Overview

SAP SuccessFactors Recruiting Management Integration with Saudi Arabia's Ministry of Interior package has the iFlow "SAP SuccessFactors Recruiting Management with MOI".

In this integration SAP Cloud Integration receives the assessmentOrderId, queries the JobApplicationAssessmentOrder entity for the necessary fields to communicate with the MOI (and fields required for future SuccessFactors operations), and uses this response to upsert the current fields in the JobApplication. Throughout the processing, the fields regarding the Assessment's status are updated.

This document:

- Describes the information needed to integrate SFSF with the MOI
- Provides reference information for each field in the iFlow parameters
- Provides an inventory of possible errors scenarios that can happen, along with tools that you can use to troubleshoot the integration

Prerequisites

Before working with this iFlow, make sure you have met the following prerequisites:

- Must have the appropriate licensing to use the assessment integration.
- SAP Cloud Integration, integration service account must be provisioned for access to integration flows.
- Must already have acquired the MOI API key (by contacting them) and have the IPs used whitelisted for the service.

This document focuses on the specific integration with MOI. For another vendors/partners, please refer to the generic template over at: https://api.sap.com/package/SuccessFactorsRCMwithThirdPartyAssessmentVendorGenericTemplate/overview

Connection Reference

This list provides reference information for each connection to sender and receivers of the iFlow, including the fields queried, upserted or inserted.

Sender - SuccessFactors_RCM

Connected through SOAP protocol, waits for messages incoming on "/<Incoming Address>" and uses a Request-Reply message exchange pattern to send back a reply. As you can see in the image, the user role ESBMessaging.send is required to connect to SAP Cloud Integration. No other security or conditions are applied. The connection is:

- SOAP_Request

Receiver - SuccessFactors/SuccessFactors_2

Both receiver adapters serve the same purpose, to establish connections to the SuccessFactors Address through **OData V2**. The reason behind there being 2 adapters is merely aesthetical, as if there were only adapter, the connections would have to cross through the whole integration.

The types of connection made to these adapters are:

- **SuccessFactors_GetInfo** (Query to get jobApplication/NationalIDType/externalCode, id, jobApplication/custDOB,jobApplication/nationalID, jobApplication/applicationId, where the id= assessmentOrderId).

- **SuccessFactors_UPSERT_NationalId** (Upsert for applicationId, custDOB, cust_arabicName1, cust_arabicName2, cust_arabicName3, cust_arabicName4, nationalID, cust_dateofissue fields).

- **SuccessFactors_UPSERT_Iqama** (Upsert for applicationId, custDOB, ThirdName, firstName, middleName, lastName, nationalID, cust_dateofissue fields).

- **SuccessFactors_UPSERT_Iqama_Candidate** (Upsert used to keep all instances of firstName, lastName, middleName and thirdName the same. If this UPSERT wasn't in place, even after upserting the fields in **jobApplication** the values would revert to their old value).

Receiver – SuccessFactors_AssessmentReport

This receiver establishes a SOAP connection to SuccessFactors and inserts the iFlow's progress into AccessmentReport, letting us know if SAP Cloud Integration Acknowledged the request, completed it, or failed to complete it. The connection is:

- SuccessFactors_AssessmentReport (Insert with errorCode, field1Key, field1Value, orderId, status, statusDate, statusDetails and 2 extra fields for possible future changes

Receiver – MOI

There are only 2 outgoing **HTTP** connections to the MOI API which are:
- HTTP_NationalId (Which connects to https://<Customer Host>/GetCitizenInfo and provides the required DateOfBirth, NationalIdNumber, and Lang-this last field must exist, but its content isn't used).

- HTTP_IQAMA (Which connects to https://<Customer Host>/GetResidentInfo and provides the required DateOfBirth, IqamaNumber and Lang-this last field must exist, but its content isn't used).

Please note that the value of <Customer Host> in the URL is provided by MOI.

Scripts Used

add_Application

Simple script that adds applicationId to the incoming body. This is done as to append this required field for the mapping that follows. The applicationId is saved right after the first connection to SF.

Throw_Exception

Script that follows a route where the incoming response from MOI returns null values. This means that given ID doesn't match any ID they have. As such, an exception is thrown: "Error: Person not found in vendor service".

set_Headers

This script searches for the "Key_Name" property, which should contain the name of the Secure Parameter that currently stores the MOI-ApiKey. It then saves this parameter to the headers, along with "Content-Type" and its value "application/json".

Gregorian_to_Hijri

Transforms Gregorian dates to Hijri dates, also removes the Time from the DateTime format (to make the date compliant with MOI request parameters).

Hijri_to_Gregorian

Transforms Hijri dates to Gregorian dates, and adds the Time to make it a DateTime format (to make it compliant with SuccessFactors formatting)

Candidate_AddHeader

Adds the candidateId to the message body, since this ID is required for the following UPSERT.

logMaster

Script, found in generic SAP documentation, that makes it possible to save log files regarding the payload and the headers. The Properties that control this script are the following: ENABLE_MPL_LOGGING, ENABLE_FILE_LOGGING and ENABLE_PAYLOAD_LOGGING. (These properties have the possible values of "YES" or "NO").

getProperty

Simple Used for mapping purposes of reaching a property to use inside the message mappings.

Mappings Used

SuccessFactors to MOI IQAMA mapping

Mapping file	Source file	Target file
SuccessFactors_to_MOI_Iqama.mmap	RequestFieldsGET	SuccessFactors_to_MOI_IQAMA_target.xsd

Observation: Only field with modifications is the custDOB, which transforms the date from Gregorian (SuccessFactors format) to Hijri (MOI format).

SuccessFactors to MOI NationalId mapping

Mapping file	Source file	Target file
SuccessFactors _to_MOI_NationalId.mmap	SuccessFactors _to_MOI_NationalId_target.xsd	GET_MOI_Request_Fields_2.xsd

Observation: Only field with modifications is the custDOB, which transforms the date from Gregorian (SuccessFactors format) to Hijri (MOI format).

UPSERT Iqama mapping

Mapping file	Source file	Target file
UPSERT_Iqama_Mapping.mmap	MOI_Iqama_Source.xsd	UPSERT_Iqama.xsd

Observation: Only field with modifications is the custDOB, which transforms the date from Gregorian (SuccessFactors format) to Hijri (MOI format).

UPSERT NationalId Mapping

Mapping file	Source file	Target file
UPSERT_NationalId_Mapping.mmap	MOI_NationalId_Source.xsd	UPSERT_NationalId.xsd

Observation: Only field with modifications is the custDOB, which transforms the date from Gregorian (SuccessFactors format) to Hijri (MOI format).

UPSERT Candidate Mapping

Mapping file	Source file	Target file
UPSERT_Candidate.mmap	MOI_Iqama_source_ids.xsd	CandidateEntityUPSERT.xsd

Assessment Report

Mapping file	Source file	Target file
AssessmentReport_Insert.mmap	Assessment_Source_nonError.xsd	AssessmentReportEntityInsert1.xsd

Possible Routes

In these routes, the orange arrows represent the path executed correctly in each case.

Every route starts by first saving the incoming message's assessmentOrderId, updating the Assessment Report's status to acknowledged, by calling a local integration process, and a query to SuccessFactors using the saved assessmentOrderId and receiving the ID fields along with externalCode.

Error Route

After the first three steps, for whichever path it takes, if the ID and date of birth don't match with the information in the MOI database, an exception is thrown.
This Route can also happen due to some error regarding the mapping of the incoming message (i.e., the response message from MOI had an invalid format).

The thrown exception then triggers the Exception subprocess, that updates the application status to error.
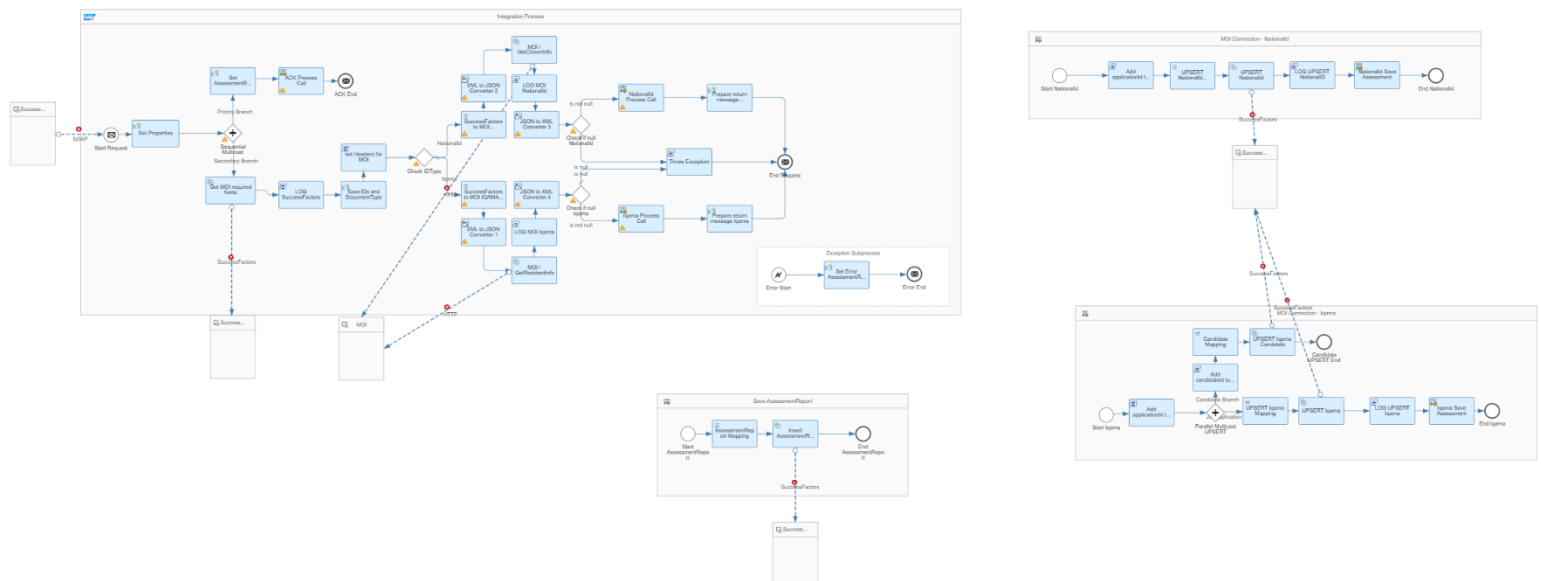
NationalID Route

The direction to follow is decided by the "Check IDType" router. It checks for the response message field externalCode.

In case this field is equal to "sni" – value defined in SuccessFactors for National ID type- then the upper route is chosen.

This path leads to where a message is prepared and then sent to the MOI, containing the Date of Birth, NationalID and lang (a mandatory field, although its content is ignored).

If the response from MOI isn't null or malformed, the message is sent and received into SuccessFactors, after mapping to the desired format.

The success of the upsert is then saved in the AssessmentReport's status, and after that the iFlow concludes.

IQAMA Route

The direction to follow for the step is decided by the "Check IDType" router. It checks for the response message field externalCode.

In case this field is equal to "iqama" – value defined in SuccessFactors for IQAMA ID type- then the lower route is chosen.

This path leads to where a message is prepared and then sent to the MOI, containing the Date of Birth, IQAMA ID and lang (a mandatory field, although its content is ignored).

If the response from MOI isn't null or malformed, the message is sent and received data into SuccessFactors, after mapping to the desired format.

For the IQAMA candidate, since the fields to update (first name, middle name, third name and last name) are present in both the JobApplication and Candidate entities, both of these are upserted, after their own mapping, to keep data consistency.

The success of the upsert is then saved in the AssessmentReport's status, and after that the iFlow concludes.

Helpful tips

- User credentials for SuccessFactors and secure parameters are created under the monitor section – manage security – security material
- Request and Response templates are available for the generic template, but not for MOI
- Throughout the iFlow you will find the logs, from the script file "logMaster". These will be very helpful to troubleshoot any errors since they create a log of the message body, along with properties and headers after important steps.