HCL Universal Orchestrator 2.1
Portal Order processing Demo Pack

Scenario 2

The Scheduler's perspective

Workload Automation Technical advisor's team

Table of Contents

1. Scene 1: Cover

Welcome to HCL Universal Orchestrator.

Your mission, as a Scheduler, is to Model end-to-end processes and interconnect business flows from multiple API endpoints. Performing this demo you will learn how to create different Task types, receive data from multiple API endpoints and filter the exact piece of information you need. You will also be able to move the data you got around, between the tasks.

Steps:

1. Once your solution is deployed, access to the solution console using the credentials from Active Sandboxes pages as listed on Scenario 1.
2. Open the Design view from UnO 2.1 UI.

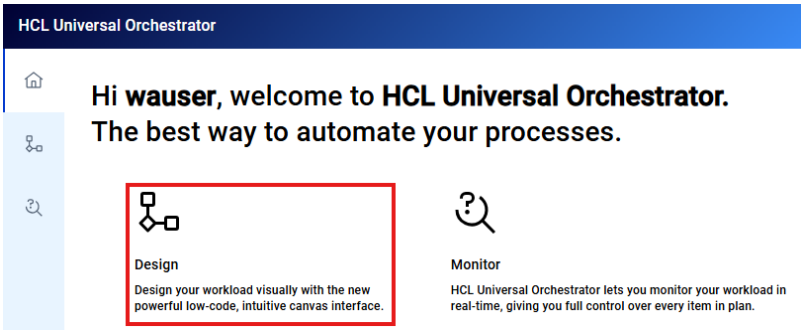

Figure 1 Open Design view

3. Open the ORDER_PROCESS Workflow created on scenario 1 while drag ‘n drop the Workflow into the canvas.

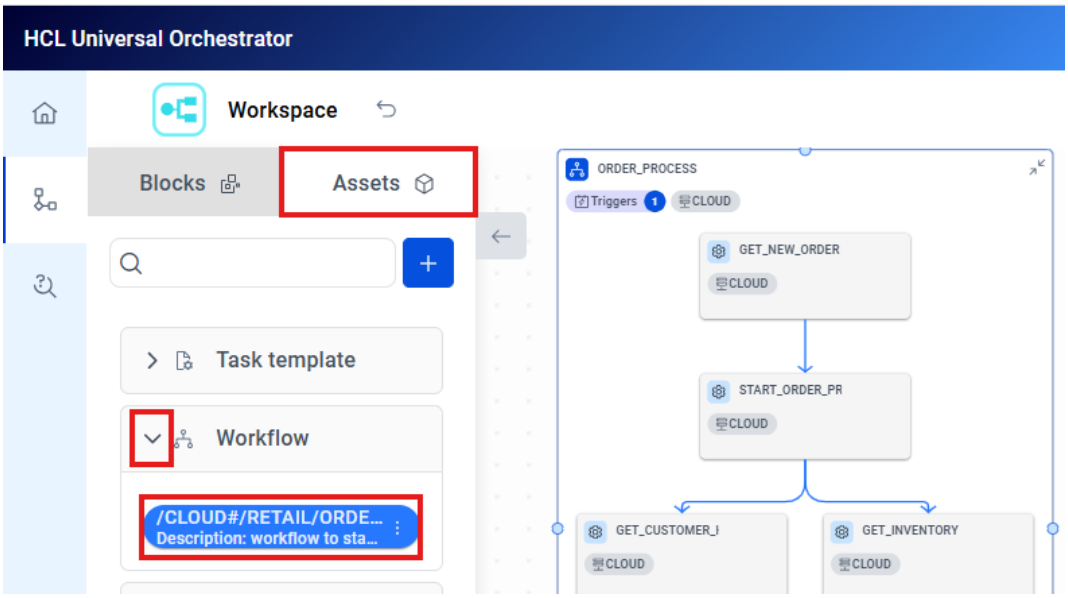

Figure 2 ORDER_PROCESS Workflow

2. Understand the Workflow

The below picture shows the entire Workflow, we will go through each step which has no code nor scripts to manipulate data or conditions.

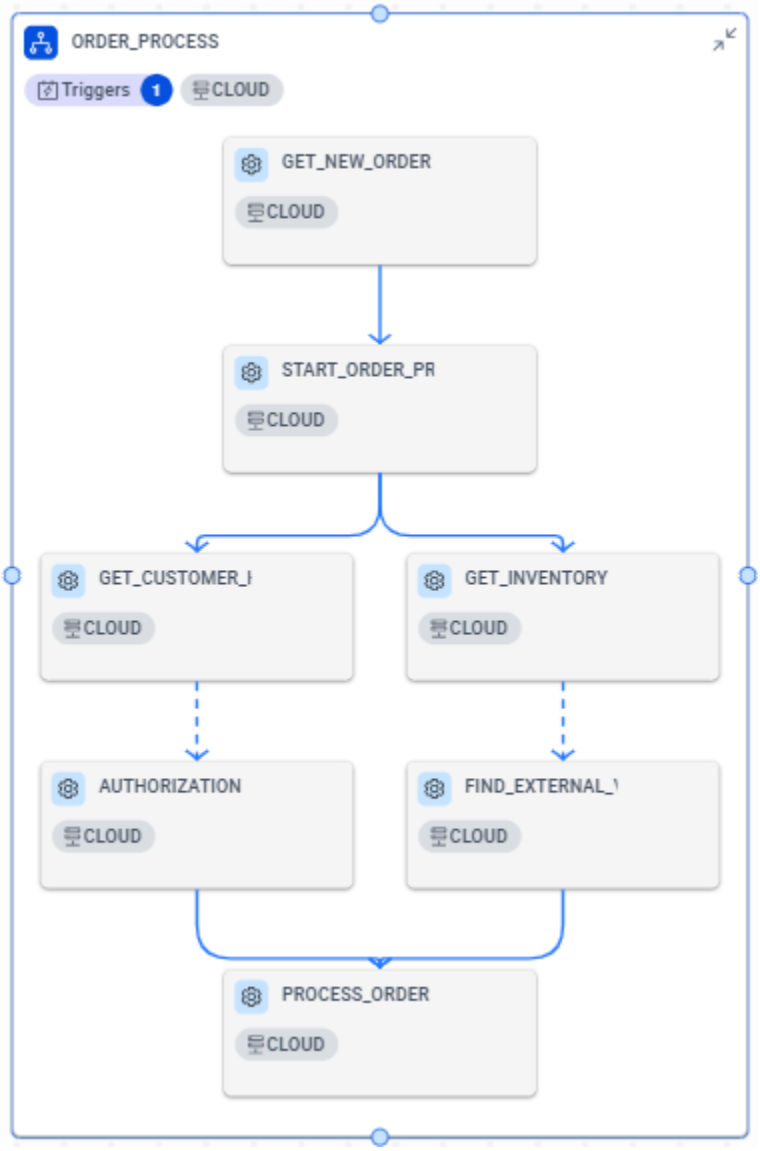

Figure 3 ORDER_PROCESS Workflow

Because the Task inside the Workflow are not actually embedded inside this demo flow, you have to show the properties of the Task while selecting and editing themselves.

1. **Task: GET_NEW_ORDER**

For showing the first Task GET_NEW_Order drop down the “Task template” and select the three dots beside the Task GET_NEW_ORDER and click “Edit”and find the properties on the right side of the canvas (for any other Task it’s the same).

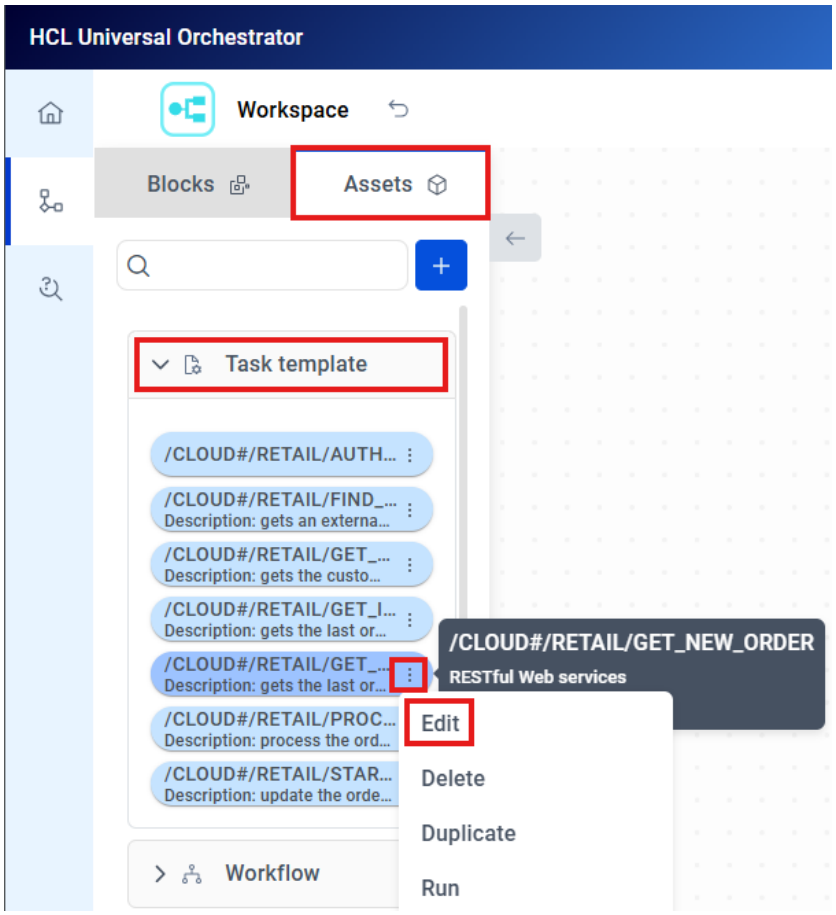

Figure 4 Edit GET_NEW_ORDER Task

Inside the “Action” area you see that we perform a GET on:

```
http://${var.HOST}:3030/v1/orders/last
```

To view the corresponding result & output of this Task, you have to select the Monitor view, click the successful executed Workflow and then the “Task” option inside the toolbar of the execution table.
Hint: you can also select “Task” inside the drop down menu on the upper left side to navigate directly to all executed Tasks.

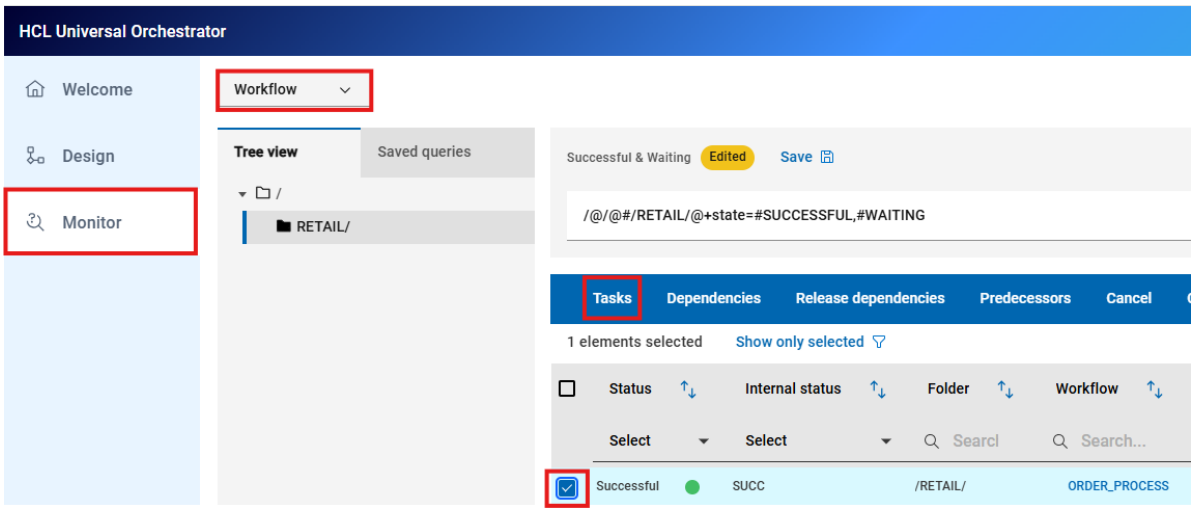

Figure 4 Selecting ORDER_PROCESS Workflow to open corresponding Tasks

For opening the job log select the first Task GET_NEW_ORDER and click the “Job log” option:

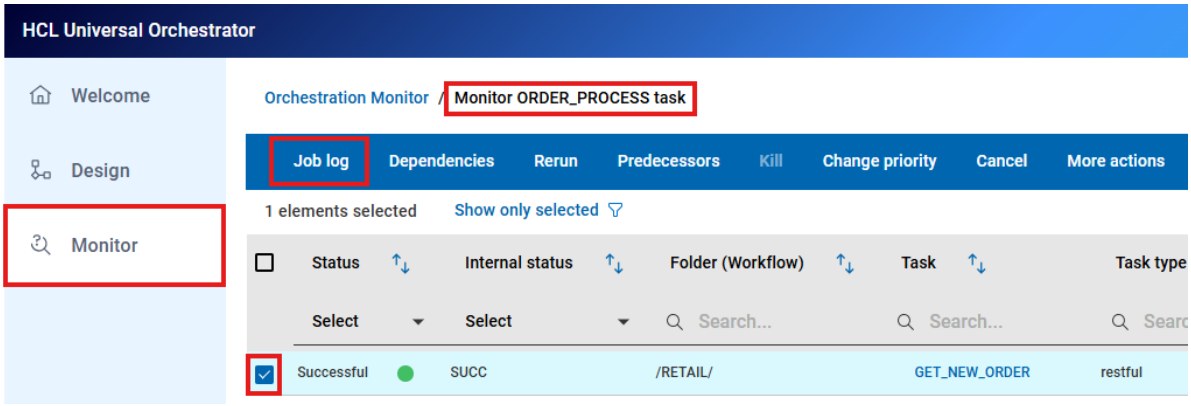

Figure 4 Selecting ORDER_PROCESS Workflow to open corresponding Tasks

After scrolling down inside the Task log details you find the following output:

```
{
    "id": 12345,
    "customerId": 321,
    "name": "John",
    "items": [
      {"id": 123, "description": "Shoes", "qt": 2, "price": 199.99},
      {"id": 234, "description": "Hat", "qt": 3, "price": 49.99}
    ]
  }
```

2. **Task: START_ORDER_PROCESS**

Will perform a **POST**:

```
http://${var.HOST}:3030/v1/orders/update?id=${jobs.GET_NEW_ORDER.JSONResult.id}
```

3. **Task: GET_CUSTOMER_HISTORY**

Perform a GET on inside Task:

```
http://${var.HOST}:3030/v1/customer?id=${jobs.GET_NEW_ORDER.JSONResult.customerId}
```

Output Body inside result (job log):

```
{
  "customerId": 321,
  "name": "John",
  "lastMonthOrders": 10,
  "lastWeekOrders": 5,
  "payingIssues": 0,
  "maxPastOrder": 300
}
```

Corresponding JSONata functions:

```
  $this().JSONResult.(
  $sum($$.jobs.GET_NEW_ORDER.JSONResult.items.(qt * price)) < maxPastOrder*2
  and lastMonthOrders>0
  and lastWeekOrders<30
  and payingIssues=0)=false
```

4. Task: GET_INVENTORY

Perform a POST on endpoint:

```
http://${var.HOST}:3030/v1/inventory
```

With Body (inside job log):

```
${J:jobs.GET_NEW_ORDER.JSONResult.items.{'id':id, 'description': description, 'qt':qt}}

[
  {"id": 123, "description": "Shoes", "qt": 2},
  {"id": 234, "description": "Hat", "qt": 3}
]
```

Will produce a result (inside job log) such as:

```
{
  "available": true,
  "details": [
    {"id": 123, "description": "Shoes", "qt": 21},
    {"id": 234, "description": "Hat", "qt": 32}
  ]
}
```

There is also a conditional dependency associated with the Task:

```
$not($this().JSONResult.available)
```

5. Task: PROCESS_ORDER

Will perform POST on:

```
http://${var.HOST}:3030/v1/orders/process
```

With the Body:

```
${J:(
  $lo:=jobs.GET_NEW_ORDER.JSONResult;
  $inv:=jobs.GET_INVENTORY.JSONResult;
  $v:=jobs.FIND_EXTERNAL_VENDORS.JSONResult;
  {
    'order': $lo,
    'total': $sum($lo.items.(qt*price)),
    'internal':  $inv.available,
    'vendor': ($inv.available ? {} : $sort($v, function($l, $r) {$l.total > $r.total})[0])
  }
)}
```

Will produce a result such as:

```
{
  "order": {
    "id": 12345,
    "customerId": 321,
    "name": "John",
    "items": [
      {"id": 123, "description": "Shoes", "qt": 2, "price": 199.99},
      {"id": 234, "description": "Hat", "qt": 3, "price": 49.99}
    ]
  },
  "total": 549.95,
  "internal": true,
  "vendor": {}
}
```