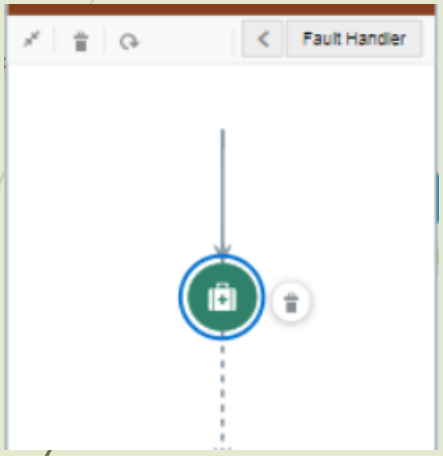
Exception Handling & Logging

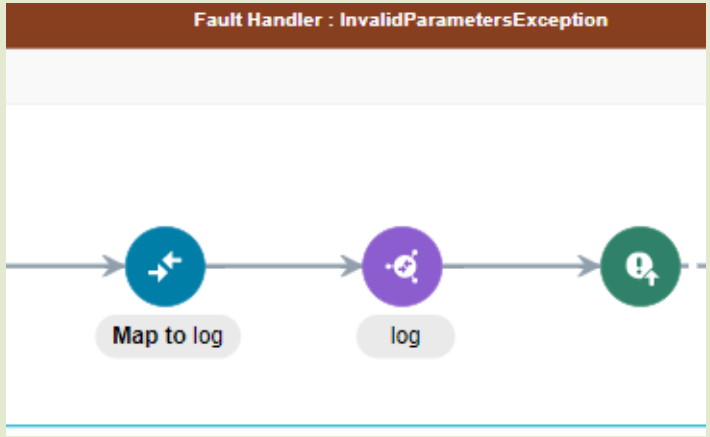
- In OIC, exceptions can be handled at two levels – **Scope Level** and **Integration/Global Level**.
- At Scope Level, if any named faults defined in the invoke actions that will generate separate fault handlers and **Specific Fault Handlers** can be implemented to handle such faults.
- Besides these, a **Default Handler** is there for every scope which can be thought of generic fault handler like CATCH ALL in SOA or Exception in java or WHEN OTHERS in PL/SQL.
- If any exception is not caught by the specific fault handler then the fault will be handled by this Default Handler.
- Now if any exception is unhandled or rethrown in the local scope or not handled in the integration, the exception will be handled by **Global Fault Handler**.
- Usually, all generic fault handling logic is written in the Global Fault Handler such as sending out notification to admin/ developers once the fault is handled and if required instances can be resubmitted.
- Varieties of Logic that can be implemented inside a fault handler – Logging of errors, Terminate, Notify, Invoke Downstream applications etc. – will see in the next slide
- Two major components under Monitoring – First, Dashboard to monitor health of services, connections, integrations being run. Second, Error Hospital to check for the errors and resubmit/ discard individual integration or in bulk. will explore in the upcoming slide(s)
- Debug messages or error log messages can be stored in Database Table or VBCS BO.

Exception Handling & Logging

- **Implementations inside Fault Handler?**
  - Raise Error and Terminate the flow



  - Invoke another Integration to log the error into a Database table and rethrow/ continue

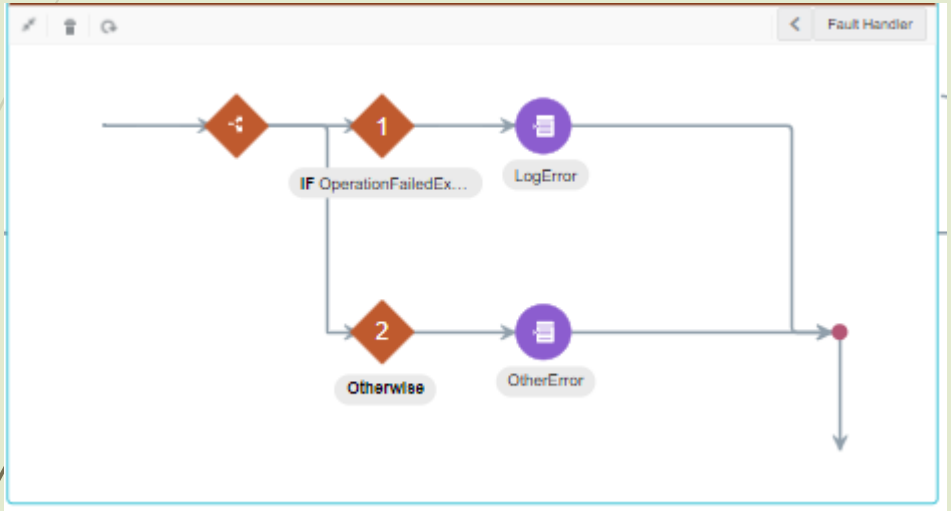


  - Return Error to the Caller (Synchronous Integrations)

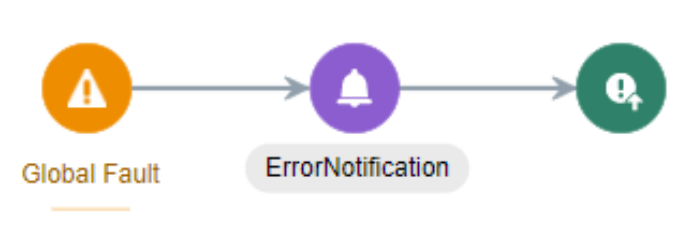
  - Append the error in a local stage file and place to a DB/ File or FTP Server

Exception Handling & Logging

- **Implementations inside Fault Handler?**
  - Check for different error types – Log using Logger



  - Send Notification to Admins/ Developers



  - Invoke a Process Workflow in PCS

Exception Handling & Logging

- **View Errors**

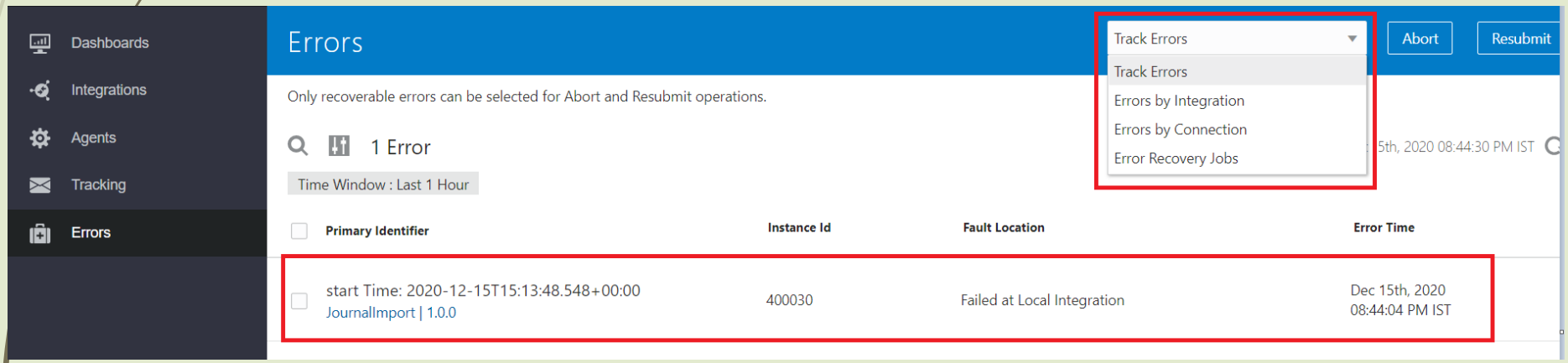
  Navigation: Monitoring > Integrations > Errors

  a) Can View specific error details for an integration instance.

  b) Error is presented in two modes:

     i. Summarized for quick understanding.

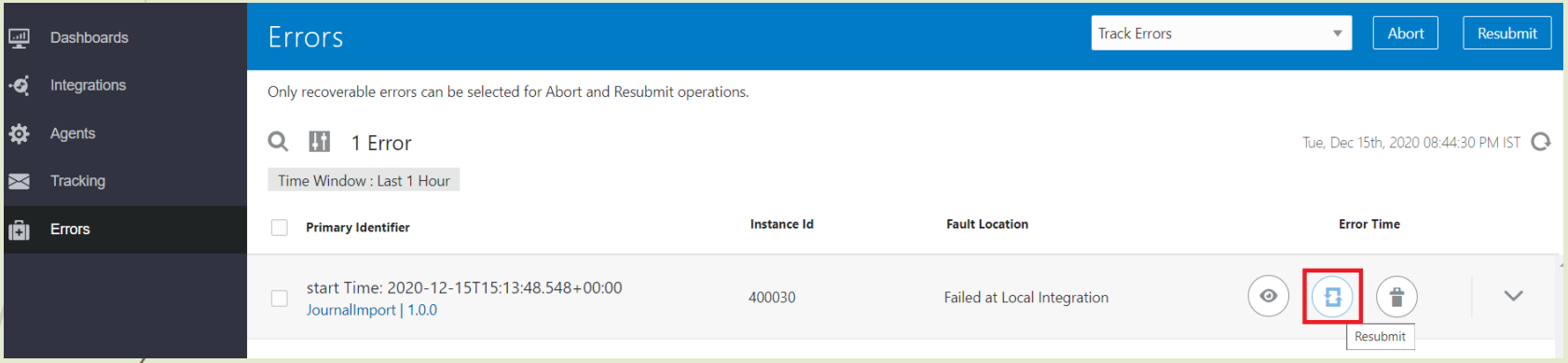
     ii. Detailed at the activity level within the integration



Exception Handling & Logging

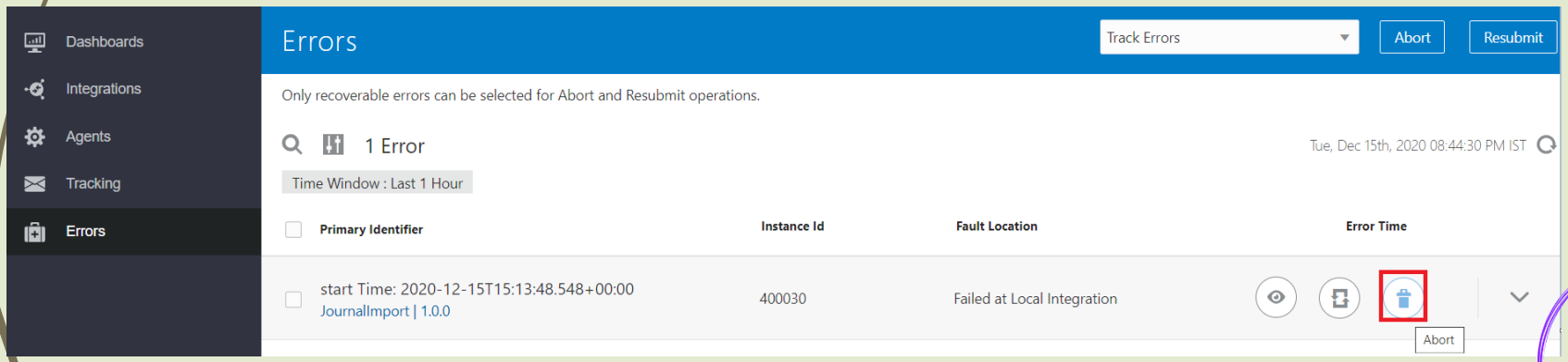
- **Resubmit a Failed Integration**

  A failed integration can be resubmitted manually.



- **Abort a Failed Integration**

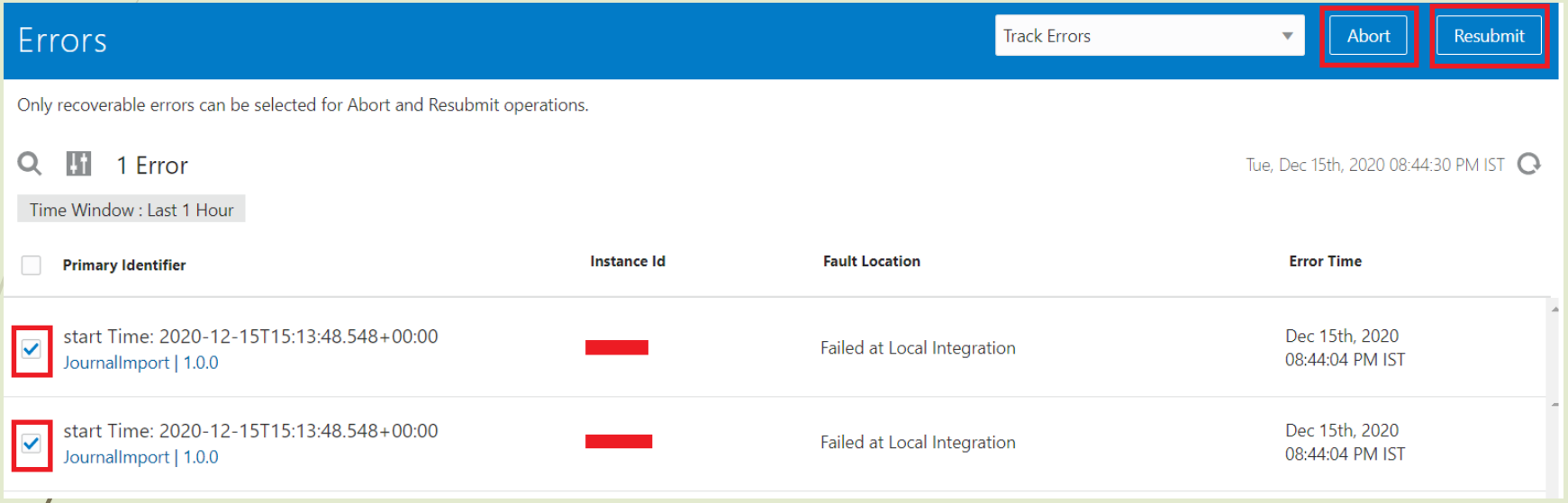
  A failed integration can be aborted manually too.



Exception Handling & Logging

- **Bulk Resubmit/ Bulk Abort all Failed Integrations**

All the failed instances can be resubmitted or aborted in bulk.



Note: Only Asynchronous Services can be Resubmitted/ Aborted**. For Synchronous Service these options will not be available.

*** Caution should be taken while resubmitting any asynchronous flow. Since the integration will start from beginning and start executing previously executed activities that might lead to data inconsistency – so the repeatable activities should be idempotent in nature. Look at the Best Practices of Exception Handling in the next slide.*

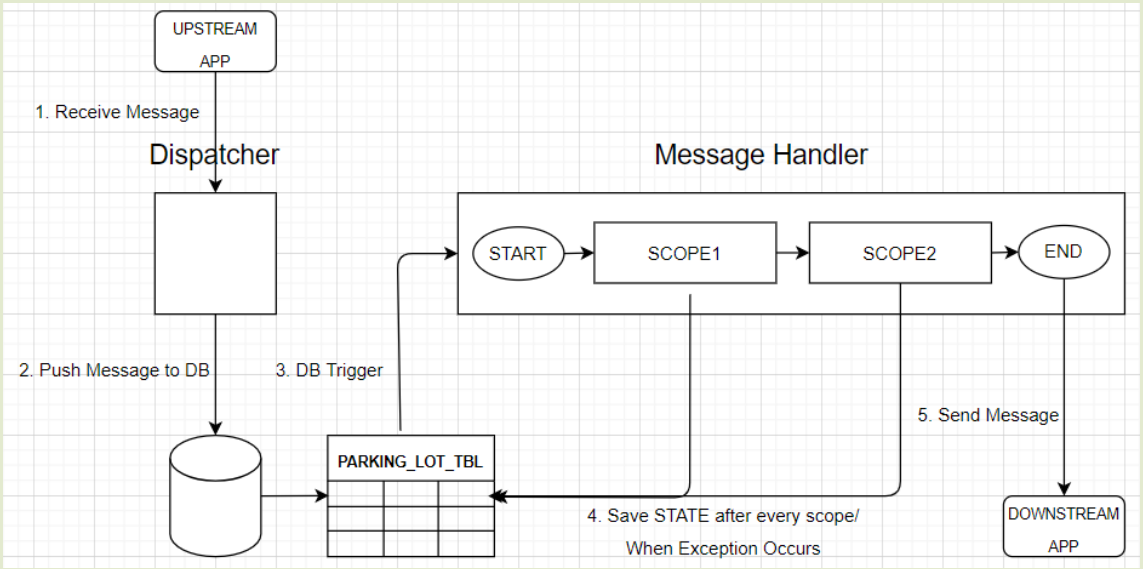
Exception Handling & Logging

Exception Handling – Best Practices

1. Handle Exceptions in scope level for different types of Exception with Different Handlers along with Default Handler. Create Error/ Raise Error in case of Business Logic Fault
2. Generic Implementations such as sending emails, fail the instance etc. should be handled inside Global Fault Handler.
3. In order to resubmit failed instances – to avoid execution of previously executed activities a very popular design pattern can be implemented – **The Parking Lot Pattern**. Database can be used to implement this design pattern – to track the state and storing the metadata in a database table.

Parking Lot Pattern – Scenario 1: App Driven Integration (Asynchronous)

1. **Dispatcher Service** receives the message from Upstream application and saves the message to a Database Table.
2. **Message Handler Service** will poll for a new record or can be triggered.
3. Now, after execution of each scope save the STATE be it success or exception occurs.
4. Now in happy path scenario, the message will be sent to Downstream App.
5. When the instance fails and needs to be resubmitted, the integration will start from the point of failure, not the beginning.



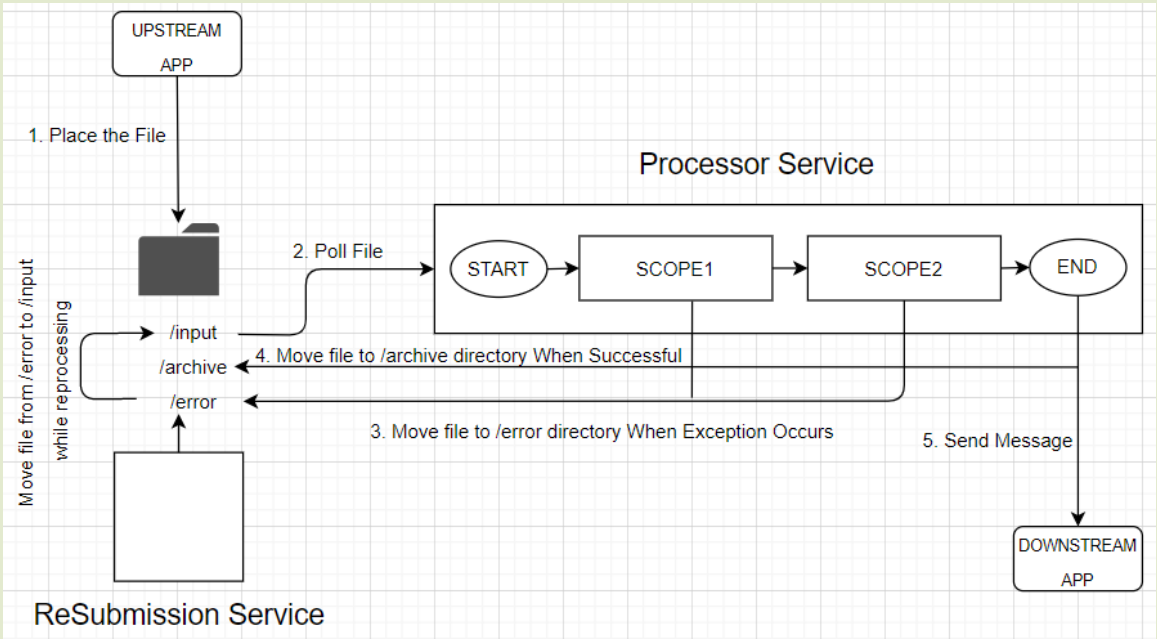
Exception Handling & Logging

Exception Handling – Best Practices (Continue…)

6. The Parking Lot database Table would look like PARKING_LOT_TBL(ID **NUMBER**, PAYLOAD **CLOB**, STATUS **VARCHAR2**, MESSAGE **VARCHAR2**)
7. Database can be Database – Be it On-Premise, or DBaaS or ATP. Except ATP (using wallet), all other databases need agent to connect to from OIC. So Agent can be setup in HA mode.
8. Delete or Archive the Data once processed or schedule purging at regular interval to ensure faster performance of the integration as increase in volume of data in this table may hinder the performance. Moreover create indexes to improvise the performance.

Parking Lot Pattern – Scenario 2: Scheduled FTP/ File based Integration (Asynchronous)

1. Upstream application will place the file in /input directory.
2. Processor Service will poll the file from /input directory.
3. For happy path scenario, the file will be archived to /archive directory before sending the message to downstream applications
4. In case of exception, the file will be placed to /error directory.
5. Another flow, ReSubmission Service will move the file from /error directory to /input directory and the Processor Service will poll again to process.

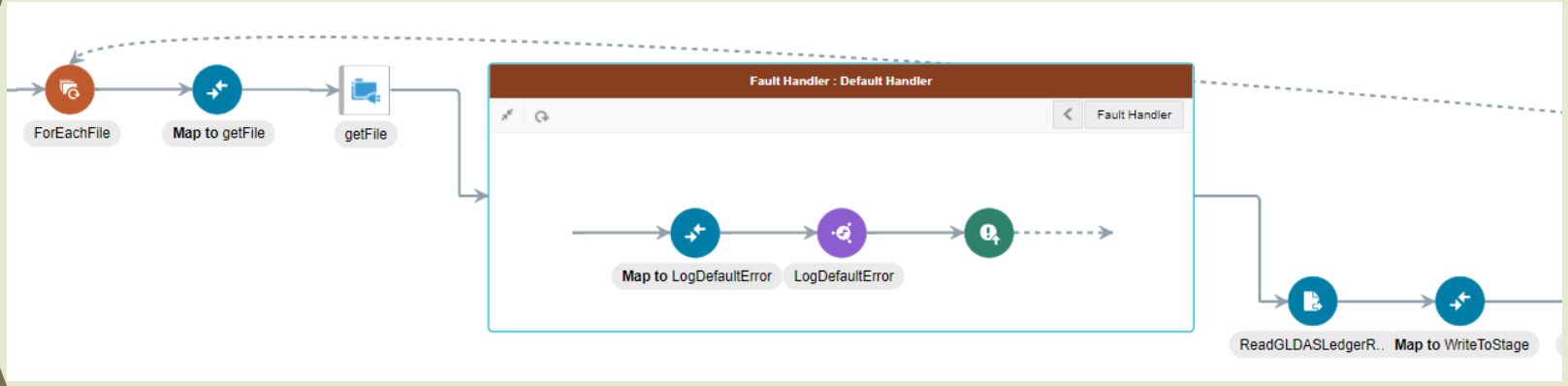


Exception Handling & Logging

- Use Case – **Handle Exception while invoking BI Report in Journal Import Integration**
- Integration will be modified as described below:

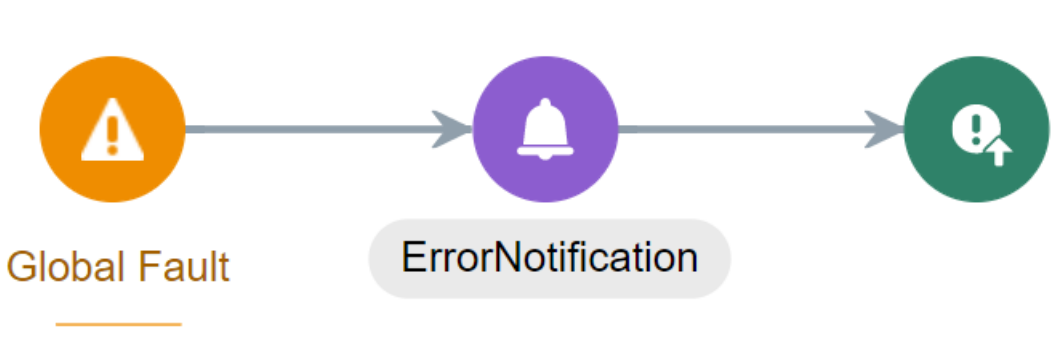
1. The BI Report Invocation part will be included into a scope and it will be like

2. Exception handled at local scope will look like this. Default Fault Handler is configured to invoke Publish to OIC Integration to push the messages to OIC Managed Topic and which will be consumed by the Subscribe To OIC integration to log the exception in a Database table. Then Rethrow the exception to Global Fault Handler.



Exception Handling & Logging

3. In Global Fault Handler, notification activity will notify about the exception to user(s) through email and then Rethrow so that OIC instance fails



4. Sample email will be like: