Integration Package Documentation: Figaf Jira Integration

Version: 1.1.0
Last Updated: 15.08.2025
Contact: Alexandru Florea (afl@figaf.com)

Contents

1. Package Overview

This integration package facilitates communication and process automation between the Figaf DevOps tools and Jira. It enables organizations to synchronize ticket and transport information, automating Jira issue updates based on Figaf events, and providing utilities for querying Jira data.

The package currently includes the following key integration flows (iFlows):

- **Search Jira Issues:** Provides an endpoint to search for Jira issues based on specified criteria, called by Figaf.
- **Update Jira Issues:** Processes webhook events from Figaf to update or create corresponding Jira issues.
- **Update Figaf Transport from Jira:** Allows Jira webhooks to trigger transport imports in Figaf.
- This document provides details on the configuration and functionality of each iFlow.

2.Prerequisites

- SAP Integration Suite (CPI) tenant.
- Access to a Figaf instance with webhook capabilities.
- Access to a Jira Cloud instance with API access.
- Necessary authorizations in both Figaf and Jira for the integration user/API token.
- User Credentials artifact must be deployed on the CPI tenant The credential (ex: JIRA_ApiService_Credentials), containing the Jira API token (e.g., Username: your_jira_email, Password: your_jira_api_token).

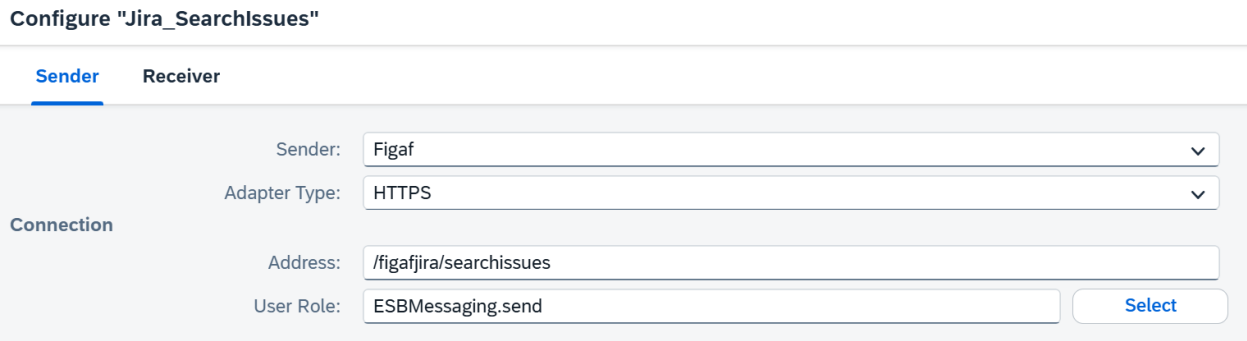
3. Integration Flows

3.1. Search Jira Issues

Provides an endpoint for Figaf (or other systems) to query Jira and retrieve essential field data (key, summary, description, status) for issues within a specified project.

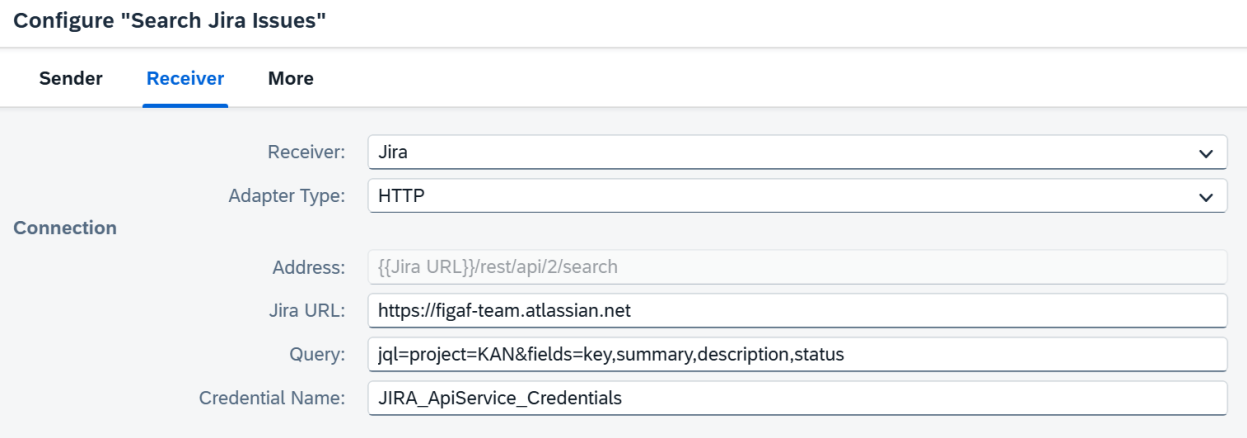
3.1.1. Configuration (from iFlow's "Configure" tab):

Sender Details:



- **Address:** This is the path appended to your Integration Suite tenant URL to call this iFlow.
- **User Role:** The calling system must authenticate with a user having this role.

Receiver Details (Connection to Jira):



- **Jira URL**: The base Jira search API endpoint. This should be changed to your Jira instance's URL.
- **Query**: This needs to be configured by the user to specify their target Jira project (e.g., change “KAN” to their project key) and the fields they require.
- **Credential Name**: The alias of the User Credential artifact in Integration Suite for Jira API authentication.

3.1.2. Using the IFlow

IFlow Trigger: HTTPS call, typically from Figaf.

Input: None explicitly via body for this configuration (JQL is in the adapter).

Output: JSON response from Jira containing the search results.

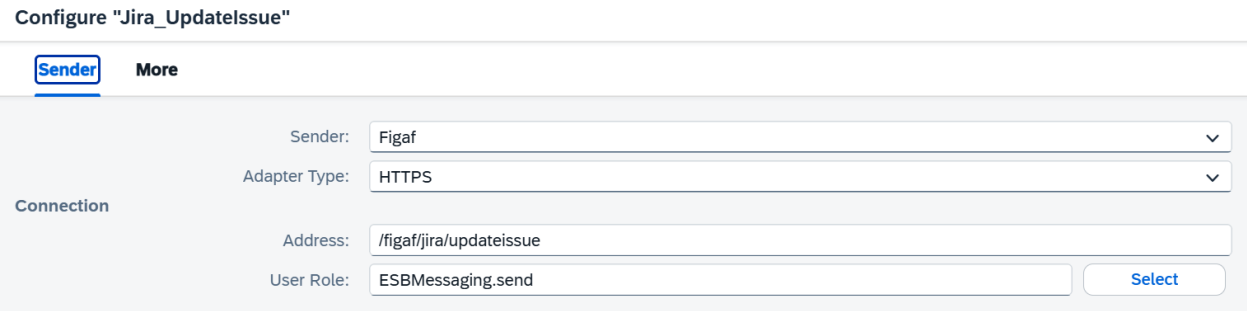

```
[
    {
        "ID": "KAN-3",
        "URL": "https://figaf-team.atlassian.net/browse/KAN-3",
        "Title": "Ticket initiated in landscape overview from trial (DEV) to trial
(QA)",
        "Description": "Description - Test transport of 2 iflows\n\nTracked
Objects:\n- GetData (1.0.1)\n- TestIflow (1.0.8)",
        "Status": "To Do"
    },
    {
        "ID": "KAN-2",
        "URL": "https://figaf-team.atlassian.net/browse/KAN-2",
        "Title": "Transport 2",
        "Description": "Second TP",
        "Status": "To Do"
    },
    {
        "ID": "KAN-1",
        "URL": "https://figaf-team.atlassian.net/browse/KAN-1",
        "Title": "Transport 1",
        "Description": "First transport\n\nTest *transport* _ever_\n\n* Dummy",
        "Status": "Waiting for Customer"
    }
]
```

3.2. Update Jira Issues

This iFlow is triggered by webhooks from Figaf whenever tickets or transports undergo changes. Based on a central JSON configuration, it dynamically performs actions in Jira, such as transitioning issue statuses or adding comments.

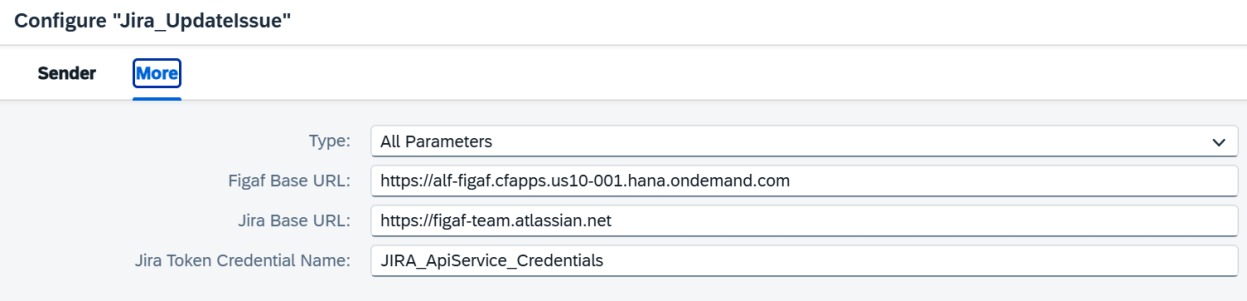
3.2.1. Configuration (from iFlow's "Configure" tab):

Sender Details:



- **Address:** This is the path appended to your Integration Suite tenant URL to call this iFlow.
- **User Role:** ESBMessaging.send as default in this case.

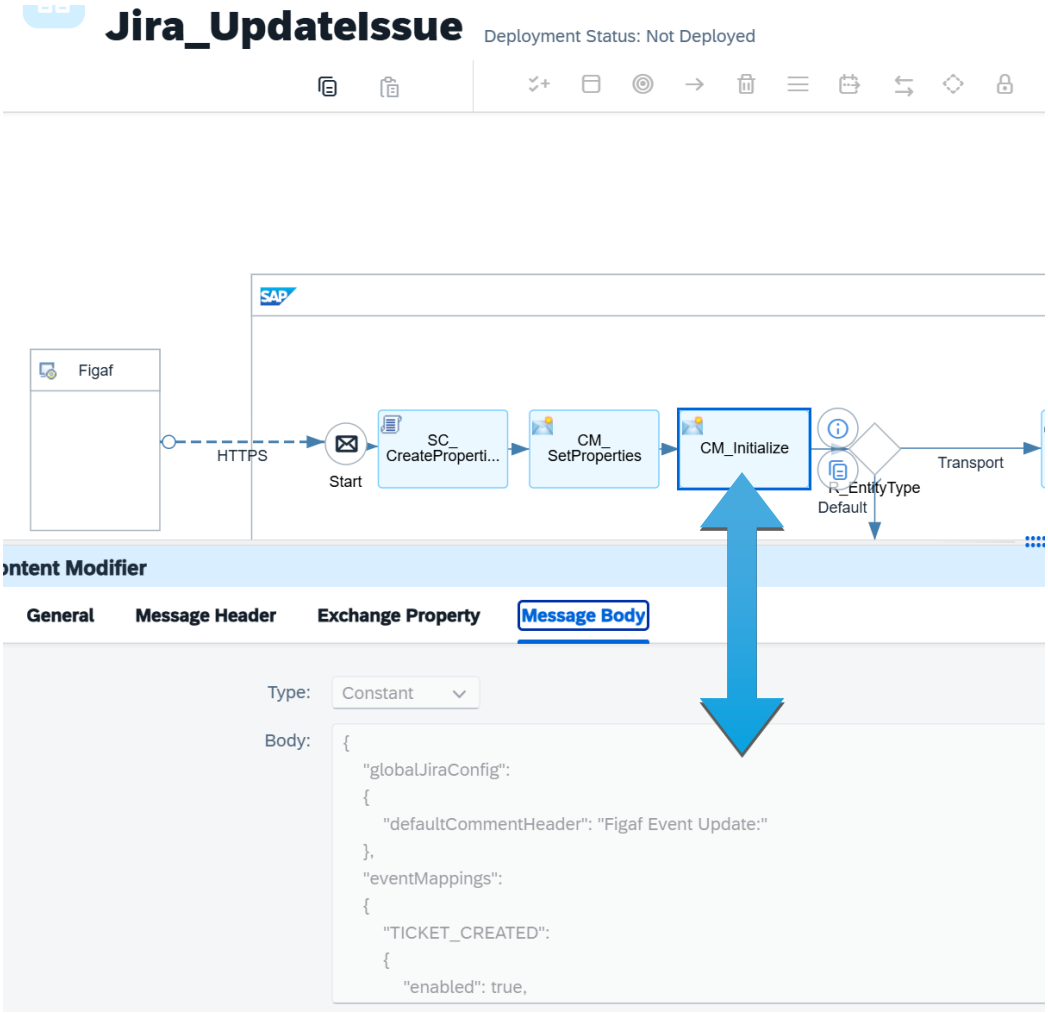
More:



- **Figaf Base URL :** This is the path to your Figaf instance.
- **Jira Base URL:** address of your Jira instance.
- **Jira Token Credential Name:** Client credentials – user and key for Jira API.

3.2.2. Options:

The primary configuration for this iFlow is managed through a JSON structure defined within the *CM_Initialize* Content Modifier step, specifically on its *Message Body* tab. This JSON dictates how different Figaf events are mapped to Jira actions.



Key Configuration Sections within the JSON:

- **globalJiraConfig**:
  - defaultCommentHeader: An optional prefix automatically added to comments posted by this integration (e.g., "Figaf Event Update:").

```
"globalJiraConfig": {
    "defaultCommentHeader": "Figaf Event Update:"
}
```

eventMappings: This is the core section where you define actions for each Figaf eventType. For each event (e.g., TICKET_CREATED, TRANSPORT_APPROVED):

- enabled: (boolean) Set to true to process this event type, false to ignore it.
- actionType: Defines the primary Jira action. Only allowed values:
  - commentOnly: Only adds a comment to the related Jira issue.
  - transitionAndComment: Performs a Jira workflow transition and then adds a comment.

- transitionDetails: (Object, used if actionType involves a transition)
    - id: The numerical ID of the Jira workflow transition to be executed. This ID must be found from your specific Jira project's workflow configuration.
    - fields: (Object, optional) Any fields that need to be set during the transition (e.g., resolution).

```
"fields": {
    "resolution": { "name": "Done" }
}
```

- comment: (String) The template for the comment to be added to the Jira issue. Placeholders in the format [[placeholderName]] will be replaced with values from the Figaf webhook payload.
    - Example: "Transport [[technicalTransportId]] for Figaf ticket [[webhookTicketDtoList[0].technicalTicketId]] is now [[status]]."

This is an example of configuration for the `TRANSPORT_APPROVED` Figaf event.

```
"eventMappings": {
    "TRANSPORT_APPROVED": {
        "enabled": true,
        "actionType": " transitionAndComment",
        "transitionDetails": { "id": "21", "fields": {} },
        "comment": "Transport [[technicalTransportId]] for Figaf ticket [[webhook-
TicketDtoList[0].technicalTicketId]] on landscape [[landscape.name]] is now [[sta-
tus]]."
    }
}
```

Explanation:

`"enabled": true`

The TRANSPORT_APPROVED event will be handled.

`"actionType": " transitionAndComment",`

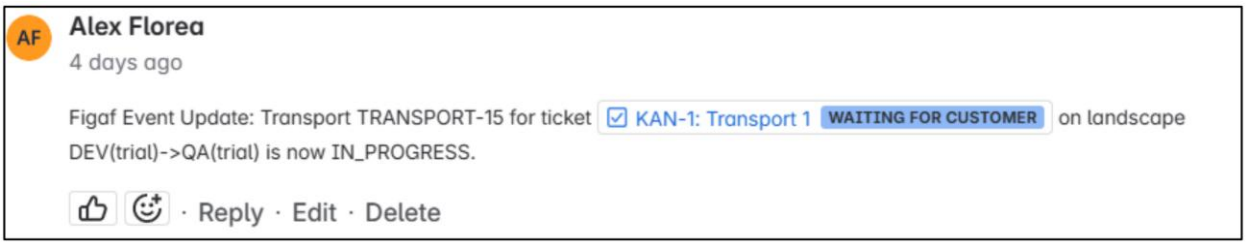
The event will produce a transition and a comment for the corresponding Jira issues.

`"transitionDetails": { "id": "21", "fields": {} },`

The new status of the ticket will be that of ID “21”, without any additional fields.

`"comment": "Transport [[technicalTransportId]] for Figaf ticket [[webhook-TicketDtoList[0].technicalTicketId]] on landscape [[landscape.name]] is now [[status]]."`

This configuration will generate a comment looking like this:



3.2.3. Using the IFlow

- **Trigger:** HTTPS call from Figaf Webhook.
- **Input:** JSON - This iFlow expects to be called by Figaf webhooks and receive a JSON file.
- **Output:** JSON - Calls to Jira API containing data about transitions and/or comments.

A list of suggested placeholders for comments:

- [[technicalTransportId]]
- [[status]]
- [[landscape.name]]
- [[webhookTicketDtoList[0].externalTicketId]] (for the Jira key if present in the Figaf payload)
- [[webhookTicketDtoList[0].technicalTicketId]] (for the Figaf ticket ID)
- [[figafEntityId]]
- [[eventType]]

3.3. Update Figaf Transport from Jira

This IFlow provides a way to automatically import a transport in Figaf via a Jira webhook call (for example, moving an issue to “resolved” status).

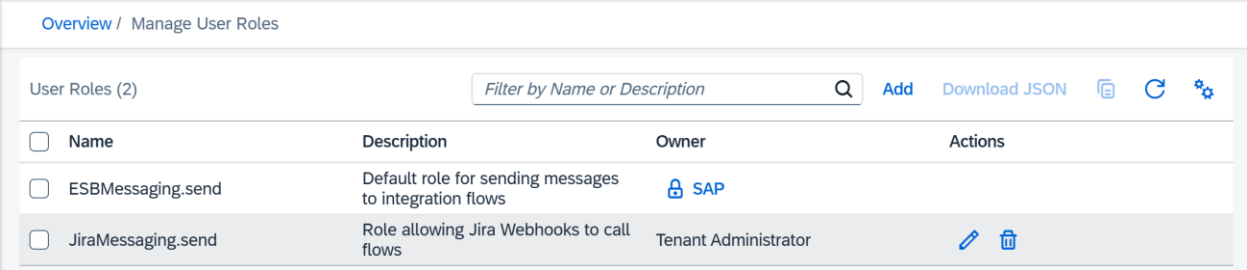
3.3.1. Setup

We need to create a new user role in Integration Suite, which we’ll later use to create an SAP Process Integration Runtime in BTP Cockpit, and create a Service Key. Then we will create a KVM with the service key and the Jira Secret. Finally, we will import the FigafJiraProxi_1 (from the iflow’s *documents* tab, into configure API’s).

These actions are required because Jira’s Webhooks only security consists of a “Secret” field. If we do not do this setup, Integration Suite will block all Jira’s calls before reaching the IFlow.

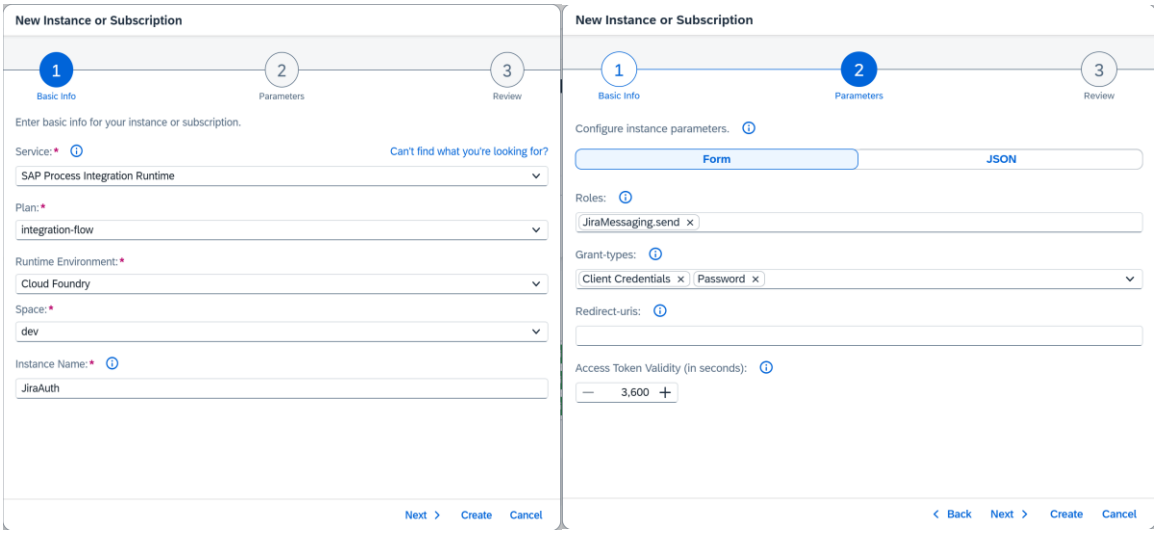
3.3.1.1. Create User Role

In Integration Suite, go to Monitor -> Integrations and API's -> Manage Security -> User Roles. Create a new User Role called JiraMessaging.send**.**



3.3.1.2. Create SAP Process Integration Runtime service

In BTP Cockpit, go to Services -> Instances and Subscriptions -> Create -> SAP Process Integration Runtime.

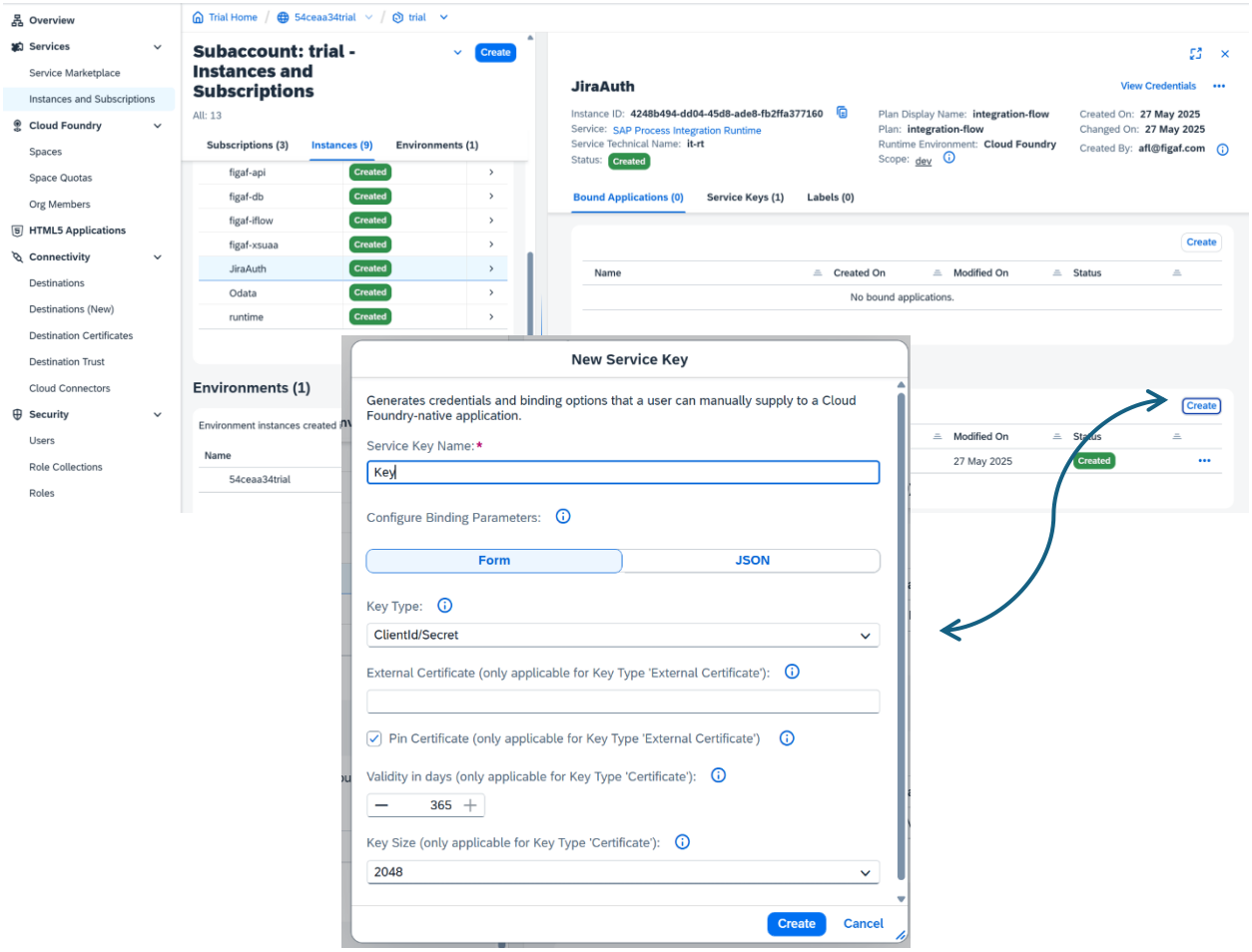


Configure the Parameters by hand or paste the JSON below in the JSON tab, then click Create:

```
{
    "roles": [
        "JiraMessaging.send"
    ],
    "grant-types": [
        "client_credentials",
        "password"
    ],
    "redirect-uris": [],
    "token-validity": 3600
}
```

3.3.1.3. Create Service Key

In Instances and Subscriptions, select the just created JiraAuth runtime process, and under Service Keys, create a new key.



Then select ••• and View for the created key and copy the “clientid” and “clientsecret” values. We will use these in the Key Value Map in the next step.

3.3.1.4. Create Key Value Map

Go to Integration Suite -> Configure -> APIs -> Key Value Maps -> Create. Create a new KVM called **FigafJiraUser**. Add the following keys:

Key	Value
protectuser	Value of clientid from JiraAuth Key
protectpass	Value of clientsecret from JiraAuth Key
secret	Jira secret (add later -> check step 3.3.1.5)

IMPORTANT: Check the Encrypt Key Value Map checkbox to be ticked.

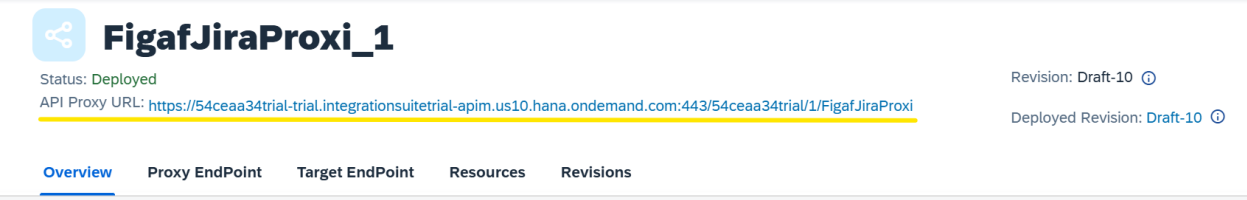
3.3.1.5. Create and Deploy API Proxi

We now need to deploy a proxi which will handle the authorizations of the Jira webhooks calls. Go to the Figaf Jira Integration package -> Documents. Download the **FigafJiraProxi** file. Then, go to Configure -> APIs -> API Proxies -> Import API -> select the **FigafJiraProxi** downloaded file.

Select the imported FigafJiraProxi -> select Edit -> Target EndPoint.

In the URL field, put in the endpoint of the Update Figaf Transport from Jira IFlow (The IFlow needs to be deployed – The endpoint can be found under Monitor -> Integrations and APIs -> Manage Integration Content -> All -> Update Figaf Transport from Jira -> Endpoints). Save and **Deploy** the Proxy.

You can view the API Proxi URL at the top, under the Status field:

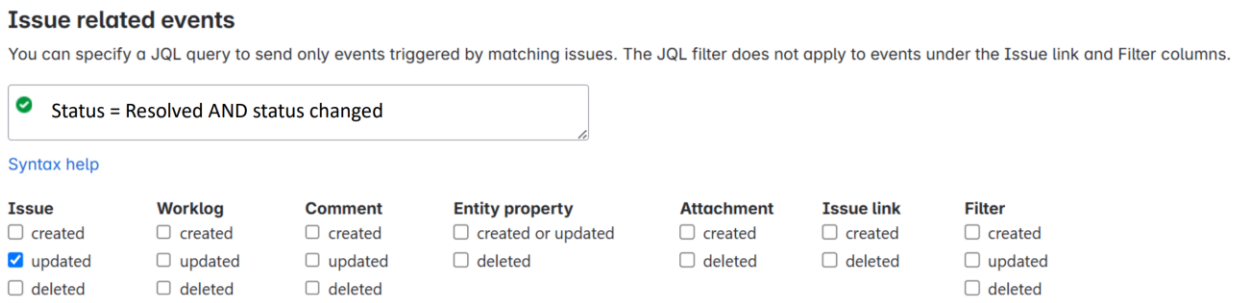


3.3.1.6. Create Jira Webhook

We now need to create a Jira webhook that will call the API Proxi, which will then call the IFlow. Go to Jira -> Settings ->System -> Advanced -> WebHooks -> Create a Webhook .

Give it a suggestive name (i.e. TransportApproved). Under URL, put the API Proxi URL of **FigafJiraProxi.**

Then, create a **Secret**. Copy this and paste it in the FigafJiraUser Key Value Map, under “secret” field value.
Now we need to configure the webhook to be triggered when we desire. An example of configuration is the following:

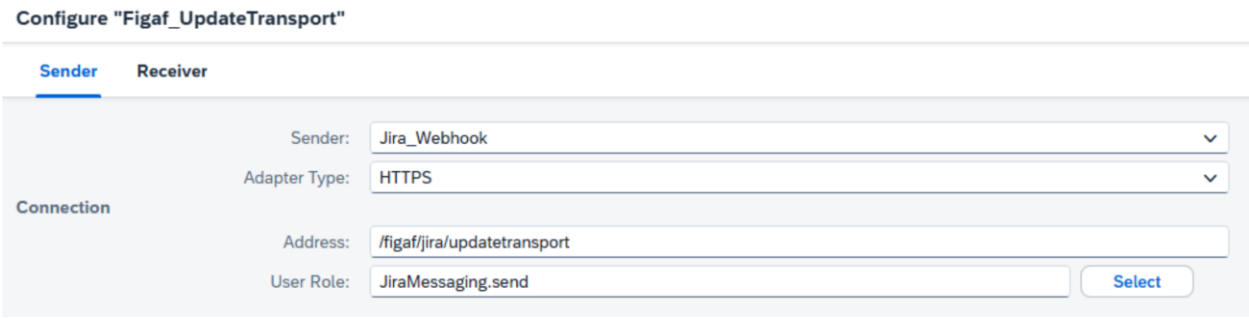


This will trigger the webhook whenever an issue is set as “Resolved”.

If set up correctly, it will further trigger the API Proxy and the Update Figaf Transport from Jira IFlow, which will call the Figaf API, requesting the transport related to the Jira issue to be imported.

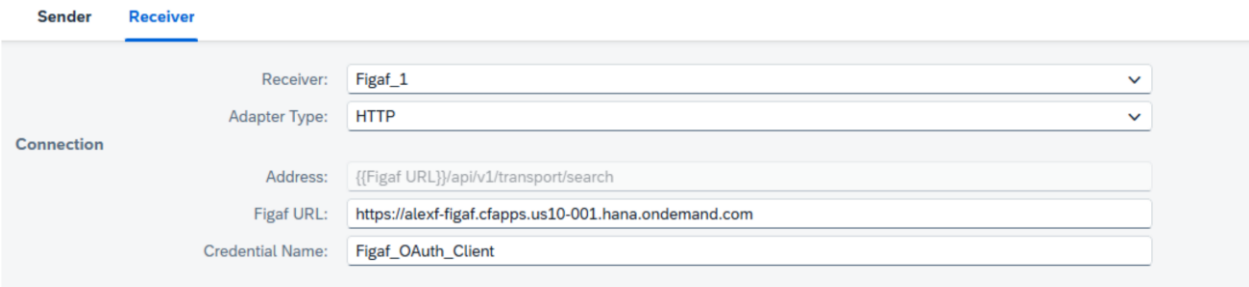
3.3.2. Configuration:

Sender Details:



- **Address:** This is the path appended to your Integration Suite tenant URL to call this iFlow.
- **User Role:** Will use a custom user role for authentication (JiraMessaging.send in this case).

Receiver Details (Connection to Jira):

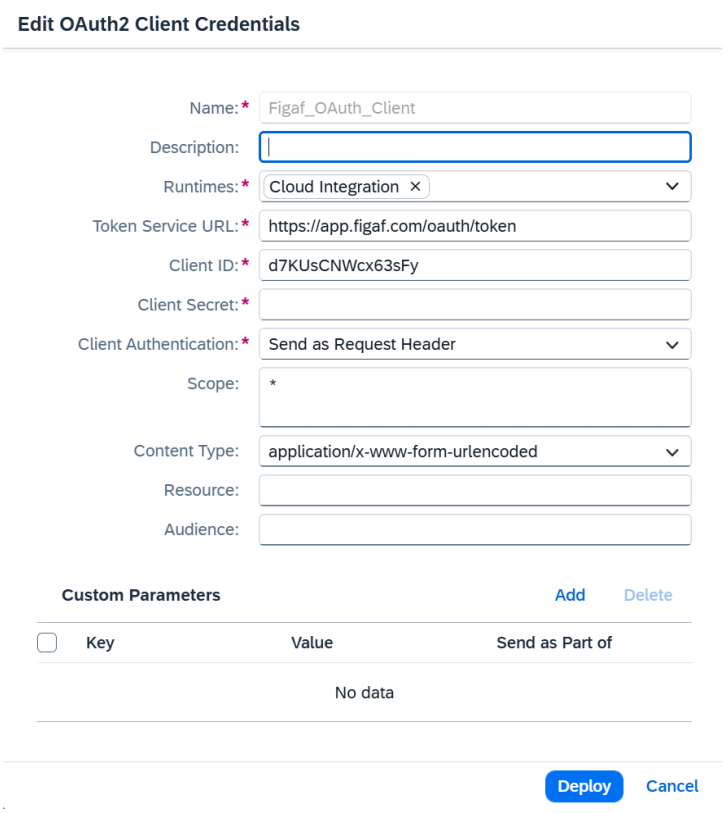


- **Figaf URL**: The base Figaf URL. Will be used to build the address to call the API.
- **Credential Name**: The alias for storing the Figaf OAuth Client Credentials.

To create the Figaf OAuth Client, in Figaf go to Configuration -> OAuth Clients -> Add OAuth Client. Select the following scopes:

ticket:read, ticket:run, ticket:import, ticket:resolve, transport:read.

Then add the credentials in Integration Suite like below. Token Service Url is your Figaf url followed by /oauth/token



4. Deployment and General Setup

1. **Deploy Credentials:**
    - Ensure the User Credential artifact containing a valid Jira API token is deployed on your Cloud Integration tenant and that the iFlows configurations reflect the same alias.
    - Ensure the User Credential containing Figaf OAuth Client certificate is deployed.
2. **Configure Update Jira Issues iFlow:**
    - Access the deployed iFlow.
    - Modify the CM_Initialize Content Modifier's Message Body with your specific Jira URL, credential name, and the desired eventMappings logic (transition IDs, comment templates, etc.) as detailed in section 3.2.
3. **Configure Search Jira Issues:**
    - Access the deployed iFlow and go to the "Configure" view.
    - Update the Receiver Address to your Jira instance URL.
    - Modify the Receiver Query parameter to match your desired JQL (project, fields, etc.).
4. **Configure Update Figaf Transport from Jira:**
    - Follow the Setup Steps from sections 3.3.1.
    - Deploy the Proxy and create a KVM.
    - Configure the Figaf URL and the Credential Name.
5. **Deploy Integration Package:** Deploy this integration package ("Figaf and Jira Integration Suite") to your tenant.
6. **Configure Figaf Webhooks:**
    - In Figaf, set up a new integration and then the webhooks to point to the endpoint URL of the API Proxy.