



IssueSYNC

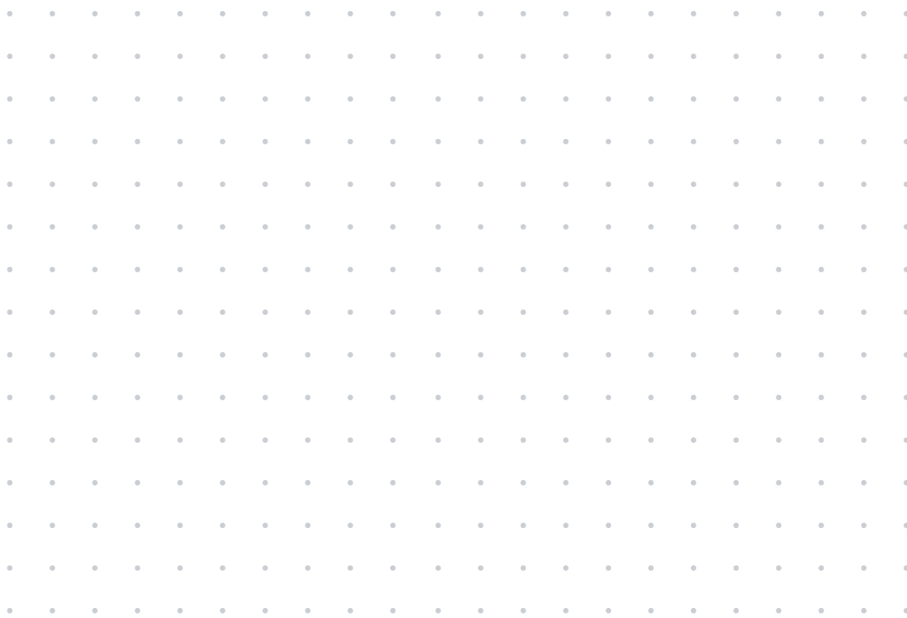
Server 101

DEVINITI

Technology-Driven Results

Contents

- Introduction
- Prerequisites
- Use case scenario
- Jira projects configuration
- How can Issue Sync help?
- What do you need?
- Modify workflows
- Connect Jira internally via IssueSync
- Synchronization Schemes
- Create Synchronization Schemes
- Configuring
- Synchronization Scheme contexts
- Mapping Workflow transitions
- Mapping Fields
- Test run of the IssueSync configuration





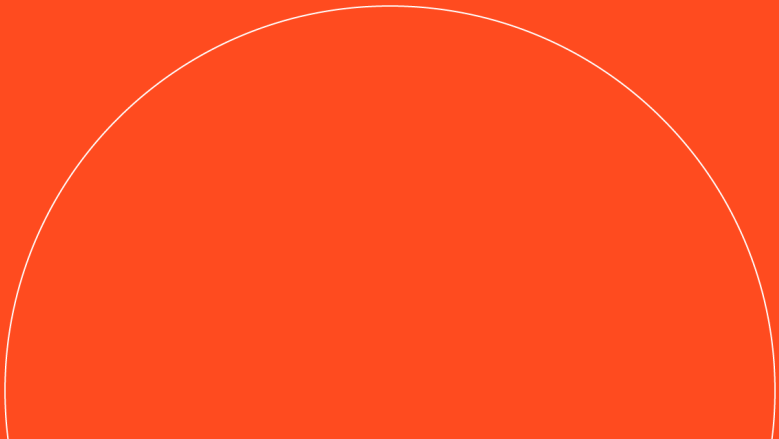
Introduction

This is the introductory tutorial for IssueSYNC new users, who would like to understand this app simply by using it in a practical step-by-step scenario. Instead of reading the full documentation and trying to figure out how to make IssueSYNC work for you, simply follow this use case scenario on your Jira sandbox instance and start learning by doing. When you understand the basics, then reading the full documentation will be much easier.



Prerequisites

In order to follow this tutorial, you need to have some familiarity with the basics of Jira Administration. You need to know how to modify default workflows and create additional workflow transitions.

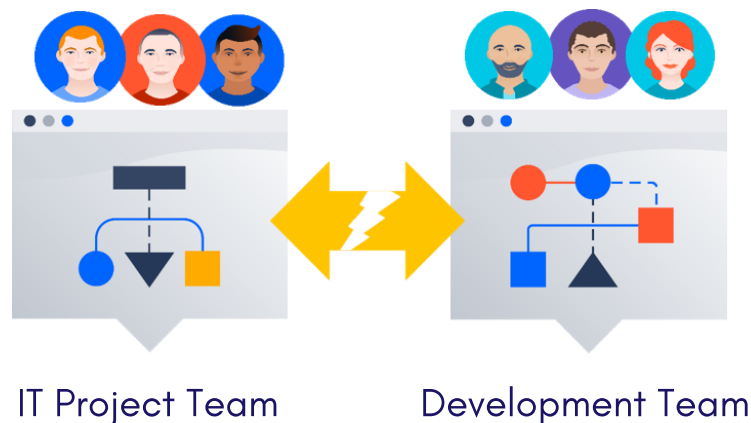


Use case scenario

There are two teams collaborating during a project. The IT Project Team is responsible for implementing an IT system and they organized their project work using Jira.

Work items (Epics): Every work item is decomposed into User Stories and Tasks.

User stories: Most User Stories and Tasks are delivered by the IT Project Team internally, but some User Stories need to be completed by other teams, which specialize in other technologies.

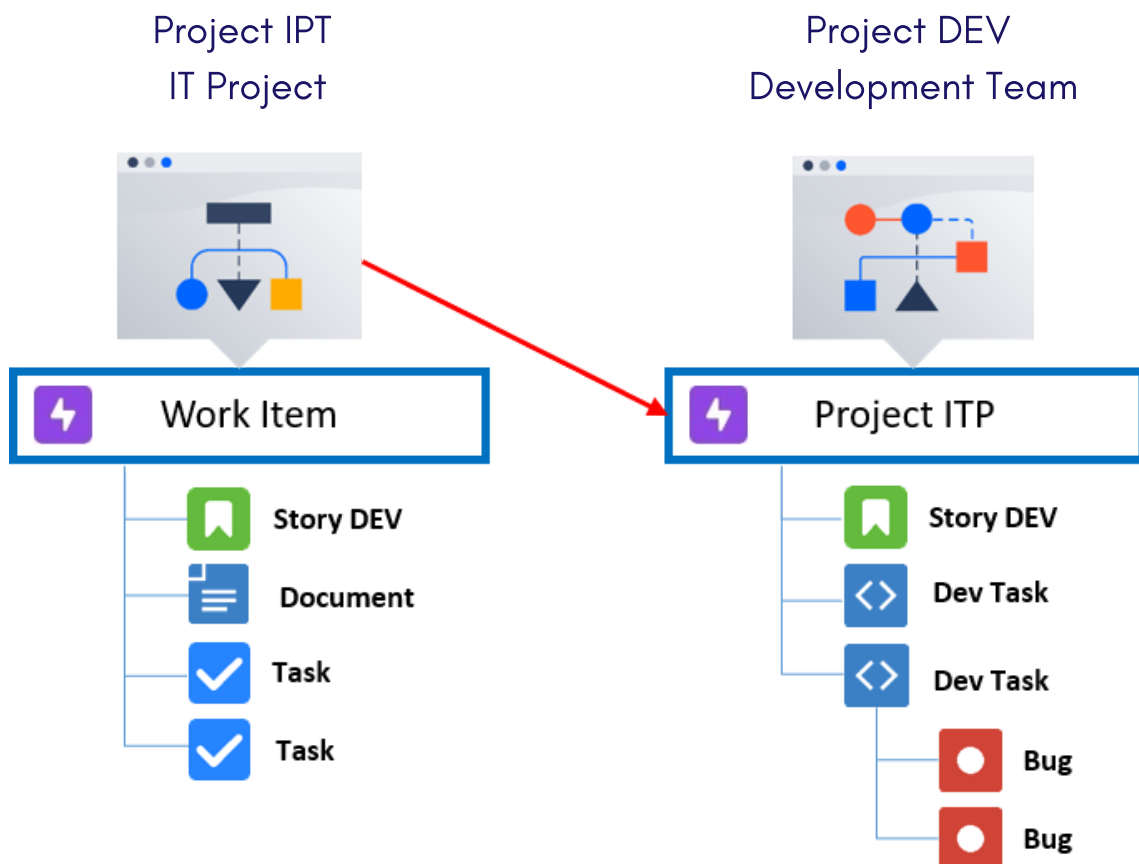


The problem: the Development Team made it clear that they work with many IT projects and they want to organize their work using only their Development Team Jira project (DEV). They reject tracking their tasks in other Jira projects. Also, they refuse giving access external users to their DEV Jira project. They want collaborate on the User Story level, report status and exchange comments but using only their Jira project.

Jira projects configuration

The IT Project Team decided to use default Jira configuration. They created their project using the Scrum Software Development template. They agreed that the Epics will be their work items. Every work item can only be transitioned to status Done when there are no unresolved issues in it.

The Development Team works in a different way. Because they work with many projects, they decided that they will use Epics to represent project teams that they work with. They use Epics in their Jira project to group all the stories and the development work that they deliver for individual projects.





How can Issue Sync help?

We will use Issue Sync to synchronize the Story issues data between two Jira projects. In this way the IT Project Team (ITP) will be able to check their Story issues delivery status and communicate with the Development Team members without having the browse permission in the Development Team project.

This solution will make the Development Team happy, because they will be able to maintain the work tracking approach that they currently have, keep their Jira project secured and they will not be forced to update issue statuses and communicate in comments in other Jira projects.

What do you need?

In order to create this solution you will need the following configuration items:

1. Jira sandbox server (or your testing server)
2. IssueSync app installed and activated
3. Two projects created with the Scrum software development project template
 - a. Project ITP – IT Project
 - b. Project DEV – Development Team Project
4. Modified workflows for the Story issue type
5. Additional technical user account (named for example Sync User). Note the password of this user because you will need it soon.

NB: To keep this exercise simple, you do not need to change anything in the default issue type scheme and in the default workflow scheme. You will only need to slightly modify the workflows.

Scrum software development

Use this project to manage your Agile development work. Create a backlog, organise work into sprints, check progress using reports, and more. This project includes a Scrum board, a basic Agile workflow and issue type configuration, which you can change later on.

Issue Types

- Bug
- Task
- Sub-task
- Story
- Epic

Workflow

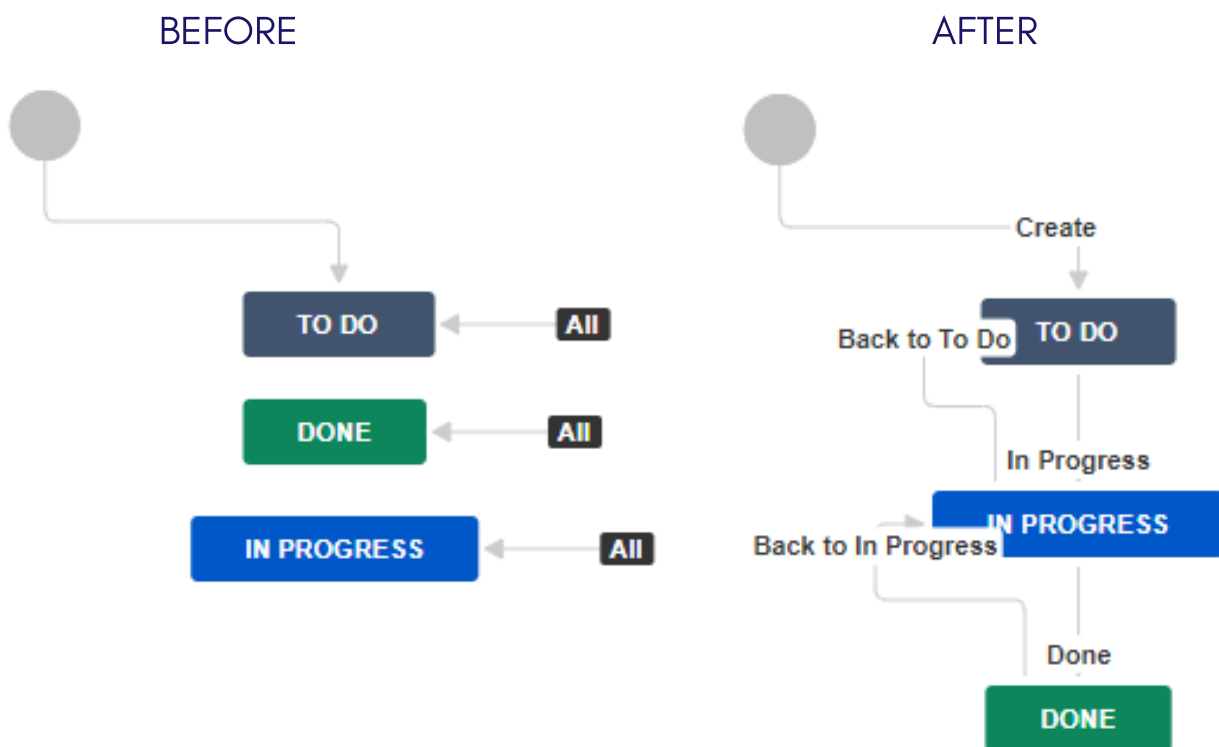


Back Select Cancel

Modify workflows

If you have your new projects created, now it is a good time to modify the default. In the default Software Simplified Workflow for Project, all you need to do is to remove the global transitions and replace them with the single transitions.

Go to your project and modify the default workflow as follows:



NB: Make sure that in both Jira projects ITP and DEV, you have the same but separate workflows in the respective workflow schemes.

Now you have configured everything you need in Jira to connect and sync two projects together with the IssueSync.



Connect Jira internally via IssueSync

In this step you will create an internal link in you Jira server. You will use this link to declare issue syncing between two Jira projects.

Go to **Jira Administration > Manage apps > Menu ISSUESYNC > Configuration**

Click on tab Connection

The screenshot shows the 'Connection' tab selected in the Jira IssueSync configuration interface. The page title is 'Choose connection type'. There are three options:

- Local**: Synchronize issues within this JIRA instance. (This option is highlighted with a red border in the image.)
- Passive server**: Choose this type if you can not access remote JIRA instance (e.g. remote server has firewall). This JIRA server has to be accessible from remote server. Issue changes will be downloaded from this JIRA instance.
- Server**: Synchronize issues with another JIRA Server instance. Issue changes will be automatically send. Choose this type also if remote JIRA instance is passive.
- Cloud**: Synchronize issues with JIRA Cloud instance.



Create the new local connection. Use the technical user and its password that you have created before.

New local connections

Local

First connection name* Local 1

Second connection name* Remote 2

Technical User

JIRA user* Sync User 3

Password* 4

Test User 5

Create local connections

1. Local connection name – Local
2. Remote connection name – Remote
3. Technical Jira user name login – Sync User
4. Technical Jira user name password
5. Press the button Test User to check the password
6. Press the button Create local connections

See the screen with the connections configured

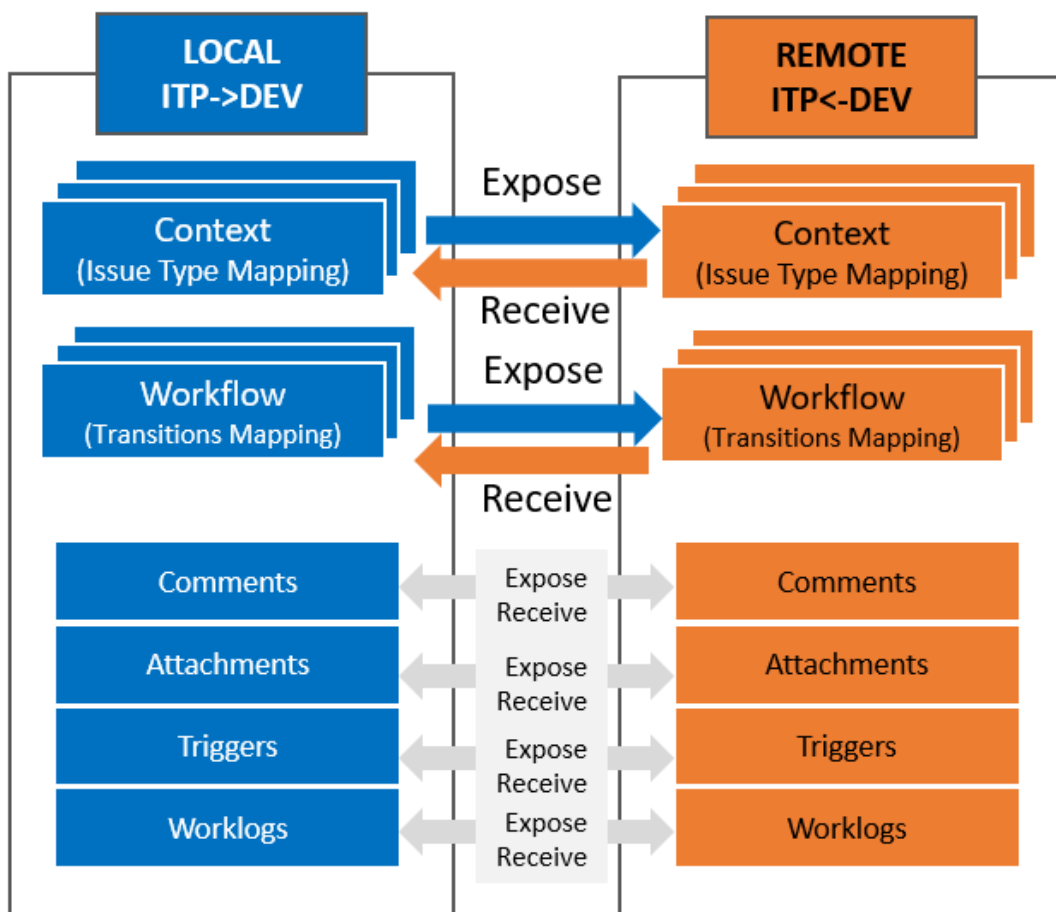
Type	Name	remote URL	User	Actions
Local	Local	http://sync@sync.jira.com:8080	Sync User	[Edit] [Refresh] [Delete]
Remote	Remote	http://sync@sync.jira.com:8080	Sync User	[Edit] [Refresh] [Delete]

Synchronization Schemes

Before you create your first Synchronization Scheme, you need to understand what elements IssueSync is actually syncing.

What is a Synchronization Scheme?

Synchronization Scheme is a sort of container for a couple of smaller schemes and configurations which make it technically possible to keep in sync elements of Jira issues. With IssueSync you can synchronize issue name, description, comments, due date, status and many others. All these objects need to be declared and configured to be synchronized. That is why you need a Synchronization Scheme to manage individual configuration items and keep them together.



What is Exposing and Receiving?

In this use case, to synchronize elements from two projects (ITP and DEV) you will need to create two Synchronization Schemes.

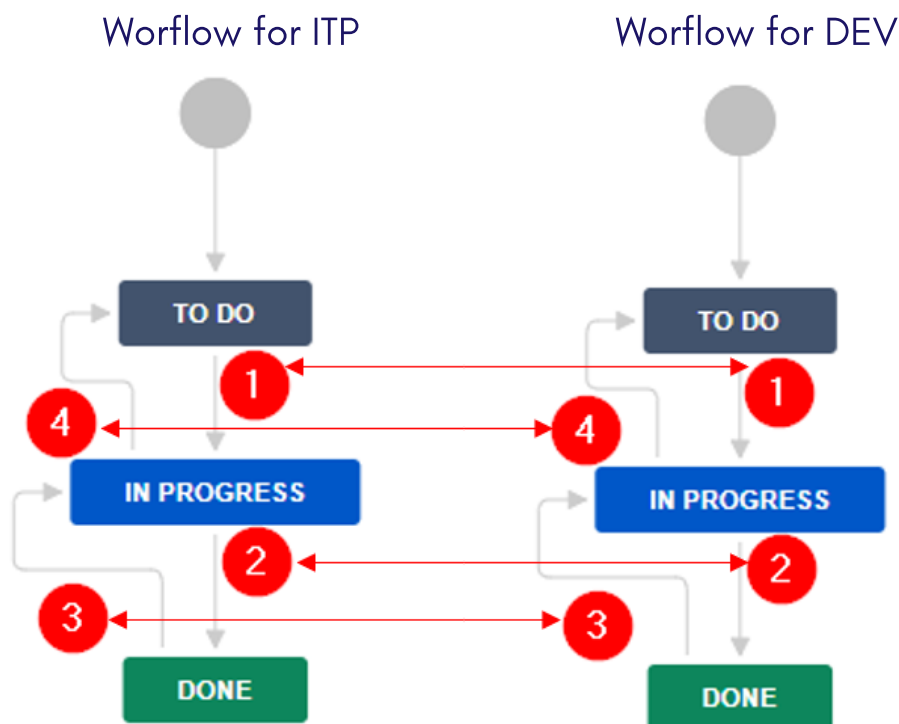
In each of two Synchronization Schemes (Local and Remote) you declare specific Jira issue items, such as issue type, workflow, comments, attachments etc. and configure how you want to expose them for the other scheme to make use of it.

If you declared (in both schemes) what you want to expose, it means that both schemes can now “see” each other. Now, also in both Synchronization Schemes, you declare how you want to receive and map the data that has been exposed by the other scheme. In this way you make pairs of what is exposed and received by each of the schemes. This will be much more clear for you after you complete the exercises below.

It is also very important to understand, that if you “Expose” some configuration item in one Scheme, you make it visible to the other scheme. However, making this configuration item visible does not mean that the other Scheme has accepted it and connected to it. You also need to declare this connection by setting up the “Receive” configuration. When you do this, you will be able to close the loop. Now, if you want to this loop to work in two directions, then you need to make this Expose<->Receive configuration in both Synchronization Schemes.

Synchronization Scheme configuration items

- 1. Context - issue type mapping:** in this configuration item you will declare which issue type you want to include in the scope of synchronization. In your Jira project you can have many issue types and may decide that not all need to be exposed to be synchronized with the other Jira project.
- 2. Workflow - transition mapping:** it is important to know that in IssueSync you are not syncing issue statuses but their transitions. For example, if you transition an issue from 'To Do' status to 'In Progress' in a Local project, this will notify the connected issue and its workflow in the Remote project and the mapped transition will be used to change the issue status there. As a result, the Remote issue will also be transitioned from 'To Do' status to 'In Progress'.



3. The other mappings are responsible for the remaining Jira issue elements:
 - a. fields mapping: which fields should exchange information
 - b. attachment: enabling attachments exchange
 - c. worklogs: enabling work log entries exchange
 - d. comments: which comments should be exchanged
 - e. triggers: which Jira events will be used to send information to the other scheme about creating, updating and disconnecting synchronization

What is the result of using Synchronization Schemes in this use case?

If you create a Story Issue in the ITP project, then the same Story Issue will be created in the DEV project. If someone from the DEV team makes a comment or changes something in this Story Issue in their DEV project, this change will be automatically reflected in the ITP project Story Issue.

To give you another example - if something changes in a Story Issue of the ITP project, then a corresponding Story Issue in the DEV project will also change. And if something changes in a Story Issue in the DEV project, then the associated Story issue in the ITP project will be updated. In this way you will utilize bi-directional synchronization.

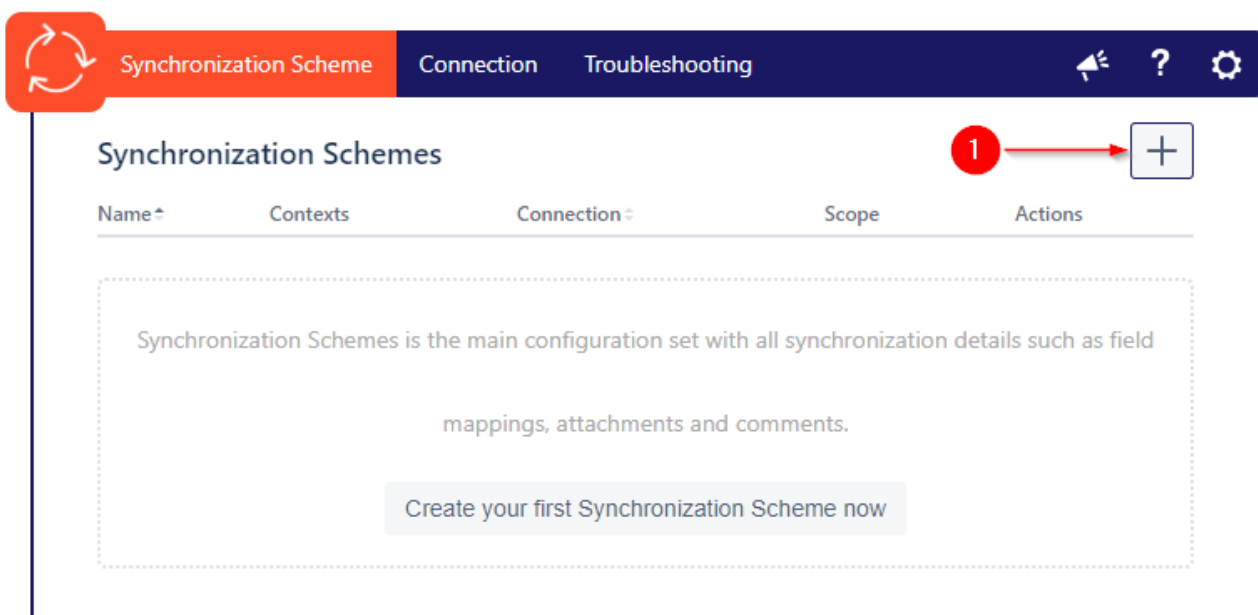
To summarize, this is what the Synchronization Scheme is all about. It is a set of mappings of Jira issue elements that you want to keep in sync in two separate projects in the same or in remote Jira instances.

Create Synchronization Schemes

Let's begin with creating two synchronization schemes – one for each direction.

Go to **Jira Administration > Manage apps > Menu ISSUESYNC > Configuration**

Click on tab Synchronization Scheme and then the button Plus



Fill in the form and name your Synchronization Scheme. Use the naming convention which will help you navigate while configuring other items later. I will refer to the scheme as Local because I treat the ITP project in this exercise as Local, from which I will be sending data to the DEV project, which for me is Remote.

Synchronization Scheme

Name

Synchronization Scheme

Name * DEV->ITP

Connection * Remote (http://...)

Create

Now create the remote Synchronization Scheme. Press the Synchronization Scheme button and then the Plus sign again.

Synchronization Scheme

Synchronization Schemes

1

2

Name	Contexts	Connection	Scope	Actions
ITP->DEV		Local (http://...)	Global	[Edit] [Delete] [Copy]

Now, use the name DEV->ITP so you will know in which direction the Synchronization scheme will work. Also, use the Remote connection. As previously stated, this Scheme will be referred to as Remote.

Synchronization Scheme

Name

Synchronization Scheme

Name * DEV->ITP

Connection * Remote (http://...)

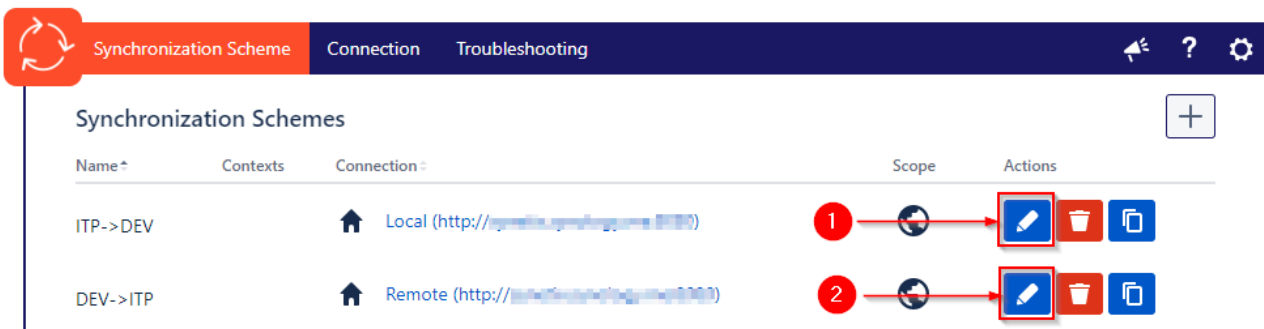
Create

You have just created two synchronization schemes, where you will now configure and link ITP and DEV projects together.

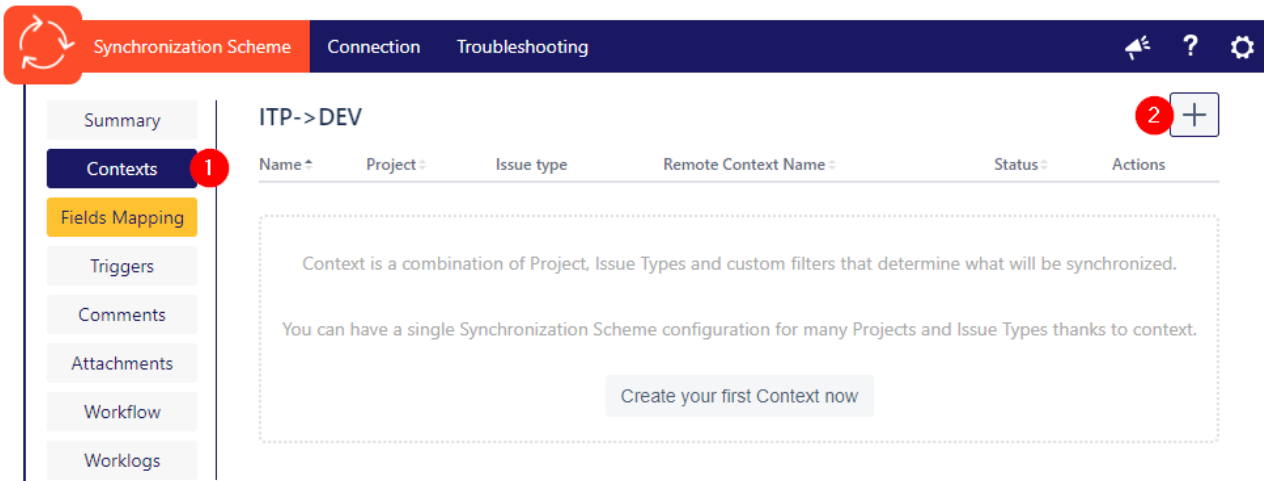
Configuring Synchronization Scheme contexts

As you know, the context is used for issue type mapping. Here you will declare which issue type you want to include in the scope of synchronization in both Synchronization Schemes. Then, you will bind both contexts to each other.

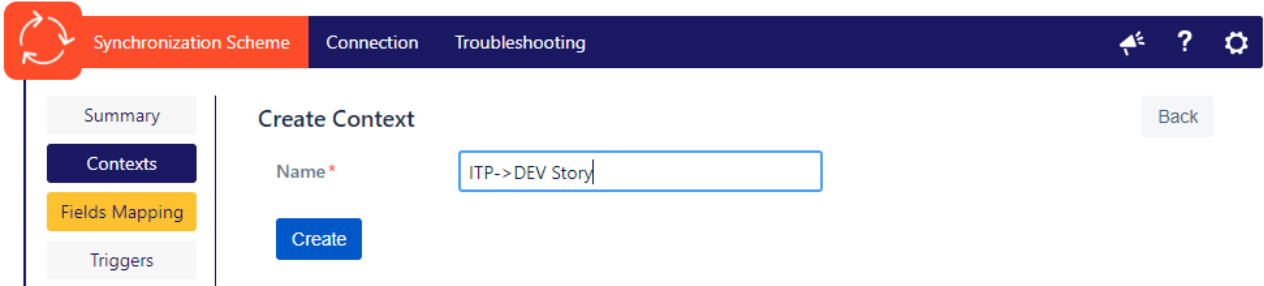
Press the pencil button of the first Synchronization Scheme (1)



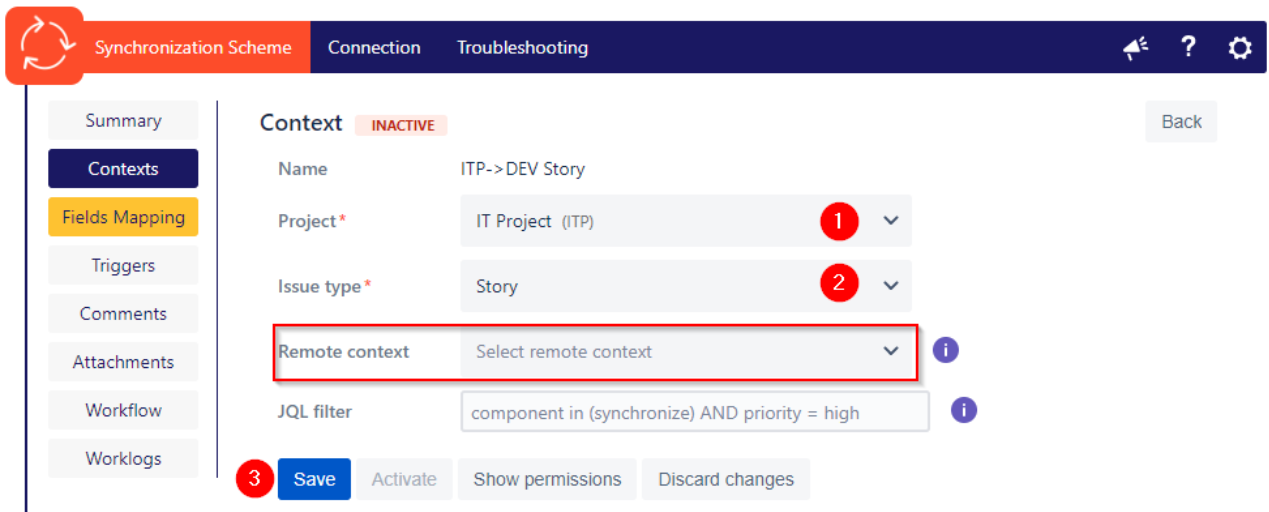
Inside the Synchronization Scheme, press the button Contexts (1) and then the plus button (2)



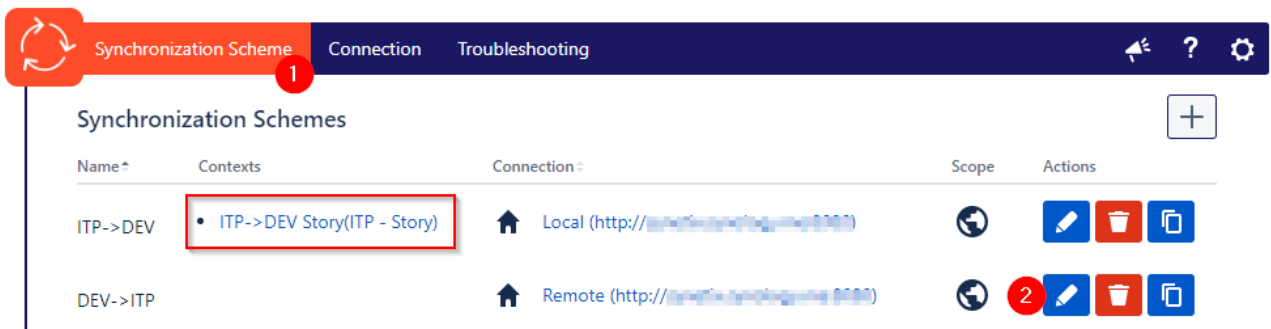
Give your context a name. It is best to use some naming convention again.



When you press the Create button, you will be able to configure the Context.



Select the Local project (1), the issue type (2) and press Save (3). You will not be able the Remote context because there is not any other created yet. You will need to come back to the Synchronization scheme and select the Remote context when you create it in the next step. For now just press the Save button.



Press the Synchronization Scheme tab and the pencil button for the Remote Synchronization Scheme. Notice that the first context is now made visible and should be also visible in the Remote Scheme.

Just like before, inside the Remote Scheme press the Contexts button and press the plus button. Use the same naming convention and name the Context as DEV -> ITP Story, because you declare now the context in the opposite direction.

The screenshot shows the 'Synchronization Scheme' interface. The top navigation bar includes 'Synchronization Scheme', 'Connection', and 'Troubleshooting'. The left sidebar has a 'Contexts' button highlighted with a red circle '1'. The main form is titled 'Context INACTIVE' and contains the following fields:

- Name: DEV->ITP Story
- Project*: Development Project (DEV) (dropdown, red circle '2')
- Issue type*: Story (dropdown, red circle '3')
- Remote context: ITP->DEV Story (dropdown, red circle '4')
- JQL filter: component in (synchronize) AND priority = high (with an info icon, red circle '6')

At the bottom, there are 'Save' (red circle '5') and 'Activate' (red circle '6') buttons, along with 'Show permissions' and 'Discard changes' buttons. A 'Back' button is in the top right corner.

1. Press the button Contexts
2. Select Project DEV
3. Select issue type: Story
4. Select Remote context - now it is possible because you have just created it in the previous step
5. Save the context
6. Press the button Active. Only active Contexts can be used by IssueSync.

As soon as you create and activate the Context, on the Dev issue screen there will be a new section visible. This means that the context is active and ready to be used.

Development Project / DEV-1

Story Issue to be synchronised

1 of 1

Edit Comment Assign More Back to In Progress Admin Export

Details

Type: Story Status: **DONE** (View Workflow)

Priority: Low Resolution: Resolved

Labels: None

People

Assignee: Unassigned
Assign to me

Reporter: Sync User

Watchers: 1 Stop watching this issue

Description

Click to add description

Attachments

Drop files to attach, or browse.

Dates

Created: Yesterday

Updated: 3 minutes ago

Resolved: Yesterday

Activity

All Comments Work Log History Activity Synchronization History

There are no comments yet on this issue.

Remote Issue

Status: NOT CREATED

Action: Create Remote

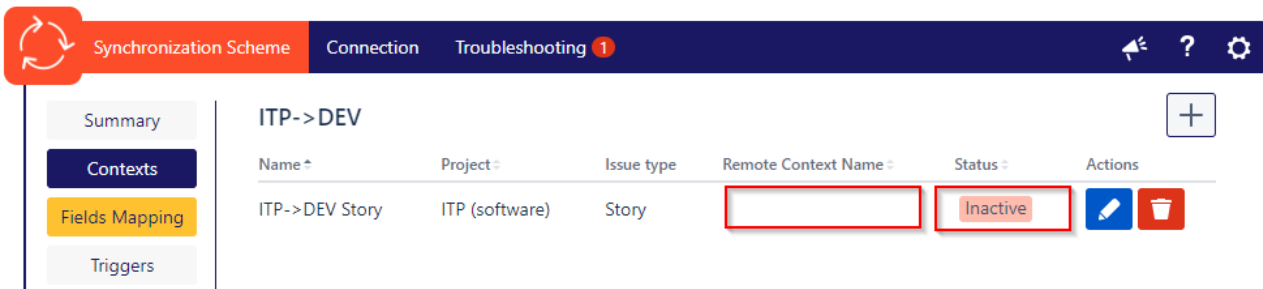
Now go back to the Local Synchronization Scheme and select the remote context and activate it there.

Synchronization Scheme Connection Troubleshooting 1

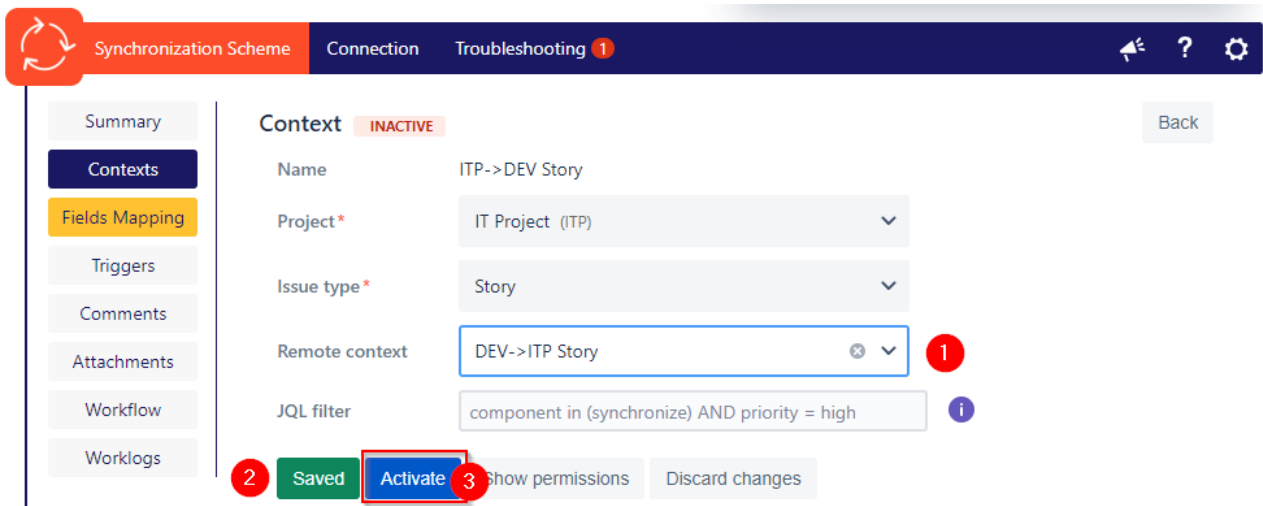
Synchronization Schemes

Name	Contexts	Connection	Scope	Actions
ITP->DEV	• ITP->DEV Story(ITP - Story)	Local (http://synetix.synology.me:8080)	1	
DEV->ITP	• DEV->ITP Story(DEV - Story)	Remote (http://synetix.synology.me:8080)		

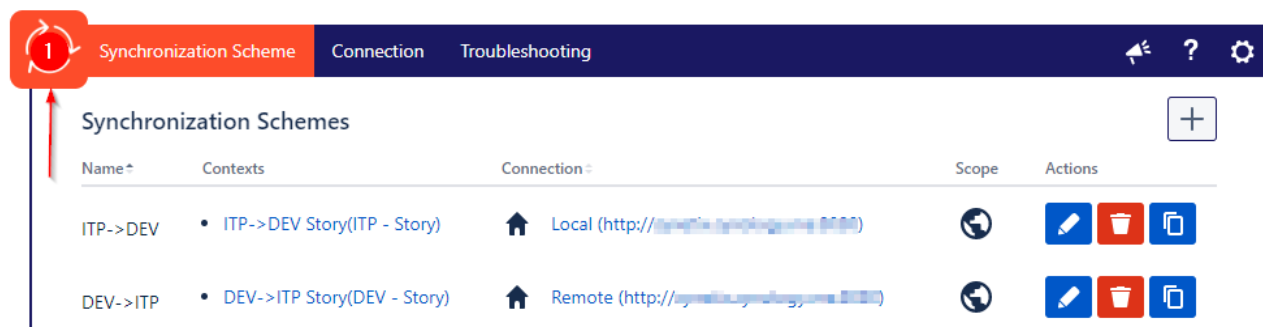
Press the button Contexts and see that the remote context name is empty and the status of this Context is still inactive. You need to remember that even if you activate the Scheme in one direction, it will not be automatically activated in the opposite direction. You still need to do it manually. Press the pencil button to edit the Context.



Press the pencil button to edit the Context configuration.



1. Declare the Remote context
2. Save the current context
3. Activate the context to finish the bi-directional synchronization

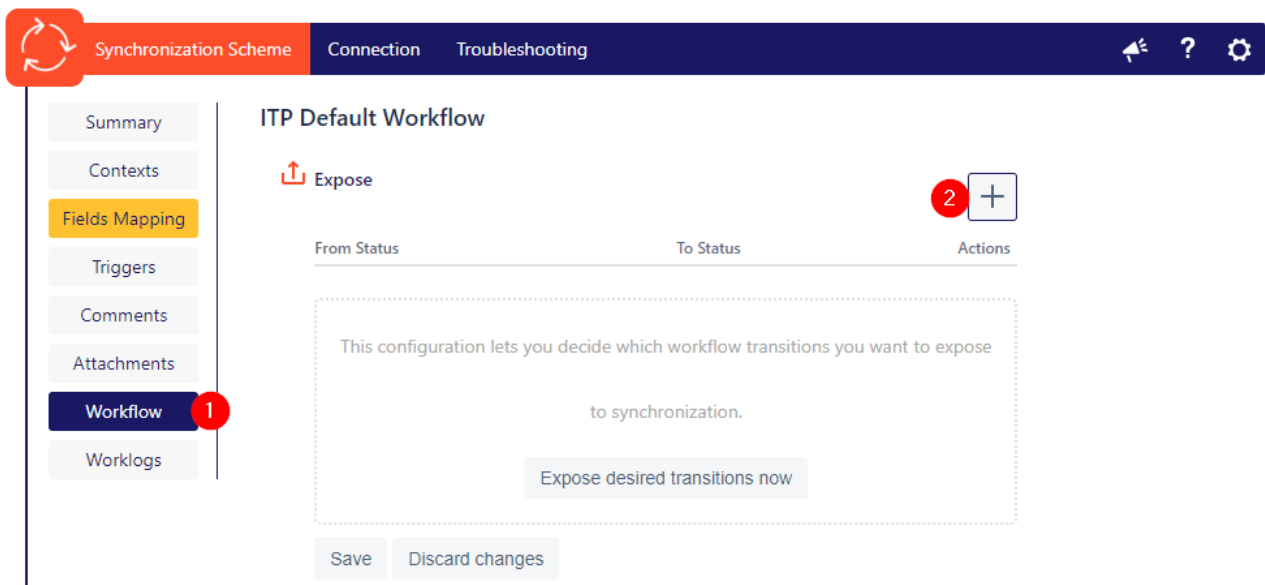


When you press the Synchronization Scheme tab and the Synchronize button (1) your Context configuration is complete.

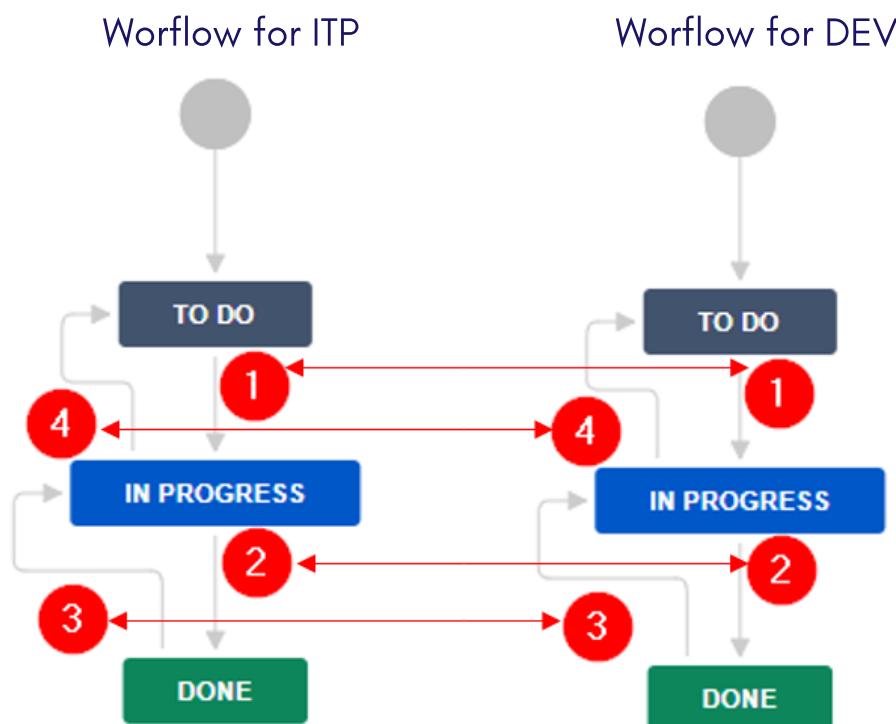
Now it is time to configure Workflow synchronization.

Mapping Workflow transitions

Now, similarly as before, go to the Workflow section of the Synchronization Scheme and press the plus button. You will now declare these workflow transitions, which you want to have exposed and make visible to the Remote Synchronization Scheme.



Recall how our workflow is designed. In this configuration you will be creating these red arrows between your local ITP project Story workflow and the remote DEV Story workflow.



Look how easy it is to declare the transitions that you want to expose to the Remote Synchronization Scheme.

Synchronization Scheme | Connection | Troubleshooting

Summary | Contexts | **Fields Mapping** | Triggers | Comments | Attachments | **Workflow** | Worklogs

ITP Default Workflow

Expose

	From Status	To Status	Actions
1	TO DO	IN PROGRESS	🗑️
2	IN PROGRESS	DONE	🗑️
3	DONE	IN PROGRESS	🗑️
4	IN PROGRESS	TO DO	🗑️

Save | Discard changes

Similarly as before, you cannot yet map the remote transitions, because you have not declared them in the remote Synchronization Scheme. Go there now and press the pencil button to edit the Scheme and then go to the Workflow section

Synchronization Schemes

Name	Contexts	Connection	Scope	Actions
ITP->DEV	• ITP->DEV Story(ITP - Story)	Local (http://...)	🌐	✏️ 🗑️ 📄
DEV->ITP	• DEV->ITP Story(DEV - Story)	Remote (http://...)	🌐	✏️ 🗑️ 📄

Red arrow points to the 'Edit' button for the 'DEV->ITP' scheme.

Here, in the remote Synchronization Scheme, you will be able to declare the Expose section but also the Receive section. In the receive section you will see all the transitions that you have just declared in the Local Synchronization Scheme.

- Summary
- Contexts
- Fields Mapping
- Triggers
- Comments
- Attachments
- Workflow**
- Worklogs

DEV Default Workflow

Expose

From Status	To Status	Actions
TO DO	IN PROGRESS	
IN PROGRESS	DONE	
DONE	IN PROGRESS	
IN PROGRESS	TO DO	

Save Discard changes

Receive

Local From Status	Local To Status	Remote Transition	Actions
2	3	To Do -> In Pro...	

IN PROGRESS TO DO DONE

This is how the Receive section looks like, when it is fully configured.

Receive

Local From Status	Local To Status	Remote Transition	Actions
TO DO	In Progress (11) >> IN PROGRESS	To Do -> In Progress	
IN PROGRESS	Done (21) >> DONE	In Progress -> Done	
DONE	Back to In Progress (31) >> IN PROGRESS	Done -> In Progress	
IN PROGRESS	Back to To Do (41) >> TO DO	In Progress -> To Do	

Save Discard changes

Do not to forget to press the Save button in both sections. When you do this, go back to the Local Synchronization Scheme again (ITP -> DEV) and configure the Receive section there as well.

ITP Default Workflow

Expose

From Status	To Status	Actions
TO DO	IN PROGRESS	
IN PROGRESS	DONE	
DONE	IN PROGRESS	
IN PROGRESS	TO DO	

Save Discard changes

Receive

Local From Status	Local To Status	Remote Transition	Actions
TO DO	In Progress (11) >> IN PROGRESS	To Do -> In Progress	
IN PROGRESS	Done (21) >> DONE	In Progress -> Done	
DONE	Back to In Progress (31) >> IN PROGRESS	Done -> In Progress	
IN PROGRESS	Back to To Do (41) >> TO DO	In Progress -> To Do	

Save Discard changes

Now you have the second major configuration element complete. All there is to be configured are the fields of your choice.

Mapping Fields

Mapping fields is very easy after you have completed creating context and workflow mappings.

Open the local synchronization scheme and click the Fields Mapping button. Similarly as before, follow this procedure:

1. In the Local Scheme (ITP -> DEV) declare which fields you want to Expose and save it. NB: Make sure that all the fields that you declare are on the Create and Edit screens in order to avoid errors.
2. Go to the Remote Scheme (DEV->ITP) and declare the same fields to be Exposed.
3. Go back to the Local Scheme and configure the Receive section
4. Finally, go to the Remote Scheme and configure the Receive section as well.

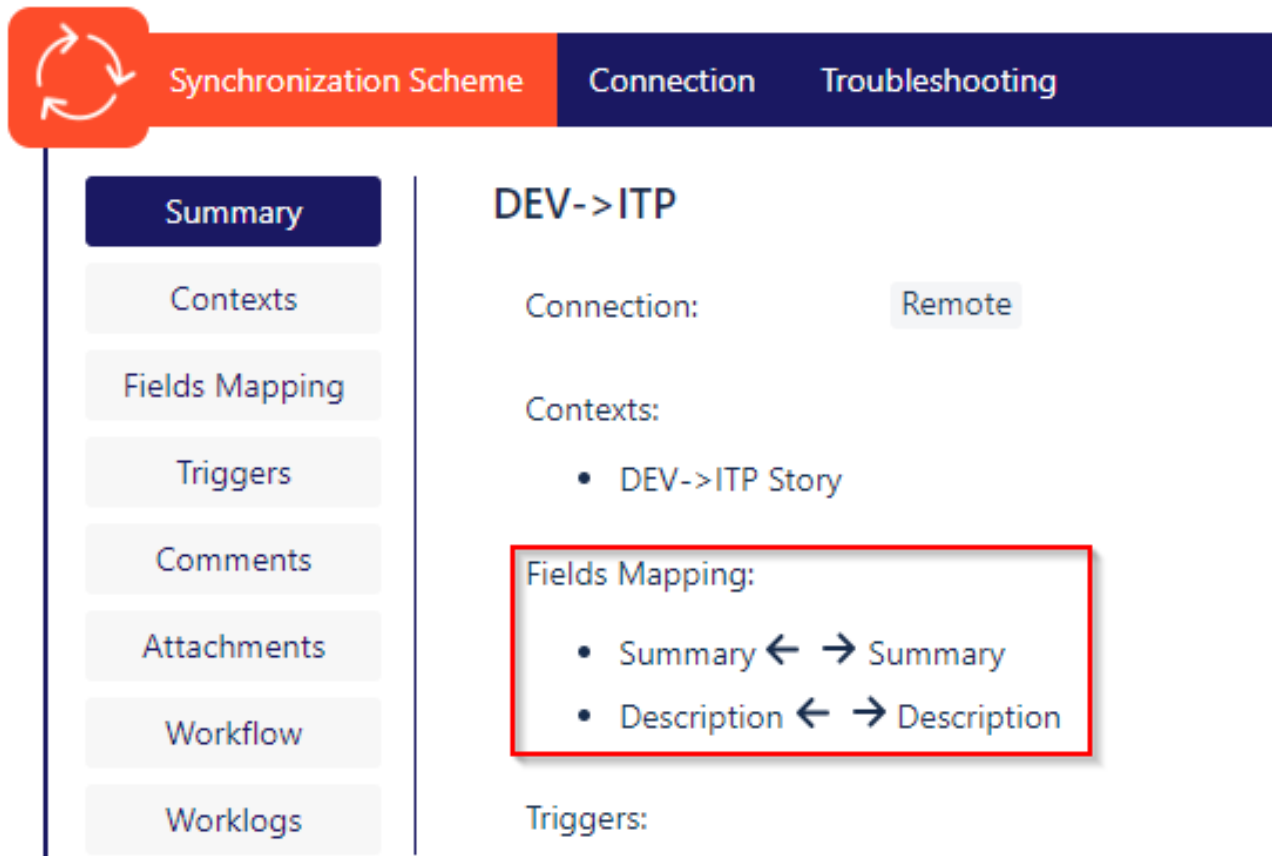
You should get the following result in both Synchronization Schemes:

The screenshot displays the 'Synchronization Scheme' configuration interface. The top navigation bar includes 'Synchronization Scheme', 'Connection', and 'Troubleshooting 3'. A left sidebar contains menu items: Summary, Contexts, Fields Mapping (highlighted), Triggers, Comments, Attachments, Workflow, and Worklogs.

The main content area is titled 'ITP->DEV' and is divided into two sections:

- Expose Section:** Features an 'Expose' icon and a '+' button. It contains a table with 'Exposed Field' and 'Actions' columns. The 'Exposed Field' column lists 'Description' and 'Summary', each with a dropdown arrow. The 'Actions' column contains a trash icon for each field. Below the table are 'Save' and 'Discard changes' buttons.
- Receive Section:** Features a 'Receive' icon and a '+' button. It contains a table with 'Local Field', 'Remote Field', and 'Actions' columns. The 'Local Field' column lists 'Description' and 'Summary' with dropdown arrows. The 'Remote Field' column lists 'Description' and 'Summary' with dropdown arrows and a left-pointing arrow between the columns. The 'Actions' column contains a trash icon for each field. Below the table are 'Saved' and 'Discard changes' buttons.

If the mapping configurations are created correctly, you will see the following Fields Mapping section in the Synchronization Scheme summary.



The screenshot shows the configuration interface for a Synchronization Scheme. The top navigation bar includes 'Synchronization Scheme' (highlighted in orange), 'Connection', and 'Troubleshooting'. The left sidebar contains a menu with 'Summary' (selected), 'Contexts', 'Fields Mapping', 'Triggers', 'Comments', 'Attachments', 'Workflow', and 'Worklogs'. The main content area is titled 'DEV->ITP' and displays the following configuration details:

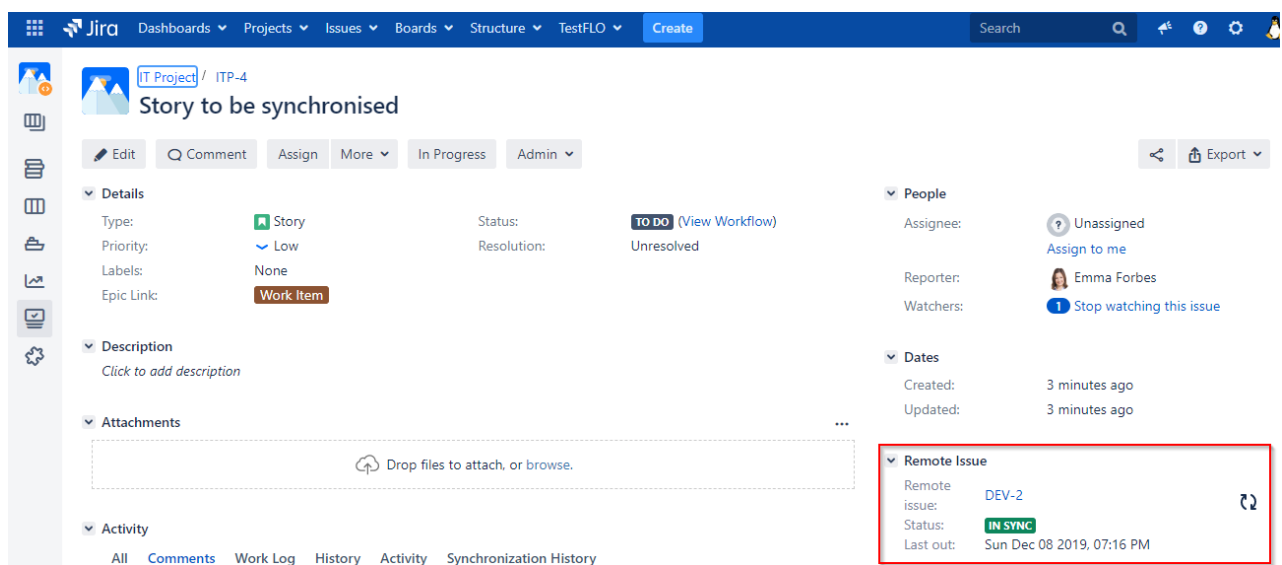
- Connection: Remote
- Contexts:
 - DEV->ITP Story
- Fields Mapping:
 - Summary ← → Summary
 - Description ← → Description
- Triggers:

Other settings like Comments, Attachments and Triggers are pre-configured and for the sake of this simple use case, there is no need to change anything there. Now you are ready to test your IssueSync configuration.

Test run of the IssueSync configuration

Go to the ITP IT Project in your Jira instance and create an Epic Work Item.

In this Epic create a Story issue and give it a name e.g. Story Issue to be synchronized. When the issue is created open it and review the Remote Issue section of this issue. If the synchronization has been successfully completed, you will see similar results.



The screenshot shows a Jira issue page for 'ITP-4' with the title 'Story to be synchronised'. The issue is a Story with a Low priority and is currently in the 'TO DO' status. The 'Remote Issue' section is highlighted with a red box, showing a remote issue in the 'DEV-2' project with a status of 'IN SYNC' and a last update time of 'Sun Dec 08 2019, 07:16 PM'. The page also includes sections for Details, Description, Attachments, and Activity.


When you click on the Remote issue link, you will see the issue that has been created automatically in the other project.

Remote Issue

Remote issue: **DEV-2**

Status: **IN SYNC**

Last out: Sun Dec 08 2019, 07:16 PM

 Give feedback

When you create a comment in this issue (1) you will see how IssueSync instantly is triggered to synchronize the comment in the other (Remote) issue (2).

The screenshot shows the Jira interface. On the left, under the 'Activity' tab, a comment is being added by 'Jira Admin' with a red circle '1' next to it. On the right, the 'Remote Issue' section shows the remote issue 'ITP-4' with a status of 'SYNCHRONIZING' and a red circle '2' next to it. A refresh icon is visible next to the remote issue details. Below the comment, there is a 'Comment' input field.


When you press the Refresh button, you will see the confirmation of the synchronization process.


The first screenshot shows the 'Remote Issue' section with the status 'SYNCHRONIZING' and a refresh icon highlighted by a red box and arrow. The second screenshot shows the same 'Remote Issue' section with the status 'IN SYNC' and the refresh icon. Below the second screenshot, there is a 'Give feedback' button.

Now go back to the ITP project, and see how the comment looks like.

▼ Activity

All Comments Work Log History Activity Synchronization History

▼  Sync User added a comment - 7 minutes ago

DEV-2 ITP->DEV Story (Local)  added comment - 08/Dec/19 07:20 PM
This is a comment in the remote project.

Optional exercise:

You can drop an attachment in one Story issue and see that the attachment will appear in the Story of the remote project as well.





DEVINITI

Technology-Driven Results