# OIC Learnings:

Earlier it is called as ICS which includes only integrations, later Oracle added couple of services and names it as OIC.

OIC mainly consists of 3 components

1. Integrations

2. VBS (Visual Builder)

3. PCS (Process Cloud Service)

Billing:

|  |  |
| --- | --- |
| ERP | Number of Users |
| OIC | Msg/hour |

OIC is of 2 editions, Enterprise and Standard.

**Challenges:**

1. Making cloud Applications talk to each other (with in single vendor)
2. Make Applications talk to each other btw the products of different cloud vendors.
3. Making Cloud Applications to talk with On-Premise products.
4. Legacy style integrations are usually too complex.
5. Data couldn’t be moved easily from cloud to on-premise and vice versa.
6. Business Process not aligned with application Integrations.
7. Analytics.

**OIC Solves:**

1. Configurations based Approach (No Code Needed).
2. Easy to integrate Cloud App with On-Premise Application (Needs Connectivity Agent).

* Packages help to deploy integrations from one instance to another instance.
* Look Ups are used to store the values that might be configurable in the future.
* Libraries allow additions functionalities (Only JavaScript Libraries).

### Integrations Styles:

1. Basic Routing: Allows to connect only 2 applications (will be deprecated in the latter versions).
2. File Transfer:

To share content from one SFTP Server to another SFTP server. Uses Encryption.

1. Publish and Subscribe:

* Two are different styles of integrations, first we need to configure the publish and then while creating a subscribe it will ask for a publish then we must select one.
* We can select only one subscription in an integration.
* Suppose we need to publish to multiple subscribers say 5, then we need to create 5 subscribe integrations for that single publisher.

1. App Driven: It needs to be triggered by an Application, it includes all the functionalities provided by an integration.
2. Scheduled: It is same as App Driven but it is triggered on schedule based.

### Global Variables and Data Stitch:

https://www.youtube.com/watch?v=f-11D9PM0aY

If we are dealing with FTP or any other integration which arise any exception it should be handling with in a scope.

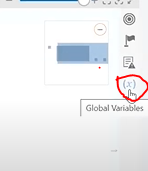
Every scope has its own variables and whatever the inbound and outbound activities invoke under the scope, that response can’t be used outside that scope, in this case Global variable and Data-Stitch become helpful.

*Scenario: To use the response of a scope outside of it.*

**Global Variable:**

* Are variables that are available throughout the integration.
* It is a part of a feature flag -> “oic.ics.console.integration.complex-variables”.
* Its types are String, Boolean, Date, DateTime, Object, Number.

To create a global variable, click on (x) symbol, then give the name and data type, once you select the type as object you can see the sources, then select the response (of the scope) element for which we need to assign to this global variable).



**Data Stitch:**

* Allows us to create assignments to the complex variable.
* It is in the feature flag of “oic.ics.console.integration.stitch-action”.

To create a data stitch, click on flag icon under Data select the data stitch and drag it into the scope, then properties window will open give the name and click on configure, then you need to assign the global variable to it, so that you can use the response out the scope as well.

### For Each Loop:

There are 2 types of loops in OIC,

1. For Each

2. While

**For Each:** Click on flag option 🡪 Under Collection select For Each and drop it, then configure window pops up, give the Name, Description, Repeating Element, Current Element Name.

### Calling a BI Report using Soap Adapter:

SO first you need to develop a report in ERP cloud, we must use a

soap wsdl : http://<cloudERPURL>/xmlpserver/services/v2/ReportService?wsdl?

In Soap connection properties give the name, then in operations select the appropriate Operation

(runReport in this case) .

In mapping you must give credentials, report absolute path, and parameters if any

Under parameter name values.

### Java-Script Library:

In OIC you can see libraries with in the integration work area, here we can add(register) java-script functions or libraries to add extra functionality to our integration.

*Things to be remembered:*

1. Your JS function should have an explicit return variable.

### FTP Adapter in OIC

<https://www.youtube.com/playlist?list=PLlzOCTRJKR1K5r8YS6mgsllTOZijDulRl>

* FTP Adapter is used to retrieve the files for processing in Oracle Cloud and can upload files from OIC to a Directory on a remote FTP Server.
* Uses PGP(Pretty Good Privacy) cryptography to encrypt and decrypt.
* Supports signing and un-signing of files.
* Can connect to FTP, SFTP servers, which are accessible over the internet.
* Enables to create schema file format(created from either a CSV file or an existing schema file) to use the files to transfer.
* Delimiters 🡪 single spaces, commas, semi columns, or tabs.
* Operations Supported :

1. List File
2. Read File
3. Write File
4. Download File
5. Move File
6. Delete File

All the above listed operations are synchronous and provide Responses, and support dynamically passing the headers(directory, file name, etc.….).

* Host key Authentication, Public key Authentication and streaming support for large files.

To create an FTP Connection:

**Use-cases 1**: To Write a file to FTP Server.

1. First create a Connection using FTP Adapter.
2. In integration page click on create and choose App Driven Orchestration. Then give the name and click on create.
3. In the integration add a rest connection to invoke this API
   1. In the properties give the name to this element, give the URI as \writeFile, select POST method.
   2. Check the options as *configure the request payload* and *configure this end point to receive response*. Click on next.
   3. Give the request payload as json in inline click next.
   4. Give the response payload by selecting JSON option and click on next and done.
4. Add the FTP connection and configure the properties.
   1. Give the name to the element.
   2. In operations select **write** operation, select transfer mode, give the output directory and specify the file name pattern, click on next.
   3. Select the respective schema and click on next.
   4. In definition upload a sample file to create xslt and fill the options with appropriate values and click on next and done.
5. Map respective mappings and click on validate. Always click on save while adding a new element as a best practice.

**Use-case 2**: To Read a file from FTP server and send the information in that file as a Response.

1. First create an FTP Connection.
2. Create an Integration and add a Rest connection to invoke this integration, in the properties give the name, URI, and select the operation as GET(since we need information from FTP Server).
   * 1. Select add review parameter for this end point and configure this end point to receive response.
     2. Add request parameter as file name with data type as string.
     3. Configure the response by adding an inline JSON code, click on done.
3. Add FTP Adapter.
   * 1. In basic info give name as Read File.
     2. In Operations, select operation as 🡪Read File, Transfer Mode 🡪 Binary, input Directory 🡪 location of FTP server where the file to be read exists, File Name 🡪 Sample file Name(we will pass the file name as req parameter).
     3. Schema 🡪 Upload a sample CSV file as of file to be read.
     4. Definition 🡪 select delimited data file, record, record-set name 🡪 Done.

In Read File Mapper map the file name from src to target and do the necessary mappings.

**Use-case 3**: To move files from one directory to another directory using FTP Adapter.

1. Create a schedule integration 🡪 MoveFiles
2. List files from source directory using FTP Adapter.
   1. Insert an FTP Adapter, in operation select LIST FILES, give directory path.
   2. File Name Pattern 🡪 Give as \*(All the files in the directory will listed)
3. Loop over files using FOR Loop.
   1. Give the repeating Element as *ListFiles🡪ListFileResponse🡪ListResponse🡪File List🡪File*
4. Move file to target directory using FTP Adapter.
   1. In the loop created above insert an FTP Adapter, and give the operation as Move a File.
   2. Give the source directory Path and Target directory path. Done, Do the necessary mappings.

**Use-case 4**: To transfer files from one SFTP to another SFTP using FTP Adapter.

1. Create a schedule integration.
   1. In the integration page click on create, select scheduled orchestration an configure it’s properties.
2. List files from source SFTP using FTP Adapter.
   1. Select source FTP connection
   2. Select LIST FILES operation
   3. Give the input directory location.
   4. File Name Pattern as \* (to select all the files).
   5. Max Files 🡪 maximum files to list, give as 100
   6. Minimum Age 🡪 last modified date, give as 0.
   7. List Files Recursively 🡪 if check then list the files from sub directory as well.
3. Loop over the files using FOR EACH LOOP.
   1. To read files one-by-one drop a for each loop and give the File under List Response/File List as repeating element.
4. Read file one-by-one using Read Operation.
   1. Add a source FTP connection in the loop to read each file.
   2. Select the operation as READ FILE and give input directory path.
   3. In schema select NO, since we want exact file structure as in source file.
   4. Map directory to directory and file name to file name.
5. Write the file to target SFTP using write operation.
   1. Select Target FTP Connection in the loop after read file.
   2. Select the operation as Write File.
   3. Transfer mode 🡪 ASCII
   4. Give the Output Directory
   5. File name pattern as “\*”
   6. Schema 🡪 No
   7. Map Read File Response/ICS File/File Reference to *ICS File/File Reference.*
   8. File Response/File Name to Out bound FTP header/File Name.

### Stage File

In File processing they may involve different operations, like for inbound files we may need to read or write it by doing unzip or decrypt or both.

In the same way for out bound files we may need to zip it or encrypt it. In such cases stage file is useful.

Operations supported by a Stage File:

1. Read Entire File
2. Read File in Segments
3. Write File
4. Zip File
5. Unzip File
6. Encrypt File
7. Decrypt File
8. List File