



Metodologías y Tecnologías de la Integración de Sistemas



Iván Mora Maciá
Joaquín Vasalo Vicedo
Arancha Ferrero Ortiz de Zárate
Jaume Moreno Cantó
Álvaro Muñoz Delgado
Jaime Sarrión Sahuquillo

ÍNDICE

- Descripción del proyecto
- SOAP
- REST
- Tecnologías usadas
- Pasos para la integración
- DEMO

1. Descripción del proyecto

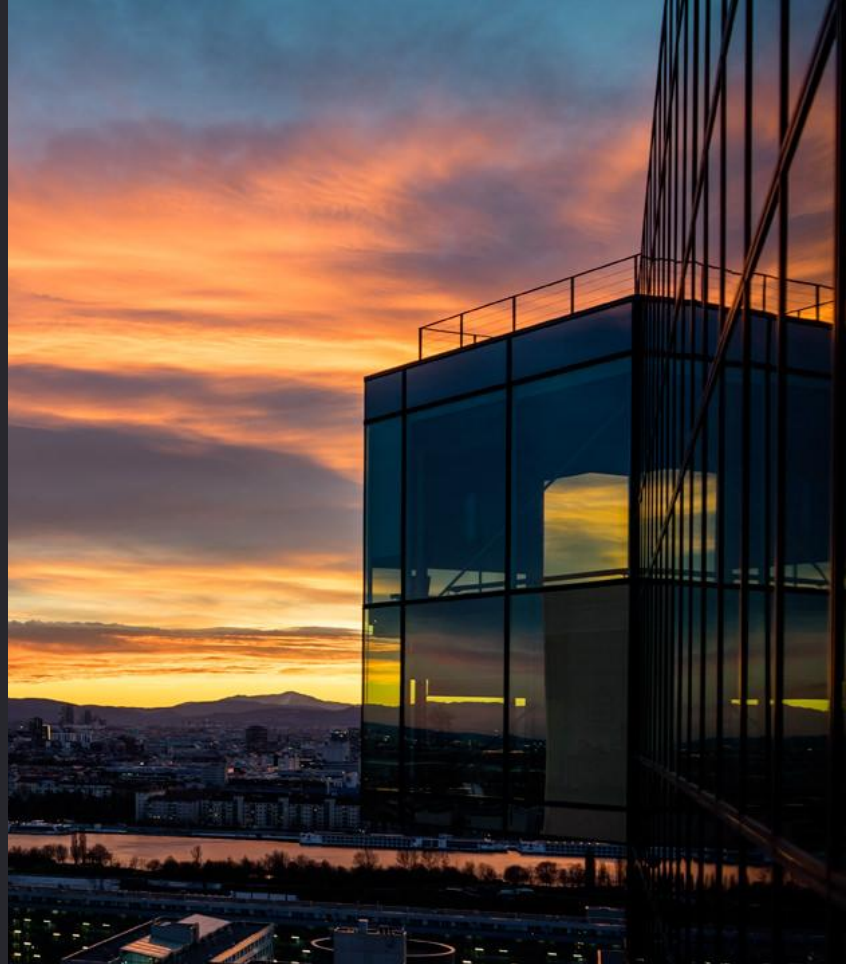


Tesla

Compañía estadounidense ubicada en Silicon Valley

¿Qué hace?

- Diseña, vende y fabrica:
 - Coches eléctricos
 - Componentes para la propulsión de Vehículos eléctricos
 - Sistemas de almacenamiento de baterías
 - Paneles solares



Procesos a modelar

SOAP

QA Aplicaciones Software

Subida de archivos de trabajo

Programación de paneles energéticos movibles

REST

Contratar personal

Manufactura de paneles solares

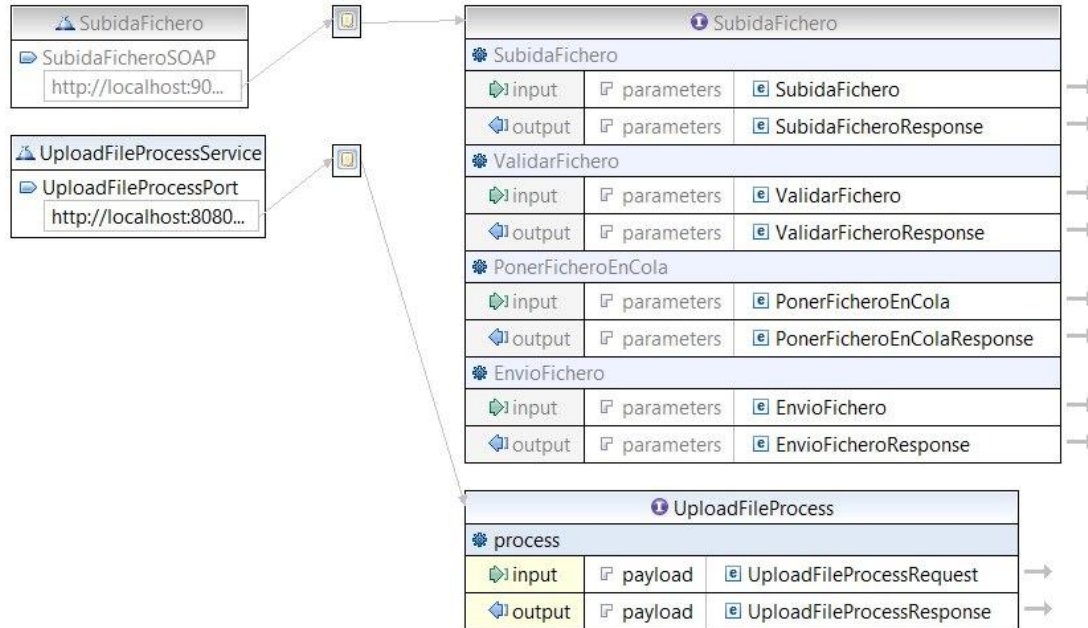
E-commerce de baterías y placas

2.

Análisis SOA: SOAP

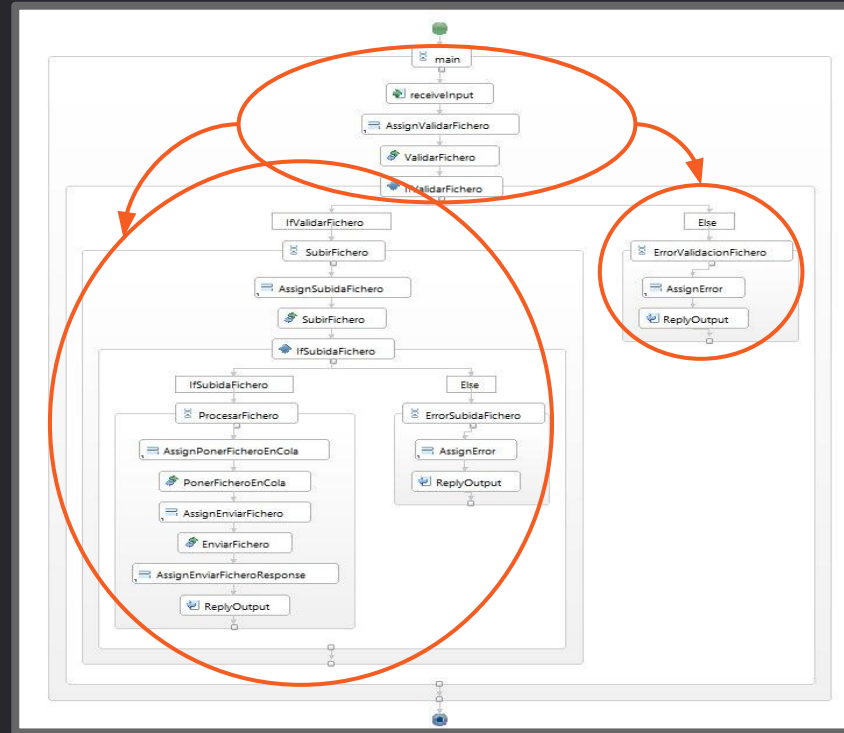


Subida de archivos de trabajo (WSDL)



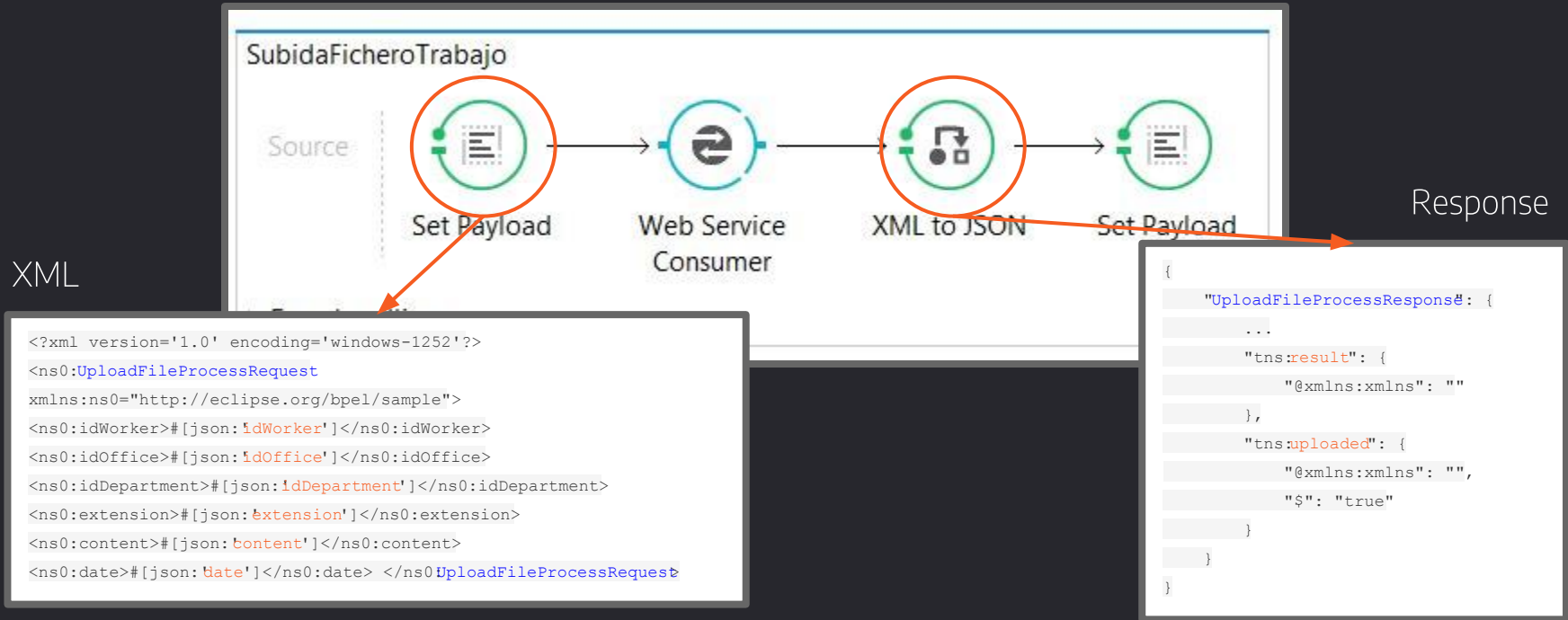
Subida de archivos de trabajo (BPEL)

- Entrada de fichero
- Validación de fichero
- Subida Fichero
- Procesado Fichero
- Asignación Fichero

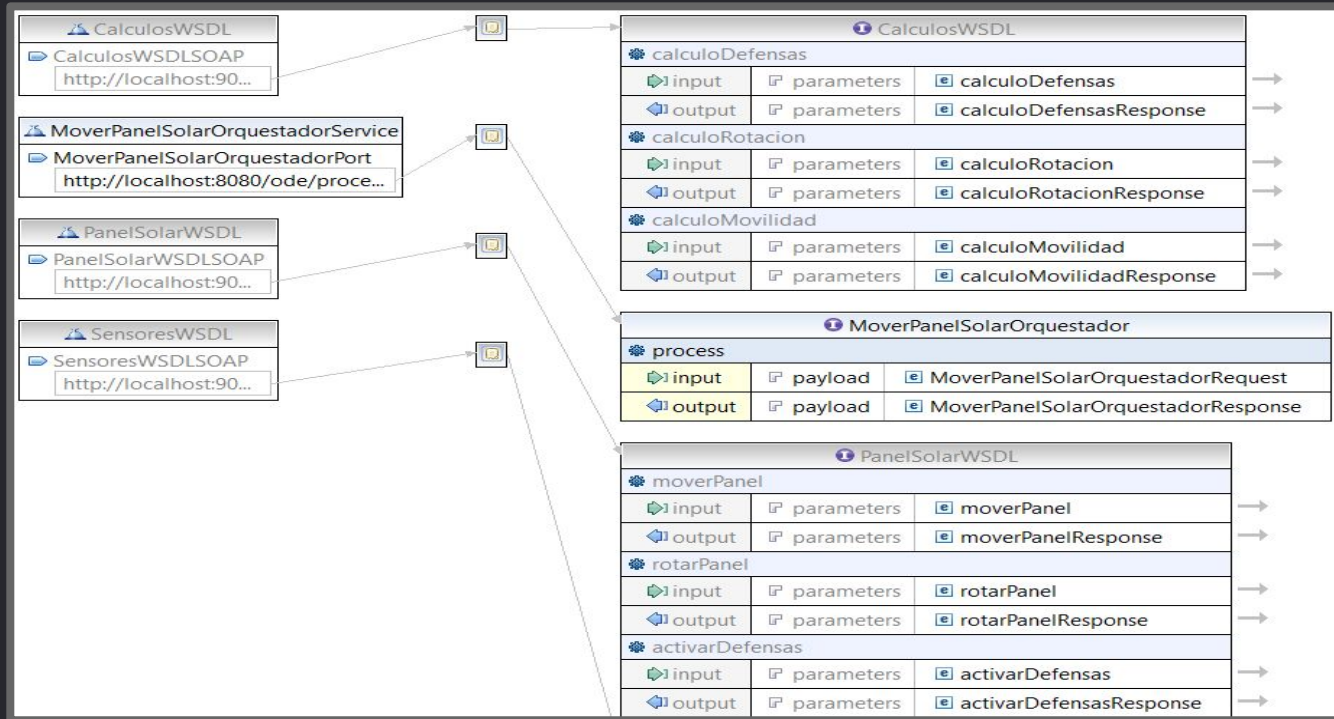


- Error de validación
- Error de subida fichero

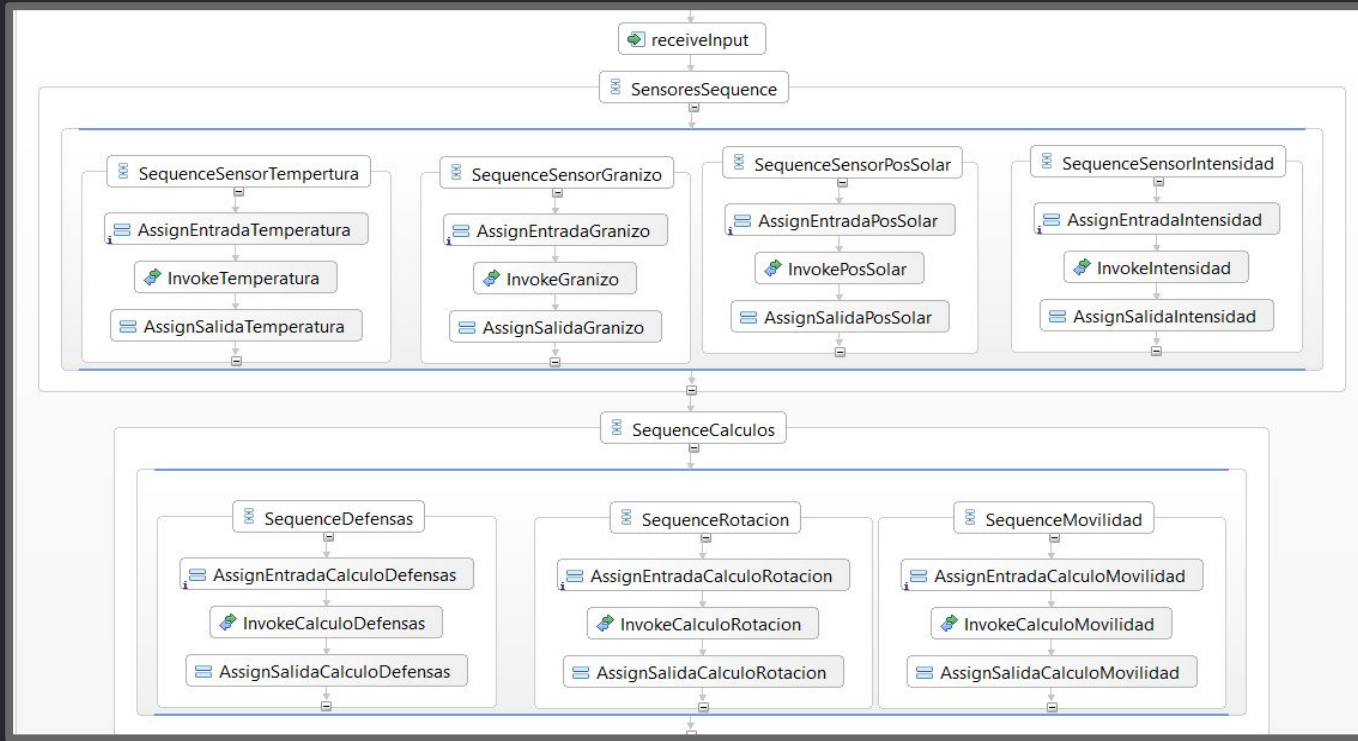
Subida de archivos de trabajo (Mule)



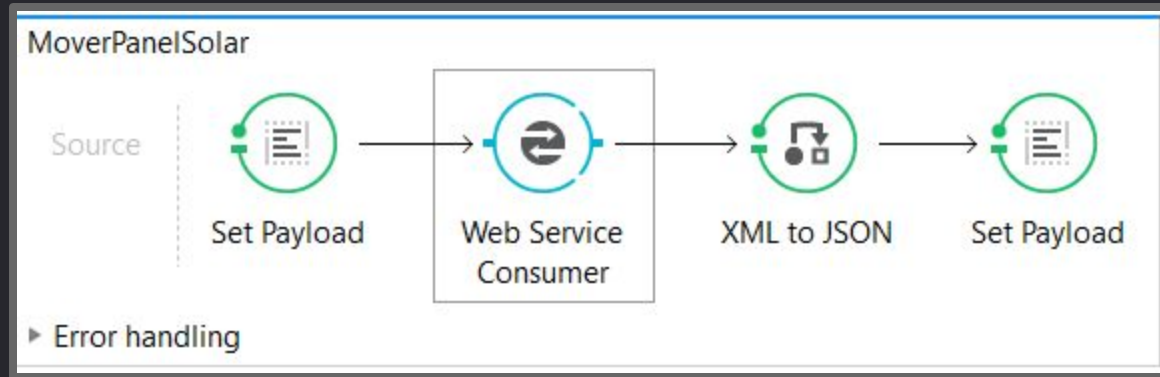
Programación de paneles energéticos movibles(WSDL)



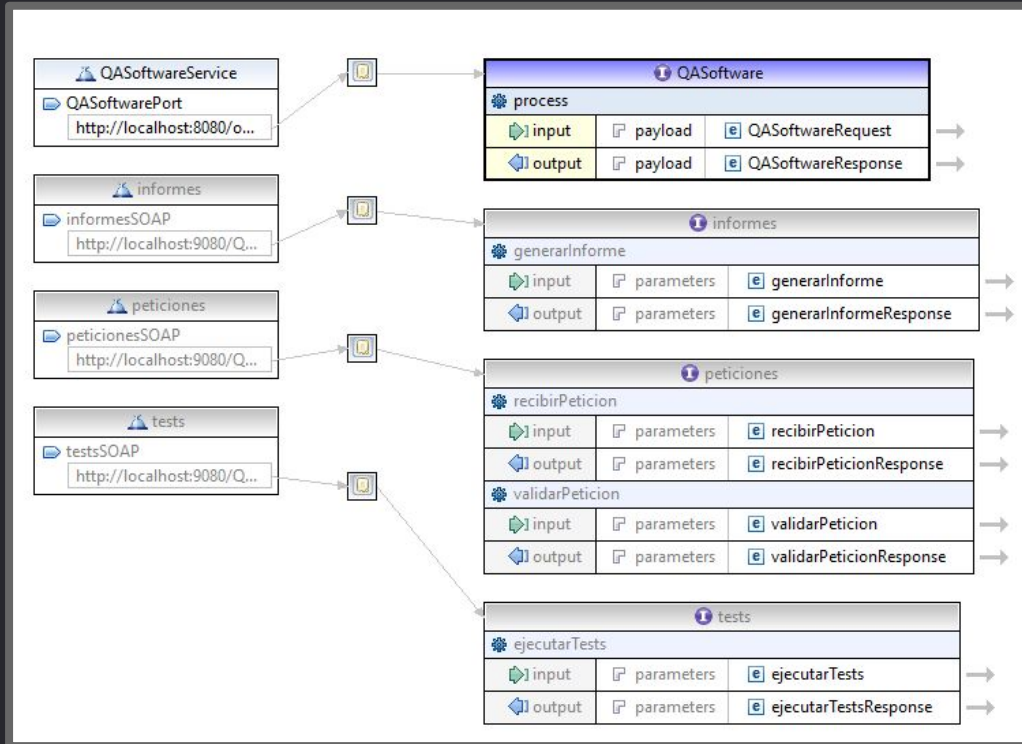
Programación de paneles energéticos móviles(BPEL)



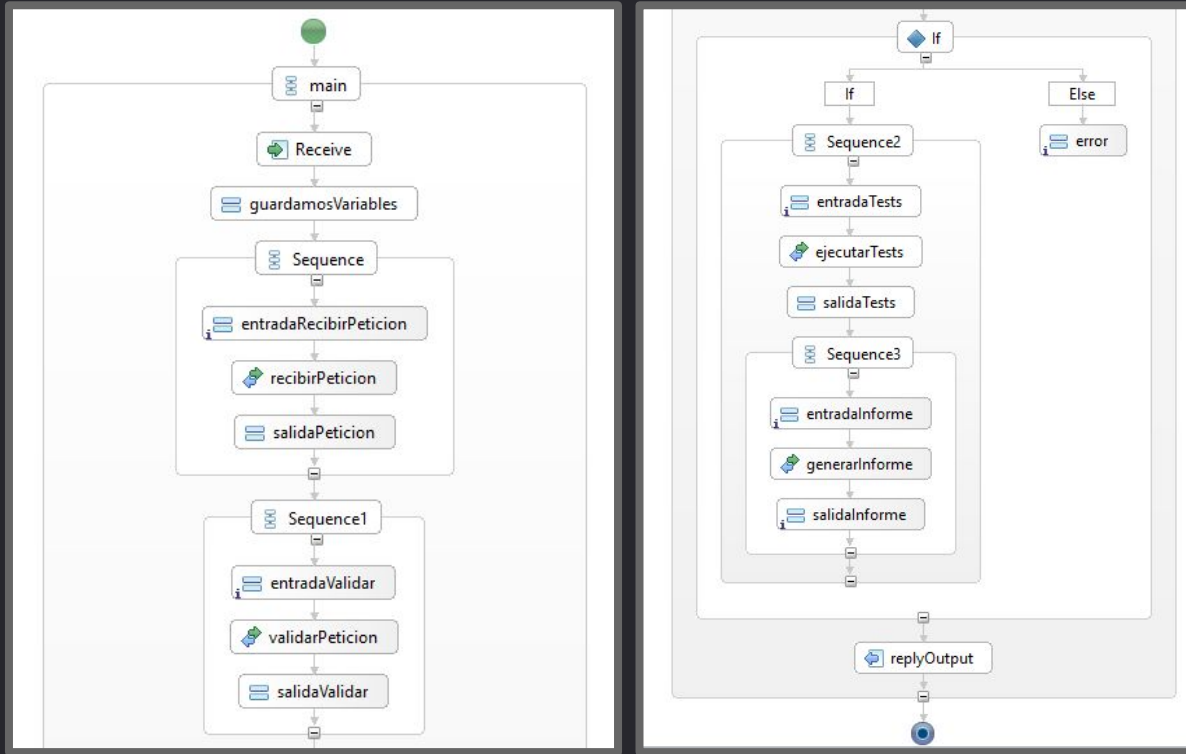
Programación de paneles energéticos móviles(Mule)



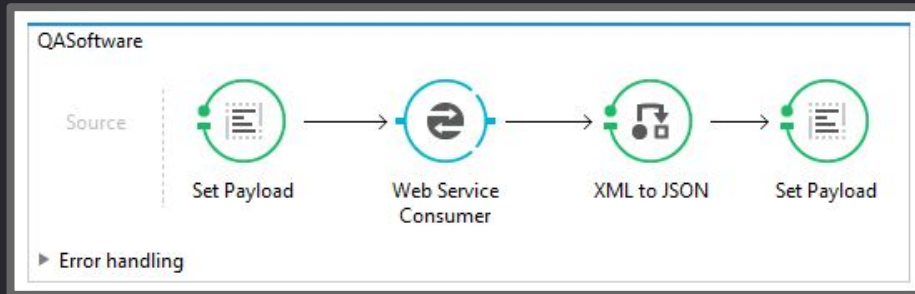
QA Aplicaciones Software(WSDL)



QA Aplicaciones Software(BPEL)



QA Aplicaciones Software(Mule)



3. Análisis REST:



Contratar personal (RAML)

```
#RAML 1.0
title: Contratar personal
mediaType: application/json
baseUri: localhost:3001/contratar
version: v1.0
```

```
types:
  OfertaEmpleado:
    type: object
    properties:
      id: number
      category: string
      title: string
      description: string
      minimumRequirements: object
  PerfilSolicitante:
    type: object
    properties:
      id: number
      name: string
      surnames: string
      birthdate: string
      location: string
      abilities: object
      appliedJob: integer
      via: string
      timestamp: string
  PerfilCandidato:
    type: object
    properties:
      id: number
      name: string
      surnames: string
      birthdate: string
      location: string
      abilities: object
      appliedJob: integer
      applicationMark: number
      applicationComments: string
      timestamp: string
  Trabajador:
    type: object
    properties:
      id: number
      name: string
      surnames: string
      birthdate: string
      location: string
      abilities: object
      job: string
```

```
/oferta:
  get:
    description: Obtiene todas las ofertas de empleo
    responses:
      200:
        body:
          application/json:
            type: OfertaEmpleado {}
      500:
        body:
          application/json:
            type: object
            example:
              "error": "No se ha podido obtener el recurso"
  post:
    description: Crea una oferta de empleo
    body:
      application/json:
        type: object
        properties:
          category: string
          title: string
          description: string
          minimumRequirements: object
        example:
          category: "PHP Programmer"
          title: "PHP Senior Programmer"
          description: "We are looking for a programmer who h"
          minimumRequirements: {
            "PHP": 8,
            "JavaScript": 8,
            "HTML": 8,
            "CSS": 5
          }
    responses:
      200:
        body:
          application/json:
            type: object
            properties:
              idOferta: integer
              ofertaCreadaCorrectamente: boolean
            example:
              idOferta: 1
              ofertaCreadaCorrectamente: true
      500:
        body:
          application/json:
            type: object
            properties:
              created: boolean
            example:
              "created": false
```

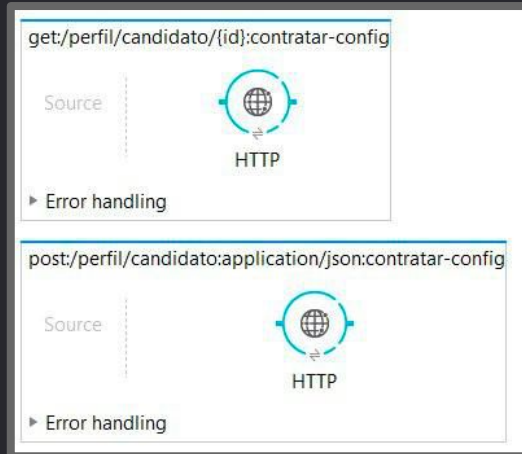
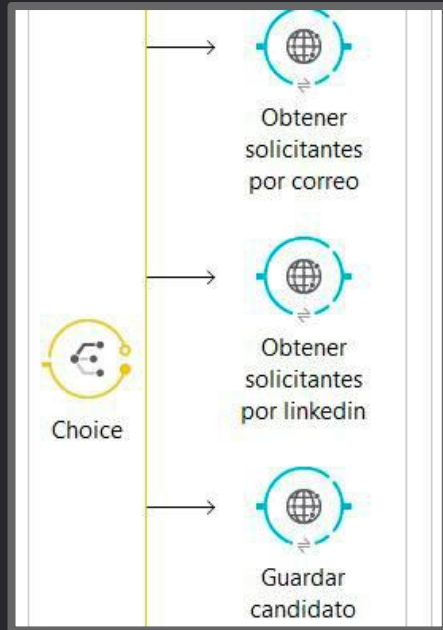
```
/idOferta:
  get:
    description: Obtiene una oferta de empleo
    responses:
      200:
        body:
          application/json:
            type: object
            properties:
              id: integer
              category: string
              title: string
              description: string
              minimumRequirements: object
              published: boolean
              timestamp: string
            example:
              id: 0
              category: "PHP Programmer"
              title: "PHP Senior Programmer"
              description: "We are looking for a programmer"
              minimumRequirements: {
                "PHP": 8,
                "JavaScript": 8,
                "HTML": 8,
                "CSS": 5
              }
              published: false
              timestamp: "2018-01-03"
      500:
        body:
          application/json:
            type: object
            example:
              "error": "No se ha podido obtener el recurso"
  /publicar:
    post:
      description: Publica la oferta
      responses:
        200:
          body:
            # declare content of response
            application/json: # media type
              type: object
              properties:
                idOferta: integer
                ofertaPublicadaCorrectamente: boolean
              example:
                idOferta: 1
                ofertaPublicadaCorrectamente: true
```

Resources

Collapse All

▼ /oferta	POST	GET
/oferta/{idOferta}	GET	
/oferta/{idOferta}/publicar	POST	
▼ /perfil		
/perfil/solicitante		
/perfil/solicitante/{id}	GET	
/perfil/solicitante/correo	GET	
/perfil/solicitante/linkedin	GET	
/perfil/candidato	POST	GET
/perfil/candidato/{id}	GET	
/trabajador	POST	GET

Contratar personal (Mule)



Global Element Properties

HTTP Request Configuration

Create reusable HTTP request manually or by adding your REST API definition

General | TLS/SSL | Proxy | Authentication | Sockets | Notes

Generic

Name: HTTP_Request_APIContratarPersonal

URL Configuration

Protocol: HTTP

Host: localhost

Port: 3001

Base Path: /contratar

API Configuration

REST API Location: contratar.raml

Other Settings

☒ Use Persistent Connection

OK Cancel

Display Name: HTTP

General Settings

Connector Configuration: HTTP_Request_APIContratarPersonal

URL Settings

Path: /perfil/candidato

Method: POST

Parameters

header Name: Content-Type Value: application/json

Add Parameter

Manufactura de paneles solares (RAML)

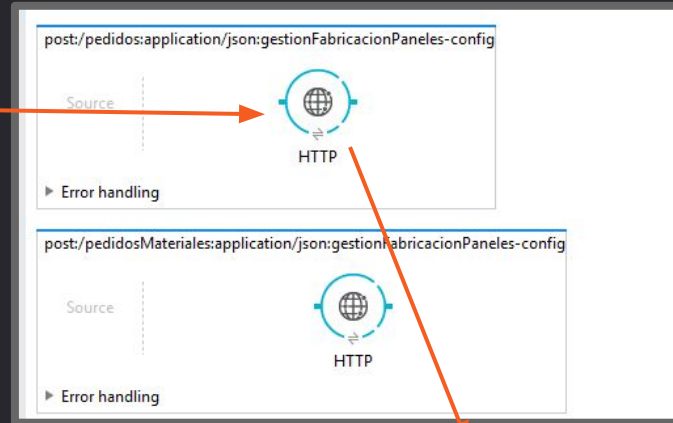
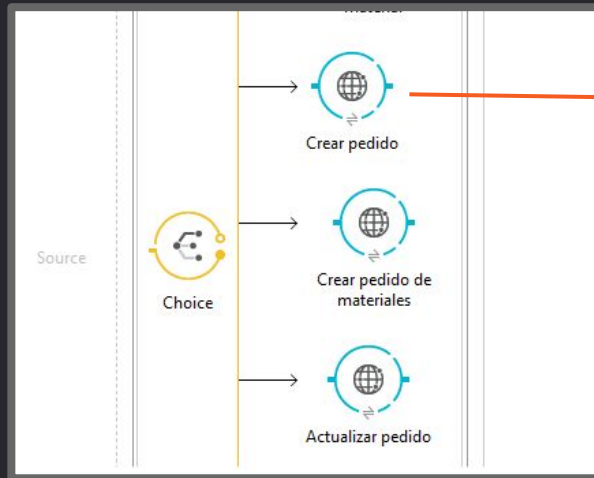
```
/pedidos:
  get:
    description: Devuelve una lista con todos los pedidos
    responses:
      200:
        body:
          application/json:
            type: listaPedidos
  post:
    description: Para crear un nuevo pedido
    body:
      application/json:
        required: true
        type: pedidoInfo
        example: |
          {
            "panelesEncargados": 3,
            "estado": "En fabricacion",
            "cliente": 5
          }
    responses:
      201:
        body:
          application/json:
            type: correctResponse
            example: |
          {
```

Resources

[Collapse All](#)



▼ /pedidos	POST	GET
/pedidos/estado		
/pedidos/estado/{estado}		GET
/pedidos/{id}	PUT	GET
/pedidos/{id}/siguienteEstado		GET
▼ /materiales		GET
/materiales/{id}	PUT	GET
/pedidosMateriales	POST	GET

Manufactura de paneles solares (Mule)



Display Name:

General Settings

Connector Configuration:  

URL Settings

Path:

Method:

E-Commerce (RAML)

```
#RAML 1.0
title: ecommerce
mediaType: application/json
#baseUrl: localhost:3002/ecommerce
baseUrl: https://mocksvc.qax.mulesoft.com/mocks/d3897616-82f4-4ae6-848c-c20d1173dbde/ecommerce

types:
  Producto:
    type: object
    properties:
      Nombre: string
      Cantidad: integer
      PrecioU: integer
  ProductoResponse:
    type: object
    properties:
      id: integer
      Nombre: string
      Cantidad: integer
      PrecioU: integer
  Factura:
    type: object
    properties:
      idProducto: number
      idCliente: number
      Cantidad: number

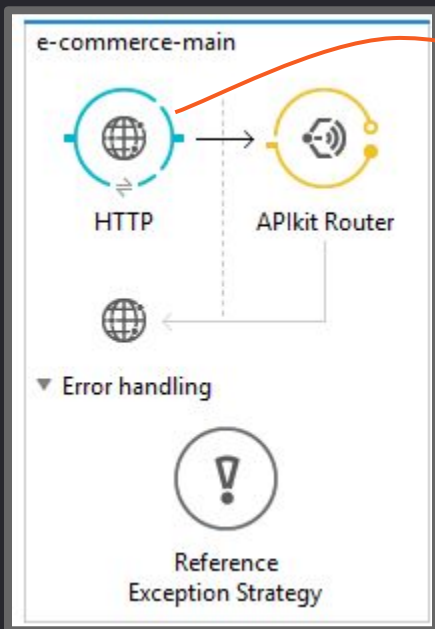
#Available resources
/stock:
  get:
    description: Get all the stock
    responses:
      200:
        body:
          type: ProductoResponse[]
```

Resources		Collapse All	
▼ /stock		POST	GET
/stock/{id}	DELETE	PUT	GET
▼ /factura		POST	
/factura/{id}	DELETE		

E-Commerce (Mule)



E-Commerce (Mule)



Display Name: HTTP

General Settings

Connector Configuration: e-commerce-httpListenerConfig

Basic Settings

Path: /ecommerce/*

Allowed Methods:

URL Configuration

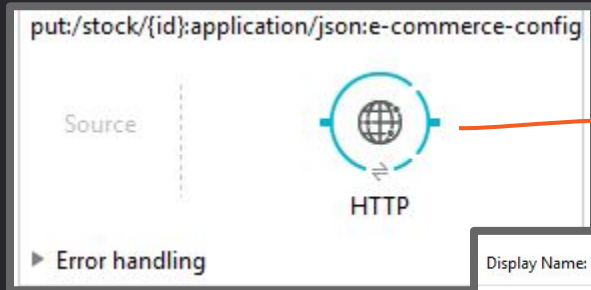
Protocol: ☒ HTTP (Default) ☐ HTTPS

Host: All Interfaces [0.0.0.0] (Default)

Port: 9092

Base Path:

E-Commerce (Mule)



Display Name: HTTP

General Settings

Connector Configuration: HTTP_Request_APlcommerce

URL Settings

Path: /stock/{id}

Method: PUT

Parameters

uri-param	Name: id	Value: #[message.inboundProperties.'http.uri.params'.id]
header	Name: Content-Type	Value: application/json

Add Parameter

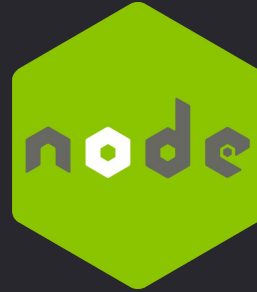
4. TECNOLOGÍAS:



Tecnologías usadas



MuleSoft®



5. INTEGRACIÓN:



Pasos para la integración

1) Creación de **bbdd** con XAMPP

REST

2) Creación de los **Raml** y definición de **APIs**
Desarrollo de los **endpoints** de la API mediante Node.js

SOAP

Creación de los **servicios web**
Orquestación con **Bpel**

Organización de los **flujos** en Mule ESB

- ☐ Componente “Choice” evalúa ruta del cliente y selecciona flujo

3)

Cliente

- ☐ Un controlador por cada proceso (BPEL/API)

Pruebas

- ☐ Postman
- ☐ Debugger de Anypoint Studio

6. DEMO:





¡GRACIAS!

¿Alguna pregunta?



Iván Mora Maciá
Joaquín Vasalo Vicedo
Arancha Ferrero Ortiz de Zárate
Jaime Moreno Franco
Álvaro Muñoz Delgado
Jaime Sarrión Sahuquillo