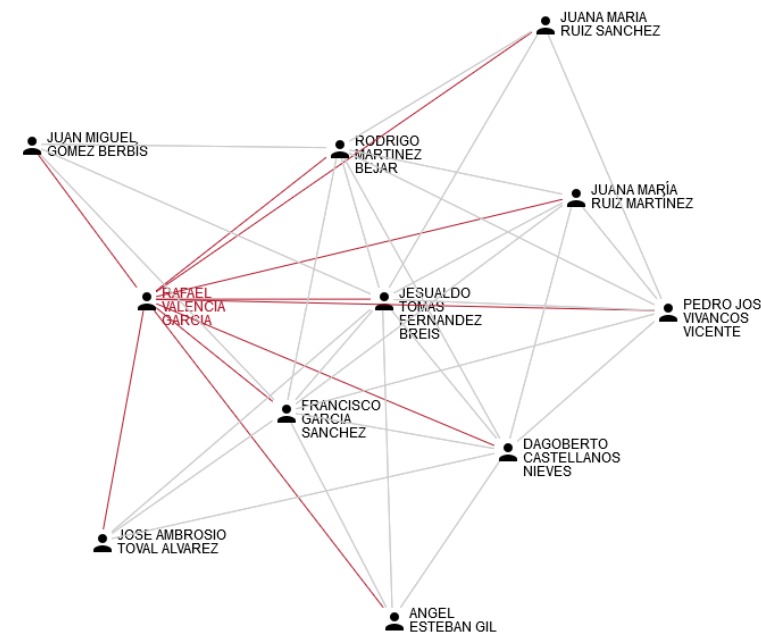

Definiciones: RDF

- Resource Description Framework. Lenguaje de definición de **instancias** de las ontología

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/">
<rdf:Description rdf:about="http://graph.um.es/res/researcher/0000-0001-8055-6823">
  <foaf:name>Diego López de Ipiña</foaf:name>
</rdf:Description>
</rdf:RDF>
```
























Definiciones: RDF Store

- Almacén RDF o base de datos de tipo Grafo.
- La información se almacena en formato triple: Sujeto – predicado – objeto



```
<http://graph.um.es/res/researchobject/10.3390/en13071700> dc:title "Socio-Economic Effect on ICT-Based Persuasive Interventions Towards Energy Efficiency in Tertiary Buildings"
<http://graph.um.es/res/researchobject/10.3390/en13071700> dc:author <http://graph.um.es/res/researcher/0000-0001-8055-6823> <http://graph.um.es/res/researcher/0000-0001-8055-6823>
```

Estructura del repositorio

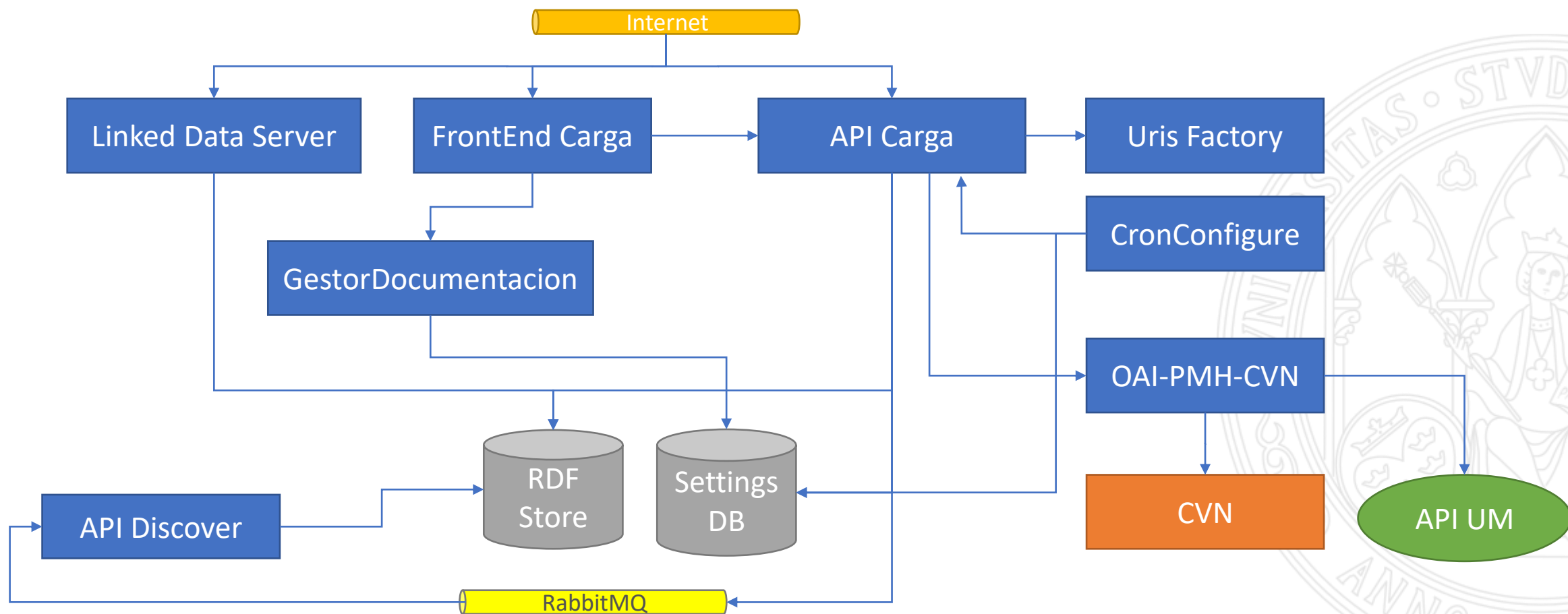
 .github/workflows	Create sonar_api_unidata.yml	18 days ago
 API_CARGA	Update README.md	18 days ago
 API_DISCOVER	Update README.md	3 days ago
 Benchmark	Revisión evaluación	8 months ago
 Build	update readme frontEndCarga	3 months ago
 CronConfigure	Update README.md	18 days ago
 Docs	Update 20200325 Hércules ASIO Ejemplos de consultas SPARQL.md	2 hours ago
 Examples/UrisFactoryLibraryExample	Merge branch 'master' of https://github.com/HerculesCRUE/GnossDeusto...	9 months ago
 FrontEndCarga	Update README.md	3 days ago
 GestorDocumentacion	Update Startup.cs	14 days ago
 IdentityServerHecules	Update readme.md	18 days ago
 Linked_Data_Server	Update README.md	6 hours ago
 OAI_PMH_CVN	Update README.md	18 days ago
 Unidata/Api_Unidata	send list triples	7 months ago
 UrisFactory	Update UrisConfig.json	5 days ago
 cvn	Transforming cache key to lowercase to allow that same academic degre...	5 months ago
 docker-images	Update README.md	2 months ago
 fair	Update EF2.1-7. MÉTRICAS FAIR I- SOFTWARE Y DOCUMENTACIÓN DE ...	8 months ago
 img	Diagrama de componentes	10 months ago
 libraries	Api Discover XML	2 months ago
 validation-questions	Fixing Q13 URL	6 months ago
 .gitignore	Update .gitignore	10 months ago

Estructura del repositorio

- Aplicaciones ASIO
- Aplicaciones externas adaptadas
- Documentacion
- Empaquetados



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 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 (incluidos los de otros datasets ASIO a través del nodo Unidata), 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.

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, este api 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 debe ser: <http://data.um.es/res/researcher/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.

En la ejecución del proyecto se ha optado por el desarrollo de un componente propio, en lugar de integrar desarrollos existentes de software abierto, como **Trellis** o **Trifid**, por varios motivos.

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 una API para convertir XML CVN a RDF ROH.

Aplicaciones ASIO: CVN

Servidor HTTP que ofrece una API para convertir XML CVN a tripletas ROH



Estructura del repositorio: Aplicaciones externas adaptadas

- Benchmark: Permite ejecutar el benchmark sobre RDF Stores y permite que el usuario modifique las características a evaluar y sus respectivos pesos
- Fair: Toma como base la aplicación Fair Sharing de Mark Wilkinson, y añade test de evaluación propios del proyecto Hércules. También se han realizado modificaciones en la interfaz de usuario.

Estructura del repositorio: Documentación

- Docs
- Examples
- Img
- Formación
- Validation-Questions



Estructura del repositorio: Empaquetados

- Build
- Docker Images
- Libraries



GitHub Actions, Codecov y SonarQube

GitHub Actions

Sirve para automatizar tareas en un repositorio Git, por ejemplo cada vez que se hace una subida en una rama.



Partes de una GitHub Action

- **Runner:** Es un servidor en el que está instalada la acción de GitHub, este runner ejecuta la acción y muestra el progreso
- **Event:** evento que desencadena la ejecución de la acción. Ej: hacer un commit en la rama master.
- **Jobs:** un conjunto de pasos que se ejecutan en el mismo runner.
- **Steps:** Es una tarea individual que puede ejecutar comandos. Entre estos pasos se puede compartir datos ya que se ejecutan en el mismo runner.

Partes de una GitHub Action

- **Event.**

```
name: Build and test API_CARGA
```

```
on:
```

```
  push:
```

```
    branches: [ master ]
```

```
  pull_request:
```

```
    branches: [ master ]
```

- **Jobs.**

```
jobs:
```

```
  build:
```

```
    runs-on: ubuntu-latest
```

```
    steps:
```

```
      - uses: actions/checkout@v2
```

```
      - name: Setup .NET Core
```

```
        uses: actions/setup-dotnet@v1
```

```
        with:
```

```
          dotnet-version: 3.1.301
```

```
      - name: Build
```

```
        run: dotnet build API_CARGA
```

```
      - name: Test
```

```
        run: dotnet test --collect:"XPlat Code Coverage" API_CARGA
```

```
      - name: Codecov
```

```
        # You may pin to the exact commit or the version.
```

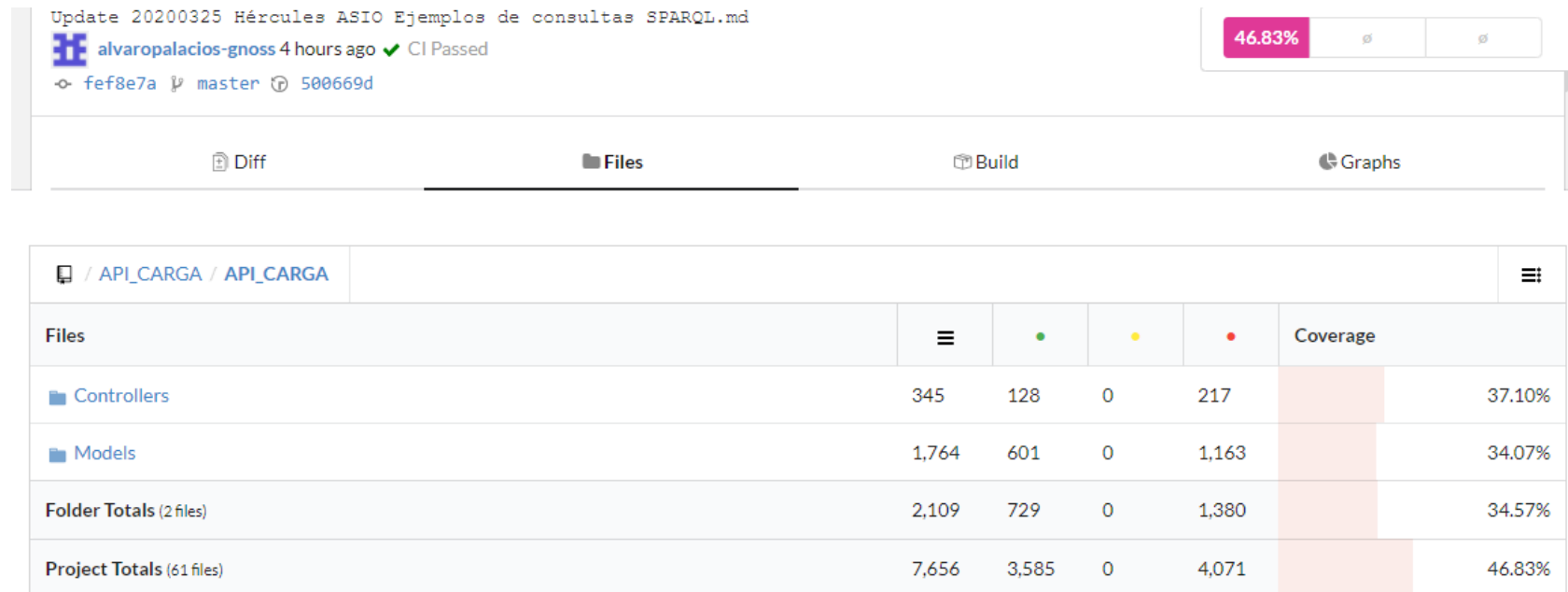
```
        # uses: codecov/codecov-action@239feb655bba88b16ff5dea1d3135ea8663a1f9
```

```
        uses: codecov/codecov-action@v1.0.15
```

- **Steps.**

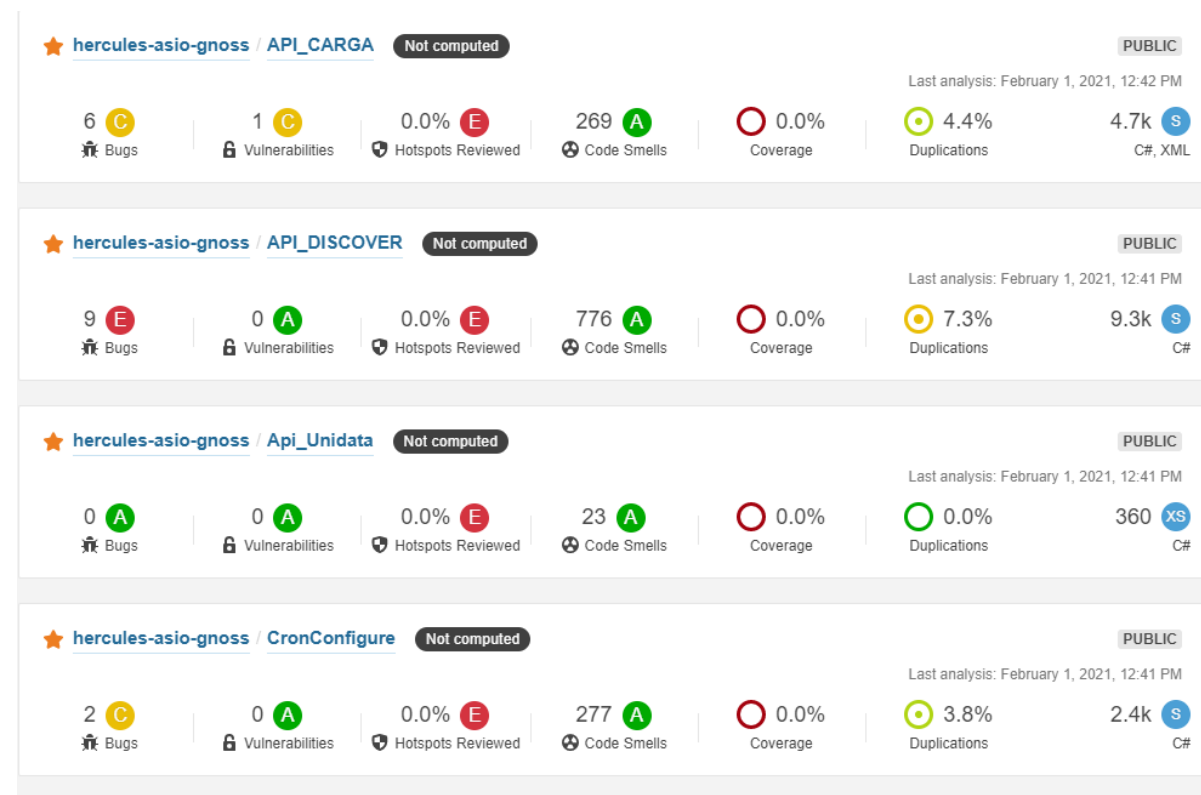
Actions en el proyecto ASIO - Build and test

- Con estas acciones lo que hacemos es compilar el proyecto configurado en la action y ejecutar los test realizados en el proyecto para posteriormente mandar los resultados a codecov.



Actions en el proyecto ASIO - Sonar

- Con estas acciones subimos a sonarcloud un análisis del código de los proyectos.
- Para ello hay que configurar un secret por cada proyecto, el secret nos lo da el propio sonar



FONDO EUROPEO DE DESARROLLO REGIONAL (FEDER)

Una manera de hacer Europa

¡Gracias!



HERCULES

