Oracle Integration

Error Handling & Best practices

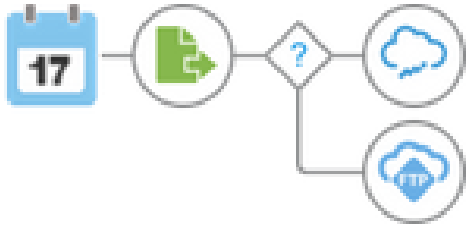
Integration Patterns

Different Integration Patterns have different Requirements

App Driven Orchestration

Multi-step Integration flow invoking applications, integrations and processes triggered by an Application or API.

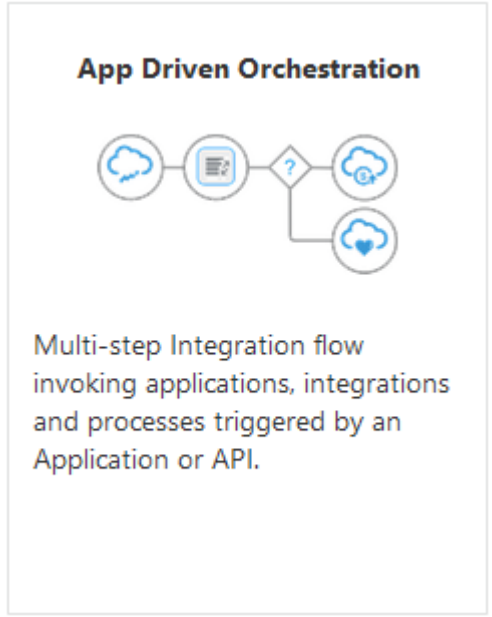
Scheduled Orchestration



Multi-step Integration flow invoking applications, integrations and processes triggered by a Schedule. Commonly used for Batch/Bulk Integrations or File processing.

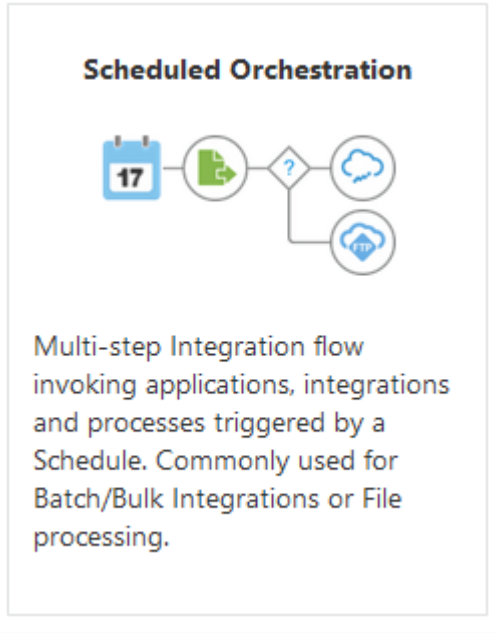
App Driven Orchestration

- Orchestrated integrations can be synchronous or asynchronous types.
  - **Synchronous Integrations** need to send a response back to the caller.
    - There will be different considerations for Post vs. Get requests.
    - These are not recoverable using OIC Error Hospital
    - In an error scenario, you might want to send a customer readable error message.
    - Or handle the error, raise an internal notification and send back a success message.
  - **Asynchronous Integrations** return a 202 Status Accepted Response to the requester.
    - There is no invoker directly awaiting for a response in case of success or error.
    - These can be resubmitted using OIC Error Hospital, resubmission success will depend on the error type.



Scheduled Orchestration

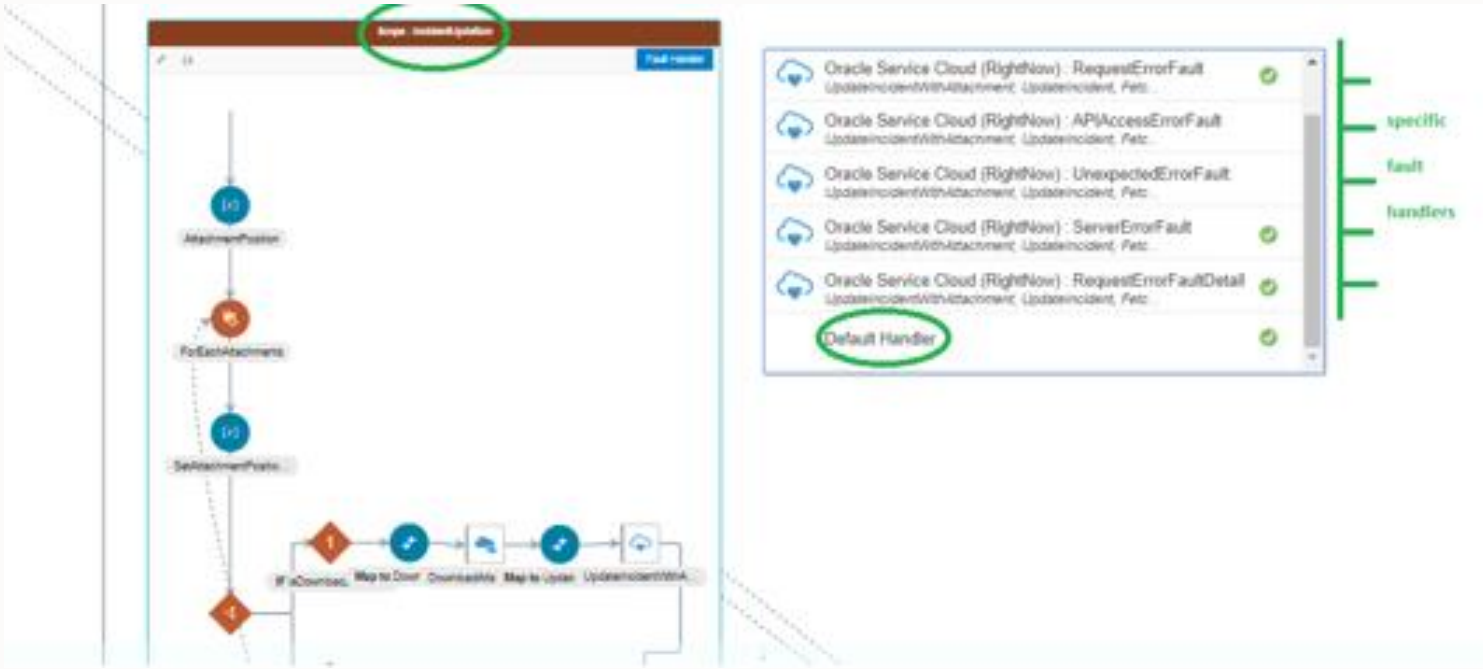
- Scheduled Orchestrations are started at a certain predefined time/schedule or submitted manually/via API
  - From an error handling perspective they behave like Asynchronous calls i.e. there is no invoker application/user waiting for a direct response.
  - Errors can be resubmitted by the OIC Error Hospital. Success of these resubmit will depend on the type of error e.g. if the third party application was not accessible due to an outage, once the application is up you resubmit this. Or an application error was thrown that you fixed in application and later resubmit the error.



Fault Handlers

Scope Level

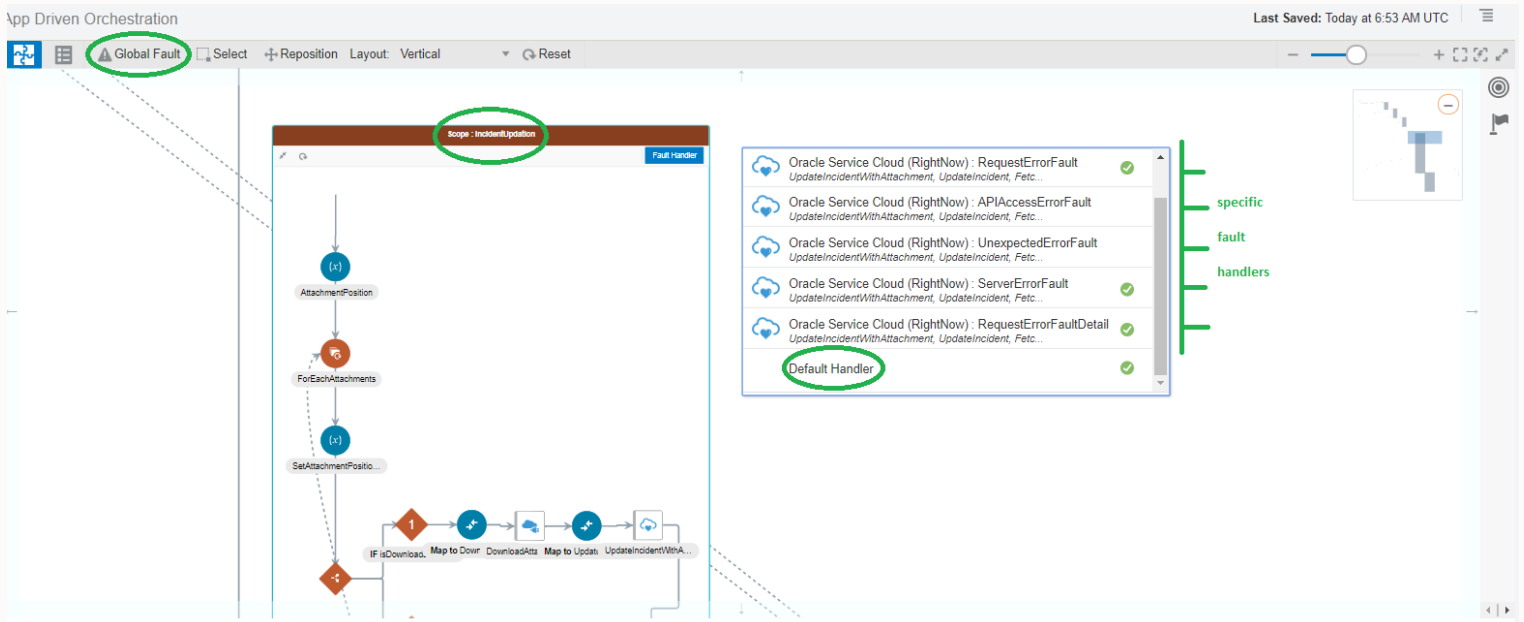
- The scope action is essentially a collection of child actions and invokes that can have their own fault handlers
- Specific fault handling (Catch a specific fault)
- Default Handler (Catch All)
- *Scopes can have fault handlers in which specific faults can be caught and re-thrown. However, in the case of connectivity agent-based invokes, the named fault handlers are not executed. All fault handling must be done in the default fault handler.*
- *Variables created inside a scope action or a looping action are not directly accessible outside the scope, if needed use Global Variables*



Fault Handlers

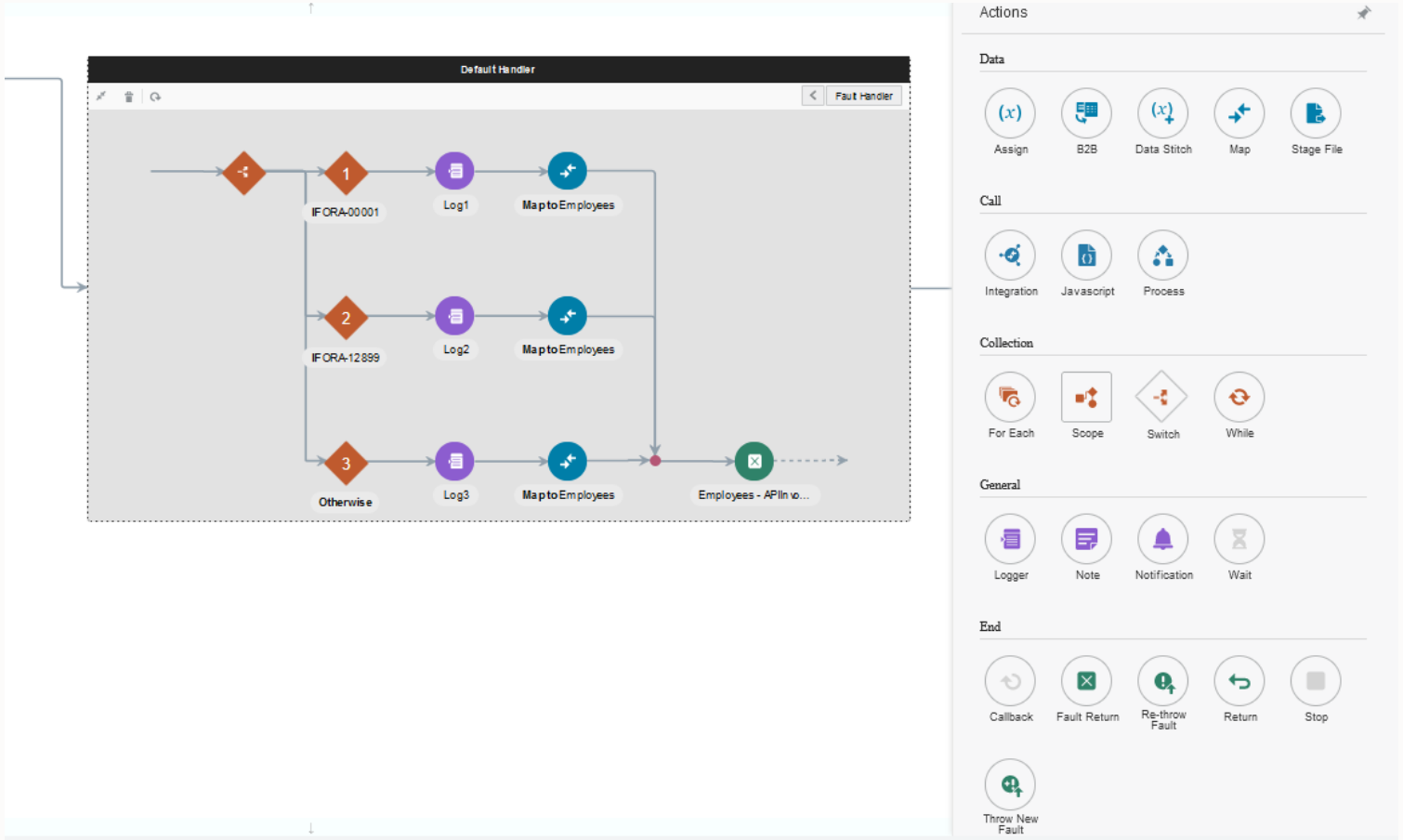
Global Level

- Available at the overall integration flow level which acts as a super catch block.
- Any errors un-handled at the scope level will bubble up to the global fault handler.
- This functionality enables you to direct business faults back to the caller or apply business logic before sending faults to the error handling framework



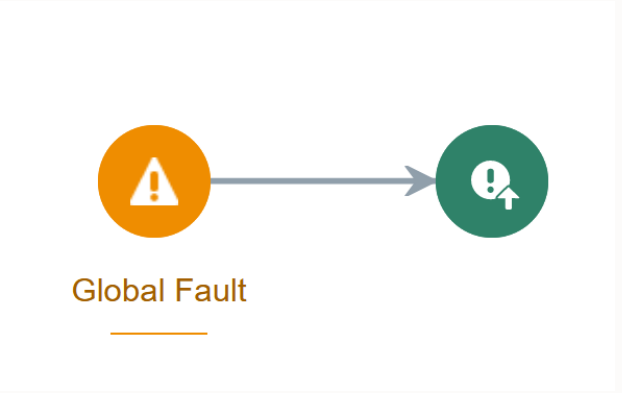
Actions in Fault Handler

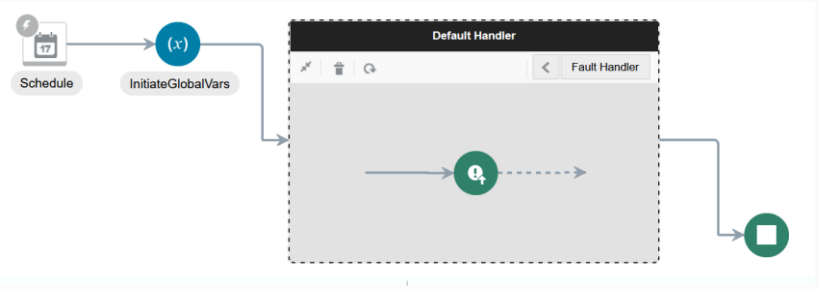
- Almost all Actions can be called in Fault Handlers
- You can build complex logic using collections (For, Switch, While), call other integrations, processes, applications & databases, throw a different error or just return a success message.
- Check for various possible error codes and conditionally perform actions
- Invoke a downstream endpoint and notify it of errors
- Locally invoke a process flow in Process Automation to initiate a workflow involving manual intervention for exceptional flow



Catch Faults with a Re-throw Fault Action

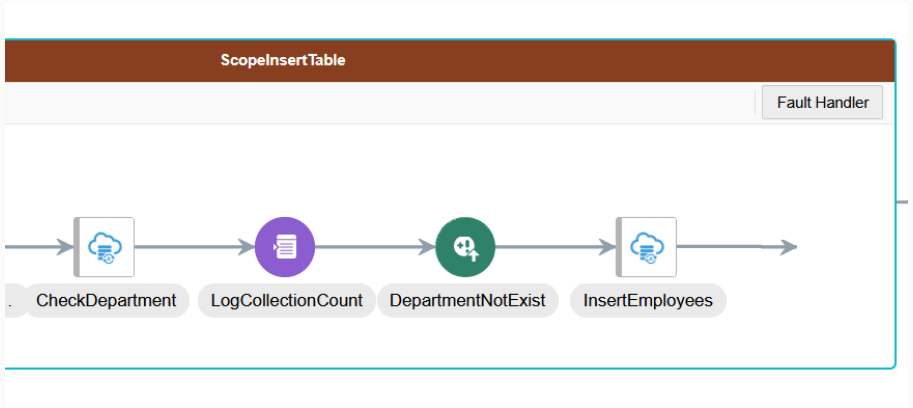
- You can send failed messages to the error hospital for further analysis with a re-throw fault action
- The re-throw fault action can only be placed inside the fault handler section of a scope action.
- The re-throw fault action operates as a catch all block and is processed if a fault is thrown by an invoke action in the scope.
- Integration flow gets terminated at Re-throw fault action
- If the integration contains a defined global fault, the error captured by the re-throw fault action is sent through the global fault and onto the error hospital for analysis

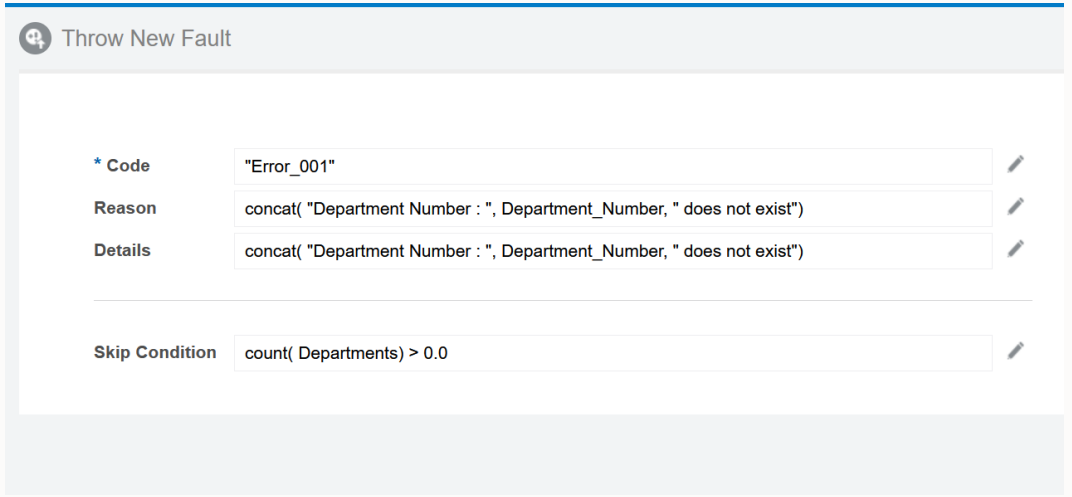




Throw Faults with a Throw New Fault Action

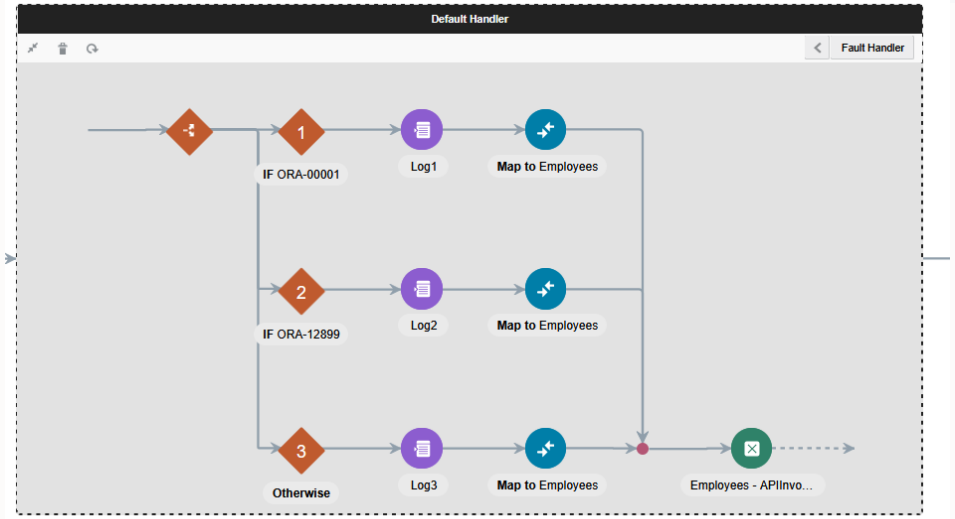
- You can create and throw your own faults in an integration with a throw new fault action.
- Throw New Fault action can be added anywhere in the integration.
- You define the Code*, Reason, Details and Skip Condition.
  - Code is mandatory field.
  - Skip Condition allows you to provide an expression that if true the Fault will be thrown otherwise not.

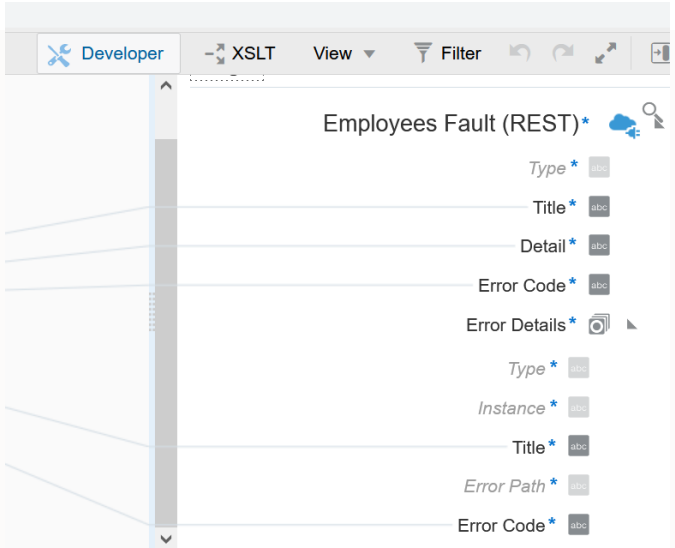




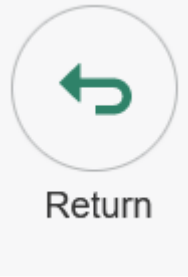
Return a Custom Fault with Fault Return

- This returns a fault to the caller.
- This activity is only available for synchronous integrations as with asynchronous flows there is no caller to return a fault to.
- If a trigger defines multiple faults, you are prompted to select a fault and then a mapper is added with that fault.
- Fault Return can be added to Scope, Fault Handler and Global Fault.
- It terminates the integration and return the mapped/defined fault to the invoker.
- You can use this activity to send a different Error Code e.g. 404 Not Found Error instead of default 500 Internal Server Error that OIC returns in case of error.

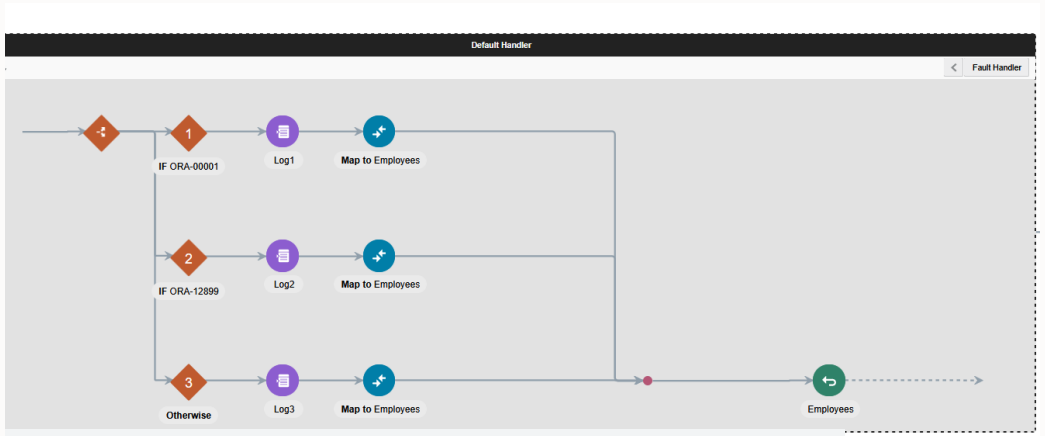


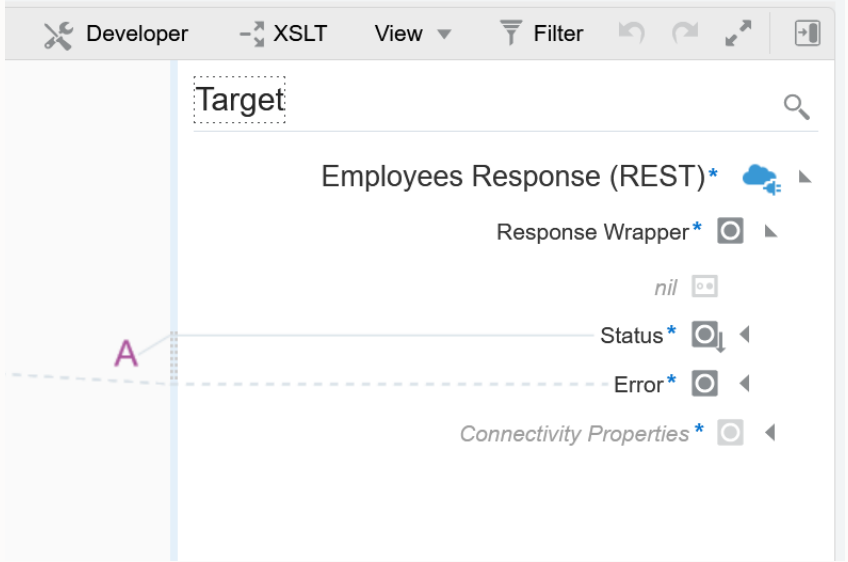


Return an immediate Response

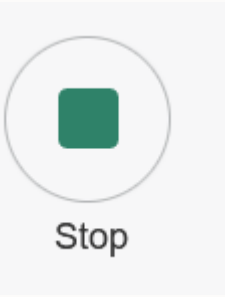


- This returns an immediate response i.e. success back to invoker.
- This activity is only available for synchronous integrations as with asynchronous flows there is no caller to return a response to.
- You can handle error and still send a successful response i.e. 200 OK back to invoker

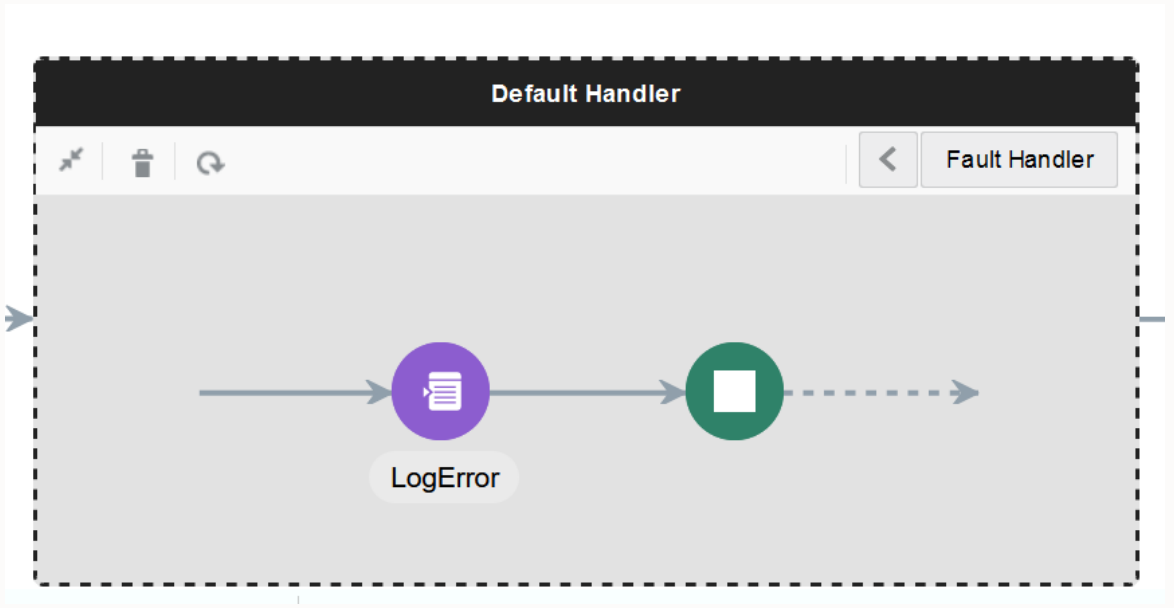




Stop Terminate the Integration

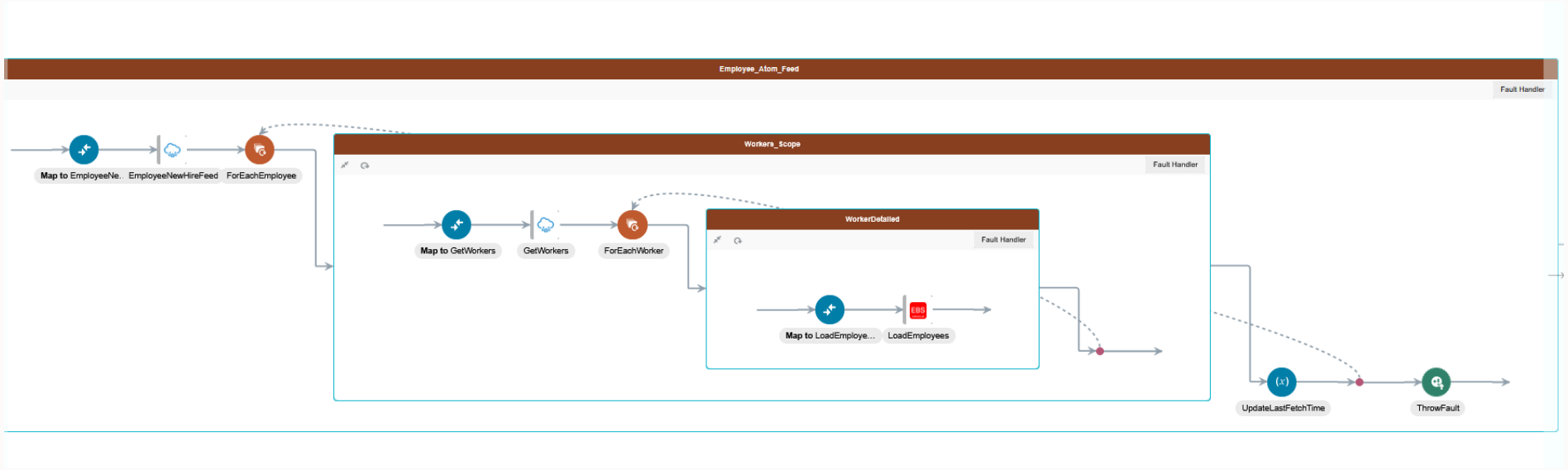


- No return message is sent to the trigger.
- This activity is only available for asynchronous integrations.
- Regular 202 Accepted status code is sent back.



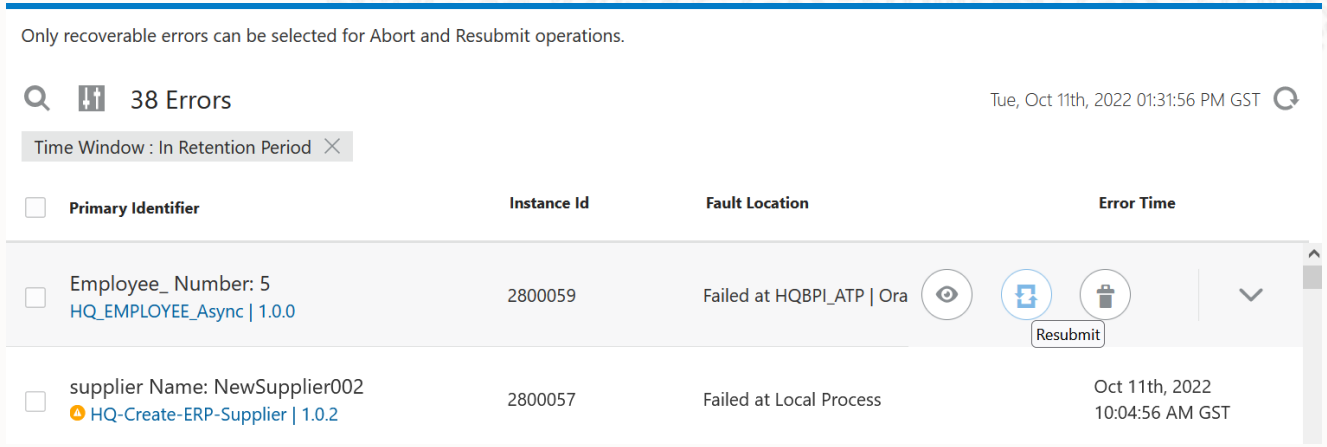
Fault Handlers in a Nested Scopes

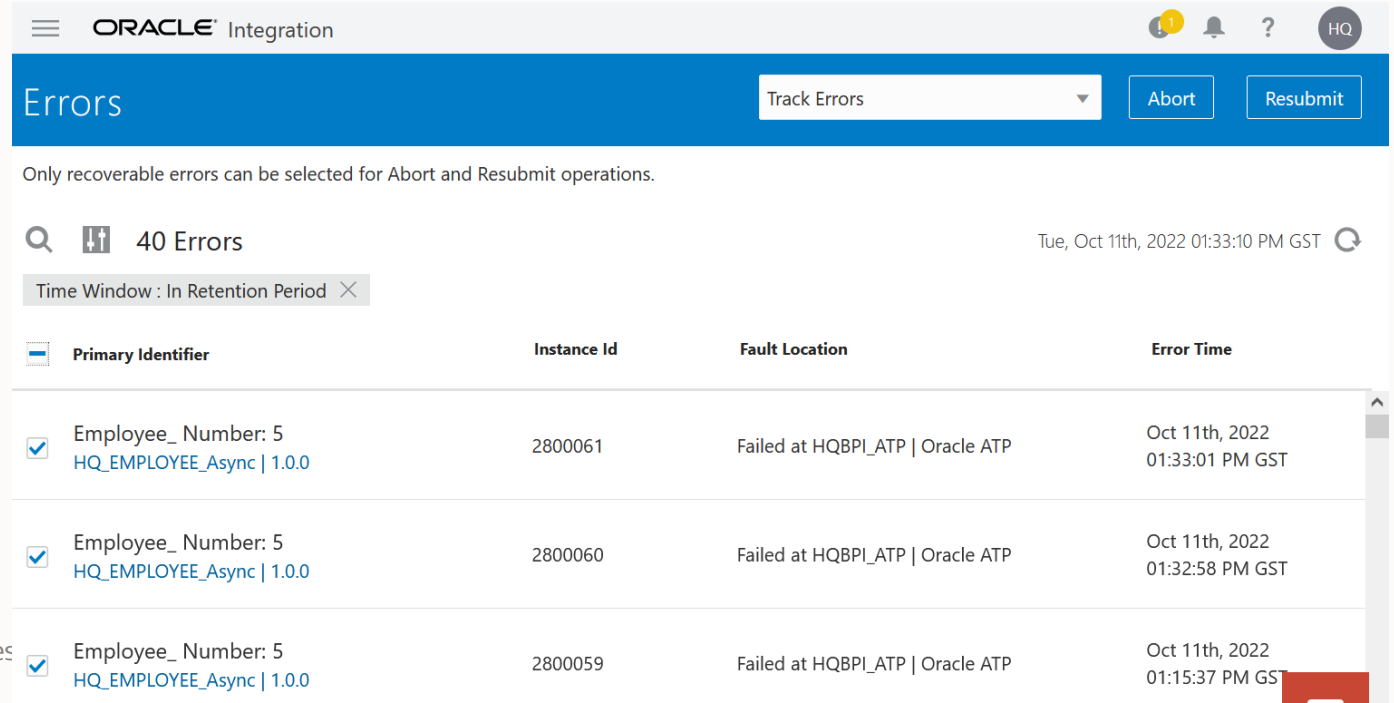
- Fault handlers can be configured in nested scopes. Errors bubble to the immediate scope if not handled
- Recommended: If the fault handling is becoming complex, handle the same by implementing a separate error handler flow and call the error handler from the parent flow



Resubmit Failed Messages

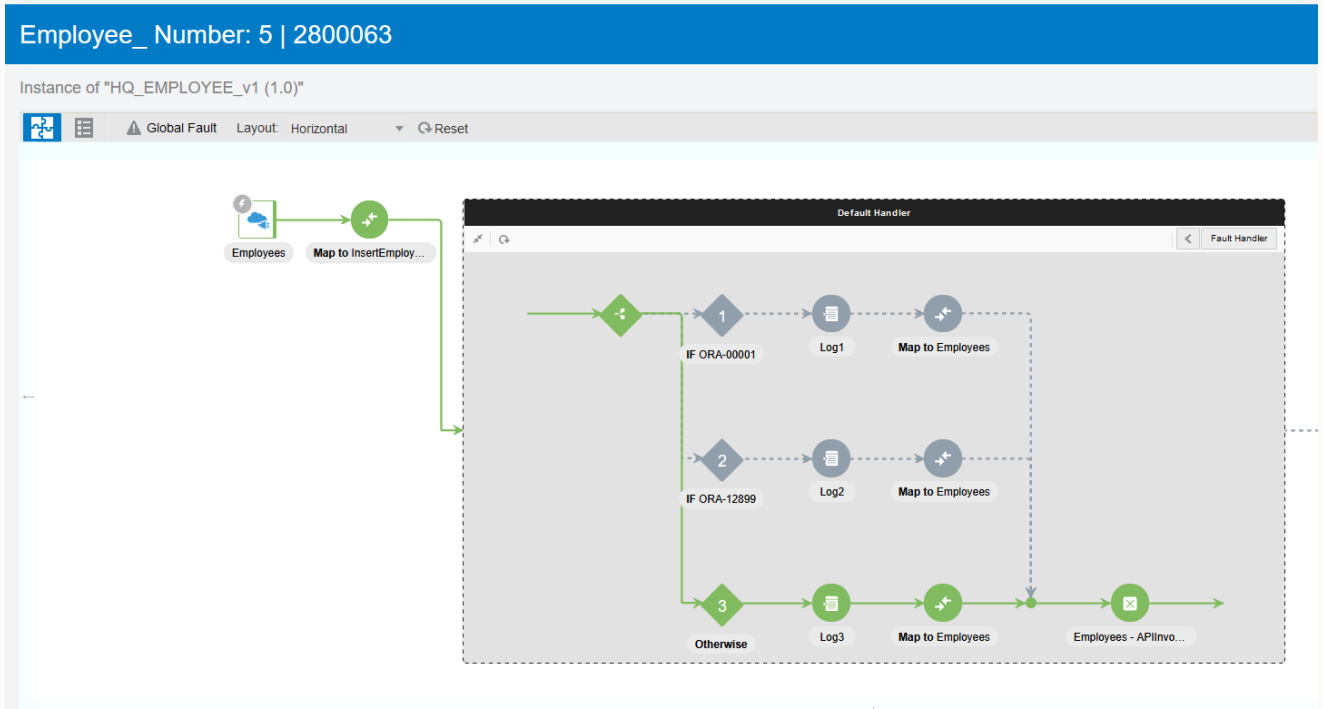
- You can manually resubmit failed messages, Oracle Integration does not automatically resubmit failed messages.
- Resubmission starts from the beginning of the integration.
- Only asynchronous integrations can be resubmitted, synchronous integrations cannot be resubmitted.
- Resubmit can happen in following ways.
  - Single failed message resubmissions
  - Bulk failed message resubmissions

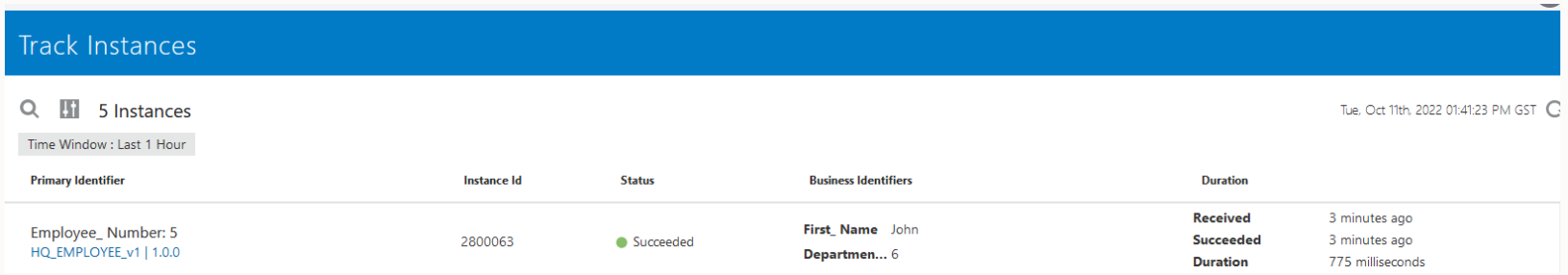




Instance Status

- Notice that the instance Status on Track Instance page for the Return Fault Action results in Succeeded, meaning error has been successfully handled.





Resources

- Downstream Throttling in Oracle Integration Cloud via Parking Lot Pattern
- Best practices for building resilient asynchronous integrations
- Alert Notification Accelerator
- Error Handling Guide - Oracle Integration Cloud
- Advanced Error handling and Scheduling Best Practices - Oracle Integration Cloud
- Fault Actions Behavior in OIC
- Retry logic implementation in OIC

Thank You

[Date]