E4.1 SERVICIOS CONTENIDO

ESPACIO DE DATOS LINGÜÍSTICO (SER 15/23 OTT)

Resumen

Este documento describe la versión final de los principales servicios de contenido del conector INESData que se usan en el espacio de datos lingüístico EDL. Los servicios de contenido están relacionados con la localización, almacenamiento y transferencia de ficheros asociados a los activos en el espacio de datos, y se usan en la interfaz de usuario del espacio de datos. Se incluye servicios que dan soporte a recursos lingüísticos disponibles como recursos HTTP o en un sistema de almacenamiento en la nube que provee INESData. Además, se describe los servicios que permiten transferir recursos usando el protocolo HTTP o a un sistema de almacenamiento en la nube soportado por INESData. En esta versión del documento se incluyen los servicios que permiten descarga al entorno local del usuario los recursos lingüísticos asociados a un activo.

Andrés García-Silva

10/12/2024

Expert.ai
Language Technology Research Lab

Calle Poeta Joan Maragall, 3-5, Escalera Izquierda, Planta 1ª, Derecha, 28020, Madrid
CIF: B-66425513, Inscrita en el Registro Mercantil de Madrid, en el Tomo 44.538, Folio 74, Hoja Número M-784613, Inscripción 1ª.

www.expert.ai

Historia

revisión	Fecha	Descripción	Autor
0.1	20/06/2024	Primera versión del documento	Andrés García-Silva
0.2	09/12/2024	Segunda versión del documento	Andrés García-Silva

Contenido

1 Introducción

El espacio de datos lingüístico EDL se apoya en el conector y los componentes asociados de INESData para el despliegue del espacio de datos y ofrecer funcionalidades a los usuarios. Los componentes de INESDATA ofrecen servicios de contenido que dan soporte a la localización, almacenamiento y transferencia de conjuntos de datos. La interfaz de usuario del espacio de datos lingüístico usa los servicios de contenido cuando el usuario crea activos o transfiere activos en el marco de negociaciones finalizadas. La interfaz se apoya en el conector de INESDATA y en servicios de almacenamiento que provee INESDATA como el sistema de almacenamiento Min.io.

En este documento se presenta la última versión disponible de los principales servicios que ofrecen los componentes de INESDATA para la localización, almacenamiento y transferencia de ficheros. Los recursos lingüísticos asociados a un activo en INESData pueden estar disponibles en i) una dirección HTTP, ii) almacenados previamente en un sistema de almacenamiento MinIO en la nube de INESData, o iii) almacenarse al momento de la creación del activo en el sistema de almacenamiento MinIO. Los recursos lingüísticos pueden a su vez ser transferidos al usuario consumidor usando i) un servicio HTTP o ii) a un sistema de almacenamiento minio en la nube de INESData que esté asociada al usuario consumidor. Además, en esta versión del documento se incluye las APIs necesarias para descargar los recursos lingüísticos asociados a un activo en el entorno local del usuario.

2 Almacenamiento INESData para recursos lingüísticos: MinIO

El espacio de datos lingüístico ofrece la posibilidad a los usuarios de almacenar los recursos lingüísticos asociados a activos en la nube de INESData. Cada conector tiene una instancia de un sistema de almacenamiento MinIO que se ejecuta en la nube de INESData.

MinIO es una plataforma de almacenamiento de objetos que se puede implementar en infraestructuras de nube privadas, públicas o híbridas. Está diseñado para ofrecer almacenamiento compatible con Amazon S3, lo que significa que las aplicaciones que funcionan con S3 pueden integrarse con MinIO sin cambios significativos. Algunas características importantes de MinIO son:

- Compatibilidad con S3: MinIO implementa el API de Amazon S3, lo que permite una fácil integración con aplicaciones y herramientas que ya utilizan S3.
- Rendimiento Elevado: MinIO está optimizado para un alto rendimiento, con capacidades para manejar grandes volúmenes de datos y realizar operaciones rápidas de lectura/escritura.
- Escalabilidad Horizontal: MinIO permite la escalabilidad horizontal, lo que significa que se pueden añadir más nodos al clúster para aumentar la capacidad de almacenamiento y mejorar el rendimiento.
- Almacenamiento Distribuido: MinIO proporciona almacenamiento distribuido, permitiendo la replicación para asegurar la disponibilidad y durabilidad de los datos.
- Seguridad: Ofrece características avanzadas de seguridad, incluyendo encriptación en tránsito y en reposo, así como control de acceso basado en políticas (IAM) detallado.
- Simplicidad y Facilidad de Uso: MinIO es conocido por su simplicidad y facilidad de uso, con una interfaz de usuario intuitiva y una configuración sencilla.

Los datos de configuración del MinIO del conector se encuentran en el fichero Docker-compose.yml que se usa para desplegar el espacio de datos. En el siguiente ejemplo se definen dos instancias minio para el conector 1 y 2 que podrían tener un rol de proveedor y consumidor de recursos lingüísticos respectivamente. Las consolas de administración se pueden acceder en el puerto 9001 y 9011 y las API en los puertos 9010 y 9000 respectivamente para cada conector cuando se accede desde fuera de los contenedores. Si el acceso es interno entre contenedores los puertos que dan acceso a la API es el 9000 en ambos casos y se debe usar la ruta interna de los contenedores: minio-c1 y minio-c2.

```
  minio-c1:
    image: minio/minio
    ports:
      - '9000:9000'
      - '9001:9001'
    volumes:
      - mdata-c1:/data
    environment:
      - MINIO_ROOT_USER=inesdata
      - MINIO_ROOT_PASSWORD=inesdata
      - MINIO_DEFAULT_BUCKETS=bucket-connector-1
    command: server --console-address ":9001" /data
    healthcheck:
      test: timeout 5s bash -c ':> /dev/tcp/127.0.0.1/9000' || exit 1
      interval: 30s
      timeout: 20s
      retries: 3

  minio-c2:
    image: minio/minio
    ports:
      - '9010:9000'
      - '9011:9011'
    volumes:
      - mdata-c2:/data
    environment:
      - MINIO_ROOT_USER=inesdata
      - MINIO_ROOT_PASSWORD=inesdata
      - MINIO_DEFAULT_BUCKETS=bucket-connector-2
    command: server --console-address ":9011" /data
    healthcheck:
      test: timeout 5s bash -c ':> /dev/tcp/127.0.0.1/9000' || exit 1
      interval: 30s
      timeout: 20s
      retries: 3
```

3 Recursos lingüísticos disponibles en dirección HTTP

La forma más sencilla de compartir recursos en el espacio de datos lingüísticos es tener disponibles los datos en una dirección URL desde donde se puedan acceder. Al crear un activo en el espacio de datos usando el servicio web /management/v3/assets con una llamada POST, el usuario proveedor tiene que especificar en la dirección de los datos (dataAddress) que el activo está disponible como datos de tipo HTTP (HttpData) y la URL donde se encuentran. A continuación, un ejemplo del documento JSON que se envía con esta información al crear el activo:

```
{
  "@context": {
```

```
    "@vocab": "https://w3id.org/edc/v0.0.1/ns/"
  },
  "@id": "corpus-ciencia-abstracts-01",
  "properties": {
    "name": " abstracts-de-ciencia",
    "contenttype": "application/json"
  },
  "dataAddress": {
    "type": "HttpData",
    "name": " abstracts-de-ciencia",
    "baseUrl": "https://jsonplaceholder.typicode.com/posts",
    "proxyPath": "true"
  }
}
```

Además, el proveedor del activo debe crear al menos una política de acceso y de contratación, y una definición de contrato donde se apliquen las políticas al activo que se desea compartir vía HTTP.

Por otra parte, el usuario consumidor debe iniciar la negociación del contrato que cubre el activo en el que está interesado y que aparece en el catálogo federado al que el consumidor tiene acceso. Está negociación se inicia con una llamada POST al servicio /management/v3/contractnegotiations y se debe enviar un fichero json con la información del activo que se quiere negociar, y la política de contratación:

```
POST /management/v3/contractnegotiations HTTP/1.1
Host: localhost:29193
Content-Type: application/json
Authorization: Bearer Access Token

{
  "@context": {
    "@vocab": "https://w3id.org/edc/v0.0.1/ns/",
    "odrl": "http://www.w3.org/ns/odrl.jsonld"
  },
  "@type": "ContractRequest",
  "counterPartyAddress": "http://connector-c1:19194/protocol",
  "protocol": "dataspace-protocol-http",
  "policy": {
    "@context": "http://www.w3.org/ns/odrl.jsonld",
    "@id": "PolicyId-from-federated-catalog",
    "@type": "Offer",
    "odrl:permission": {
        "odrl:action": {
            "odrl:type": "USE"
        },
        "odrl:constraint": {
            "odrl:leftOperand": "POLICY_EVALUATION_TIME",
```

```
            "odrl:operator": {
                "@id": "odrl:lt"
            },
            "odrl:rightOperand": "2024-12-31T23:59:59+01:00"
        }
    },
    "assigner": "connector-c1",
    "target": "{{asset-id}}"
  }
}
```

Sí la política de contratación se valida automáticamente entonces la contratación debería finalizar de forma exitosa. Esto se puede comprobar usando el servicio de consulta de estado de negociación: management/v3/contractnegotiations/{id-contract}. La negociación debe estar en estado finalizada.

3.1 Transferencia de ficheros usando HTTP

Cuando la negociación ha finalizado con éxito se puede realizar la transferencia del activo. Para esto hay que hacer una llamada POST al servicio en /management/v3/transferprocesses enviando un documento json donde se indique el activo que se quiere transferir y el id del acuerdo del contrato que se generó en la negociación.

La transferencia es un proceso que prepara el recurso lingüístico que se ha negociado para que sea descargado. La transferencia no realiza la descarga del recurso.

```
POST /management/v3/transferprocesses HTTP/1.1
Host: localhost:29193
Content-Type: application/json
Authorization: Bearer Access token
{
  "@context": {
    "@vocab": "https://w3id.org/edc/v0.0.1/ns/"
  },
  "@type": "TransferRequestDto",
  "connectorId": "connector-c2",
  "counterPartyAddress": "http://connector-c1:19194/protocol",
  "contractId": "{{contract-agreement-id}}",
  "assetId": "{{asset-id}}",
  "protocol": "dataspace-protocol-http",
  "transferType": "HttpData-PULL"
}
```

El estado de la transferencia se puede ver haciendo una llamada GET al siguiente servicio pasando el id de la transferencia:

```
GET /management/v3/transferprocesses/{{transferId}} HTTP/1.1
```

```
Host: localhost:29193
Authorization: Bearer Access Token
```

3.1.1 Descarga de recursos

Una vez la transferencia ha finalizado exitosamente el recurso lingüístico asociado al activo negociado está listo para descargar. Lo primero que hay que hacer es pedir el punto de acceso y el token de acceso correspondiente al recurso. Para esto hay que hacer una llamada GET al servicio /management/v1/edrs/{transferid}/dataaddress:

```
GET /management/v1/edrs/{transferid}/dataaddress HTTP/1.1
Host: localhost:29193
```

Este servicio retorna el token de autorización (authorization) para usar el punto de acceso (endpoint) donde se puede descargar el recurso usando HTTP:

```
{
    "@type": "DataAddress",
    "type": "https://w3id.org/idsa/v4.1/HTTP",
    "endpoint": "http://connector-c1:19291/public",
    "authType": "bearer",
    "endpointType": "https://w3id.org/idsa/v4.1/HTTP",
    "authorization":
"eyJraWQiOiJwdWJsaWMta2V5LWMxIiwiYWxnIjoiUlMyNTYifQ.eyJpc3MiOiJjb25uZW
N0b3ItYzEiLCJhdWQiOiJjb25uZWN0b3ItYzIiLCJzdWIiOiJjb25uZWN0b3ItYzEiLCJp
YXQiOjE3MzM4Mjk2NTc2NzksImp0aSI6ImQ2MmU3OTBkLTE1MjgtNDM3NS05YjE1LTgwNT
JhMDBiZWQyNyJ9.LyhicY7zhuVMniJy5mZvXNoyTYxsNeKOltlx16k2kAM1vr4dblzIzwm
NPpMfew_SH_RnzHZexvfB77pQwFg7vzvwRvVYEbshF3-
qZl8VGEjvVkBnoyosL_TJBEs6TGhJHV011m0Vvt-
iVxpBG3C7lQbhAlj674LIyXIKwCZdfw_Wc512Lx7XvYicX-
s4QXiNxQDnAX1d7aiITOA6t8wTqosOwFPi-F-zYvpq-LBz7xK8QRtIT8g6vejU6d-
Ezbk6U4QKqMVyn_9F3wqa05o4WiLMSJ8oC2SvlZnaCILJlkLdgccOWtNhPrwPCWVQZjn8F
LhRJ82nFAQRD_fX_H9jiw",
    "@context": {
        "@vocab": "https://w3id.org/edc/v0.0.1/ns/",
        "edc": "https://w3id.org/edc/v0.0.1/ns/",
        "odrl": "http://www.w3.org/ns/odrl/2/"
    }
}
```

El siguiente paso consiste en acceder al punto acceso usando GET:

```
GET /public/ HTTP/1.1
Host: localhost:19291
Authorization:
```

Esta llamada descarga el recurso lingüístico al entorno local del usuario.

3.2 Transferencia de ficheros a Minio

Para iniciar la transferencia de ficheros al sistema de almacenamiento MinIO del conector consumidor se debe especificar en el campo transferType AmazonS3. Además, en dataDestination se debe incluir los datos de acceso cómo el id de la llave de acceso y la llave secreta de acceso, e información de la localización incluyendo el servicio de MinIO y el nombre del bucket donde se almacenará.

```
POST /management/v2/transferprocesses HTTP/1.1
Host: localhost:29193
Content-Type: application/json
Authorization: Bearer

{
  "@context": {
    "@vocab": "https://w3id.org/edc/v0.0.1/ns/"
  },
  "@type": "TransferRequestDto",
  "connectorId": "connector-c2",
  "counterPartyAddress": "http://connector-c1:19194/protocol",
  "contractId": "{{contract-agreement-id}}",
  "assetId": "{http-asset-id}",
  "protocol": "dataspace-protocol-http",
  "transferType": "AmazonS3",
  "dataDestination": {
    "properties": {
      "type": "AmazonS3",
      "region": "us-east-1",
      "bucketName": "bucket-connector-2",
      "accessKeyId": "wE..",
      "secretAccessKey": "Jl.."
    },
    "endpointOverride": "http://minio-c2:9000",
    "type": "AmazonS3"
  },
  "managedResources": true
}
```

Cuando el estado de la transferencia pase a Finalizada se puede acceder al MinIO del conector consumidor (por ejemplo, http://localhost:9011/) y descargar el recurso lingüístico.

4 Recursos lingüísticos almacenados en MinIO

El espacio de datos lingüístico soporta la creación de activos y su almacenamiento en MinIO. Como se ha visto al principio del documento la nube de INESData ofrece un repositorio MinIO para cada conector del espacio de datos. Sin embargo, también es posible que el repositorio MinIO este alojado en otra infraestructura. Se soporta la descarga de recursos lingüísticos

alojados en MinIO usando HTTP o su transferencia a un sistema de almacenamiento MinIO. Además, el activo puede referenciar un recurso lingüístico a almacenado en MinIO o puede subirlo al momento de la creación del activo al MinIO de conector en INESData.

4.1 Creación de activo con recurso lingüístico existente en MinIO

Para crear un activo que referencia un recurso lingüístico almacenado previamente en MinIO se requiere una llamada POST al servicio /management/v3/assets donde se envía un documento JSON en el cuerpo del mensaje con la información del activo. En la información del activo se especifica la localización del recurso en elemento dataAddress. A continuación, un ejemplo de llamada al servicio desde el conector del usuario proveedor para crear el activo:

```
POST /management/v3/assets HTTP/1.1
Host: localhost:19193
Content-Type: application/json
Authorization: Bearer Access token

{
  "@context": {
    "@vocab": "https://w3id.org/edc/v0.0.1/ns/"
  },
  "@id": "{{asset-id}}",
  "properties": {
    "name": "{{asset-name}}",
    "contenttype": "text/plain"
  },
  "dataAddress": {
    "type": "AmazonS3",
    "region": "us-east-1",
    "keyName": "{{file-name}}",
    "bucketName": "{{bucket-name}}"
  }
}
```

Como cualquier otro activo el usuario proveedor debe crear y asociar una política de acceso y de contrato en un contrato que cubra el activo.

4.2 Creación de activo y carga de recurso simultánea a MinIO

Para crear activos en el espacio de datos lingüísticos y almacenar el recurso asociado en la nube de INESData se hace una llamada POST al servicio /management/s3assets. Se pasa un documento JSON en el cuerpo del mensaje con la información del activo incluyendo los metadatos y la información de la ubicación (dataAddress). La información de la ubicación debe incluir el tipo INESDataStore y la carpeta donde se quiere almacenar el activo en MinIO. Además, se debe pasar el fichero que se va a subir a MinIO. Tanto el JSON como el fichero se den pasar como datos de un formulario de múltiples partes (multipart/form-data).

A continuación, una llama de ejemplo para crear un activo y subir el fichero relacionado a la nube de INESData:

```
POST /management/s3assets HTTP/1.1
Host: localhost:19193
```

```
Authorization: Bearer Access Token
Content-Type: multipart/form-data; boundary=----
WebKitFormBoundary7MA4YWxkTrZu0gW

------WebKitFormBoundary7MA4YWxkTrZu0gW
Content-Disposition: form-data; name="json"
Content-Type: application/json

{
    "@context": {
        "@vocab": "https://w3id.org/edc/v0.0.1/ns/"
    },
    "@id": "{{Asset-id}}",
    "properties": {
        "name": "{{Asset-name}}",
        "contenttype": "text/plain"
    },
    "dataAddress": {
        "type": "INESDataStore",
           "folder": "{{folder-name}}"
    }
}
------WebKitFormBoundary7MA4YWxkTrZu0gW
Content-Disposition: form-data; name="file";
filename="/C:/Users/agarcia/Downloads/fibra.pdf"
Content-Type: application/pdf

(data)
------WebKitFormBoundary7MA4YWxkTrZu0gW—
```

Si la creación del activo ha sido correcta el servicio devuelve el código 200 y un json confirmando la creación:

```
{
    "@type": "IdResponse",
    "@id": "c1-internalS3-asset-1",
    "createdAt": 1718968110719,
    "@context": {
        "@vocab": "https://w3id.org/edc/v0.0.1/ns/",
        "edc": "https://w3id.org/edc/v0.0.1/ns/",
        "odrl": "http://www.w3.org/ns/odrl/2/"
    }
}
```

Para verificar que el fichero se ha subido a la nube de INESData se puede visitar la web de administración de MinIO asociada al conector proveedor. Además, el usuario proveedor debe crear un contrato donde se asignen políticas de acceso y de contratación al activo.

4.3 Transferencia de recursos lingüísticos y descarga usando HTTP

La transferencia de ficheros almacenados en MinIO usando el protocolo HTTP es exactamente igual a la transferencia que se describió en la sección 3.1. Además, la descarga del recurso lingüístico al entorno local del usuario es igual al proceso presando en la sección 3.1.1.

4.4 Transferencia de recursos lingüísticos a Minio S3

La transferencia de ficheros almacenados en minio a un sistema de almacenamiento MinIO del conector consumidor es igual al proceso que se describió en la sección 3.2.

5 Conclusiones

Se ha presentado la versión actualizada de los servicios de contenidos que usa el espacio de datos lingüísticos y que ofrece el conector de INESData en su versión 0.8.0. Estos servicios permiten localizar, almacenar y transferir asociados a los activos creados en los espacios de datos. Los recursos lingüísticos pueden estar disponibles como un recurso HTTP o en un sistema de almacenamiento MinIO externo o el que ofrece la nube de INESData. El espacio de datos soporta la creación de activos y la carga de recursos al sistema almacenamiento MinIO propio del usuario en INESData. Estos recursos lingüísticos pueden transferirse vía HTTP y descargarse en el entorno local del usuario o directamente a un sistema de destino MinIO asociado al conector del usuario consumidor del activo. En esta versión del documento se incluye los servicios que habilitan a los usuarios del espacio de datos a descargar los recursos lingüísticos a su entorno local una vez han sido negociados y transferidos correctamente en el espacio de datos.