A Pioneering Large Scale Migration to Atlassian Cloud

As presented by
Lakshmi Remani
Technical Consultant, Adaptavist

Adaptavist

Agenda

Introduction

History of Migrations

Atlassian Tooling

Introduction to Arm

The Project

Summary

Introduction

Adaptavist

Introduction: Speaker

Lakshmi Remani

Technical Consultant at Adaptavist

8+ Years in Technology Industry

Leading the second major project utilising the process and tooling of this presentation

Introduction: Adaptavist

Founded in 2005

More than 300 Employees

Headquartered in London

Offices in the US, Canada, Spain and Malaysia

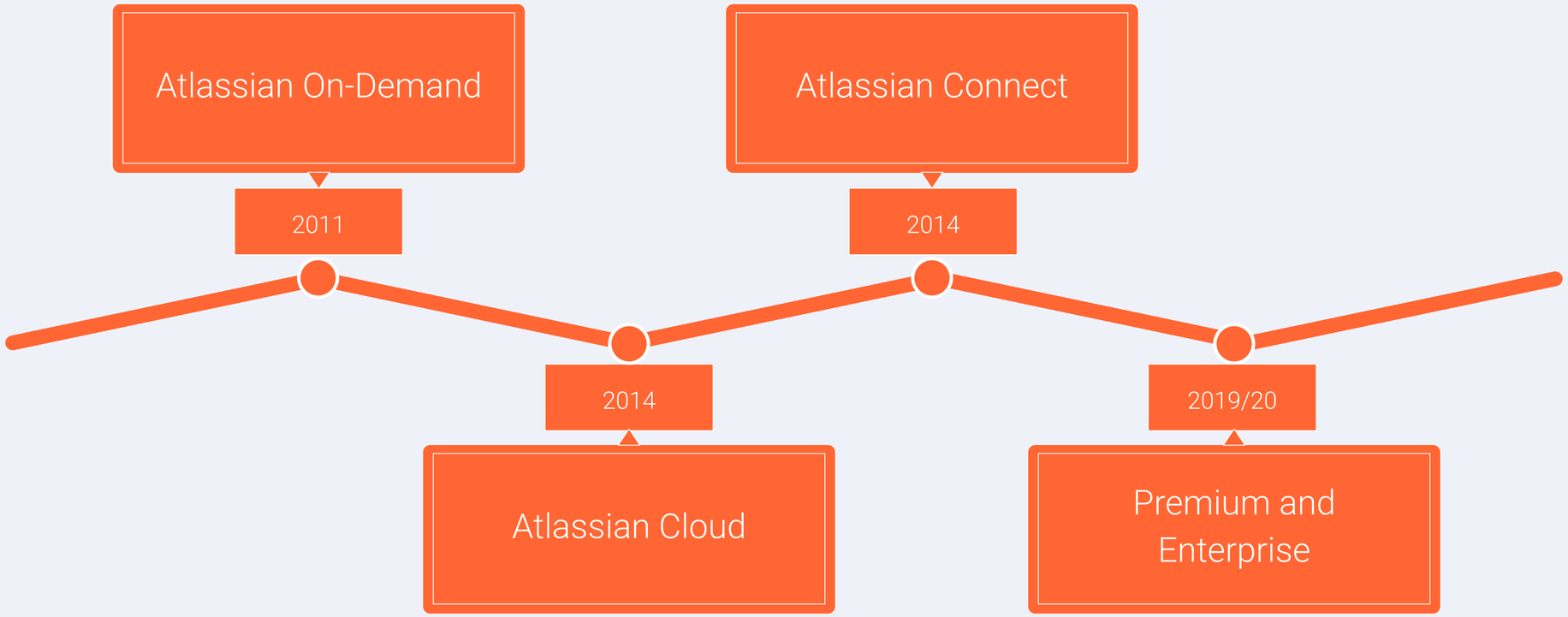
Adaptavist

slack
OFFICIAL PARTNER

ATLASSIAN
Platinum
Top Vendor

ATLASSIAN
Platinum
Solution Partner
ENTERPRISE

Introduction: A Brief History of Atlassian Cloud



Introduction: Cloud Premium and Enterprise

Cloud **Premium**

99.9% Uptime SLA
5000 Maximum Users

Cloud **Enterprise**

99.95% Uptime SLA
Unlimited Users
US/EU **Data Residency**
Built-in **Atlassian Access**

24/7 Premium Support
Unlimited Storage
Advanced Features
Sandbox Environment (coming soon)

Introduction: Focus Area

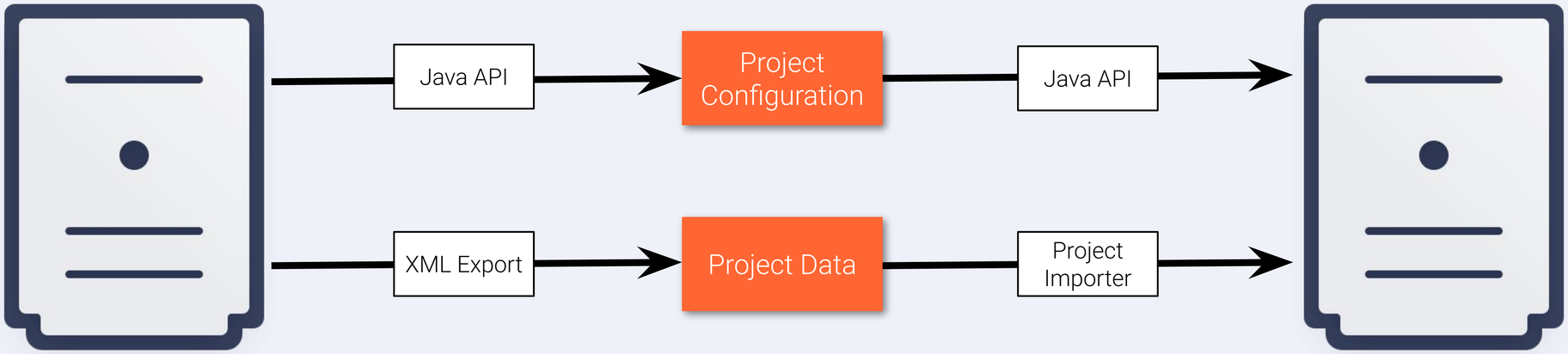
Introduction: Summary of other key tools

For Server to Cloud, the Confluence Cloud Migration Assistant works well.
For other cases, XML Space export is typically sufficient.

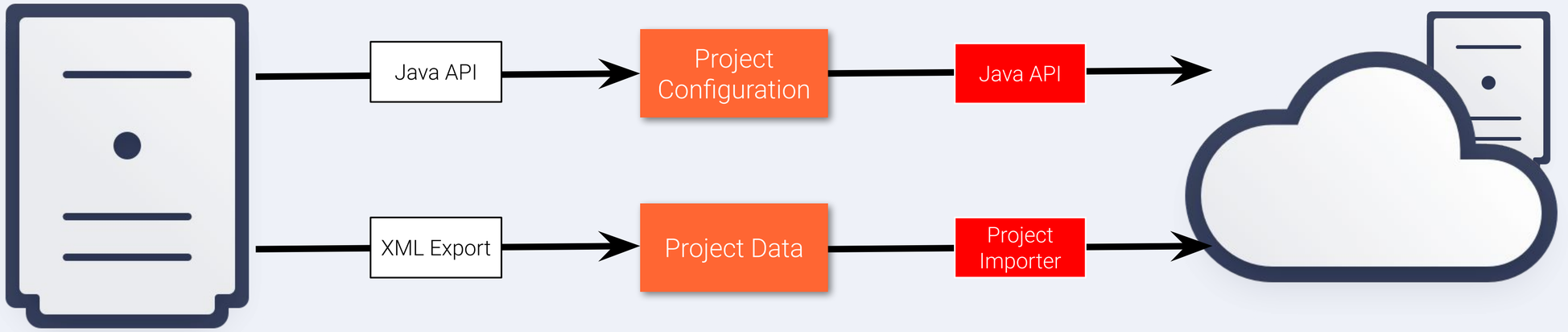
Standard Git migration will move all data, configuration is typically a manageable manual task.

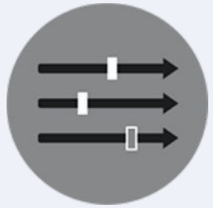
History of Migrations

History of Migrations: Server to Server Migrations



History of Migrations: Server to Cloud Challenges





History of Migrations: Cloud Migration Options

Until recently, there have been just 3 mechanisms by which **data** can be imported into Atlassian Jira Cloud environments:

XML (Site) Import:

Entire Jira site

Does not manage App data

CSV Import:

Simplistic Mechanism

Imports Single Project

Does not import issue history

Does not manage App data

Does not migrate project configuration

JSON Import:

Complex Mechanism

Imports Single Project

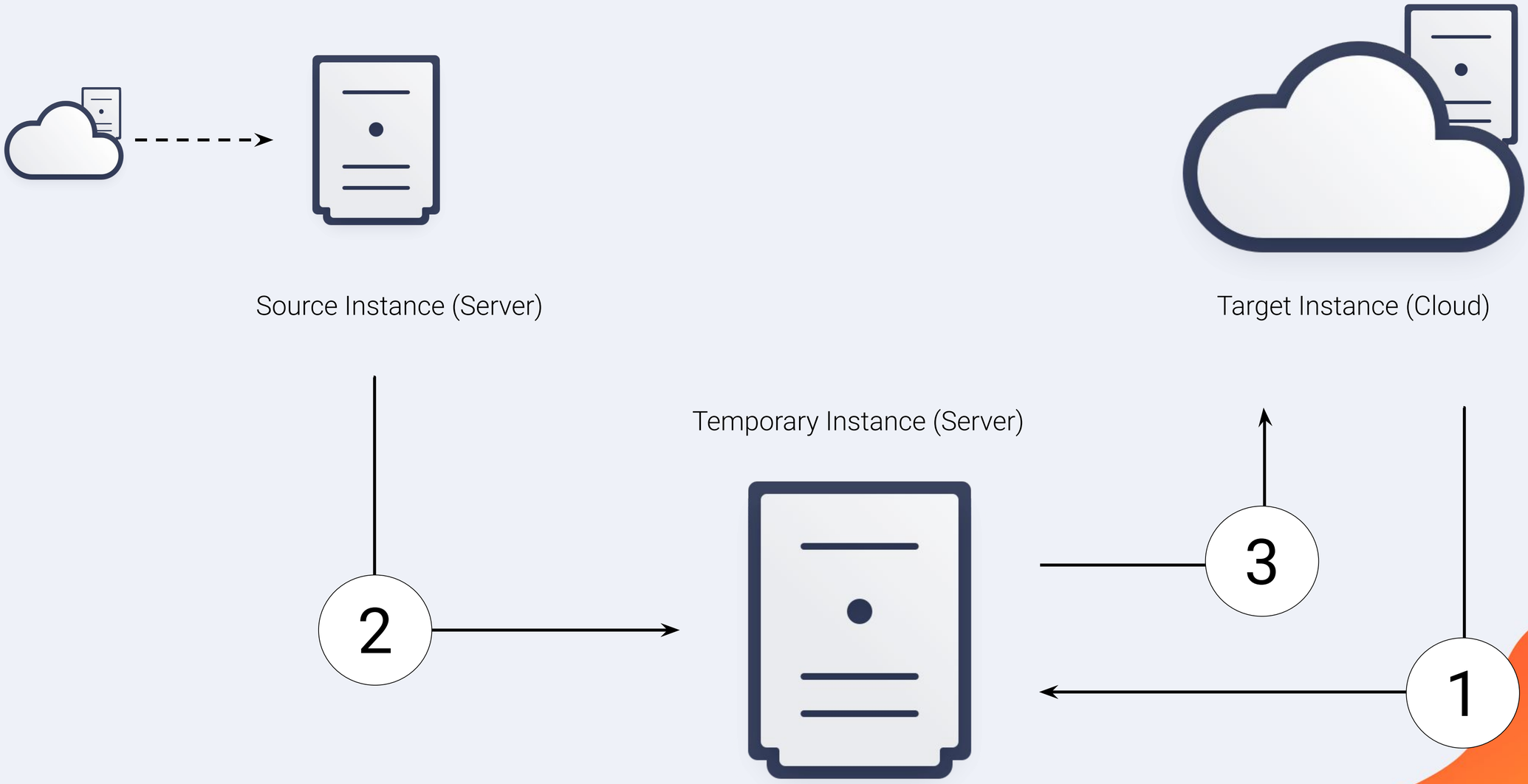
Imports issue history

Does not manage App data

Does not migrate project configuration

Heavily extensible

History of Migrations: Original Merge Process



Atlassian Tooling

Adaptavist

Atlassian Tooling: Intro to the Jira Cloud Migration Assistant

Jira Cloud Migration Assistant (**JCMA**)

Released **March 2020**, been in Beta since 2019

App for Jira **Server**

Pushes data from Jira **Server** to Jira **Cloud**

Atlassian Tooling: Features of the JCMA

Configuration Schemes

Agile Boards

Users and Groups into Atlassian ID

Project-level data (e.g. Components, Versions)

Watchers

Links

Atlassian Tooling: How the JCMA “Cheats”

The JCMA is able to migrate **Configuration Schemes**

It does this by using **non-public** APIs

It is currently **impossible** to replicate this outside of Atlassian

Atlassian Tooling: Challenges with the JCMA

Issue data migration is **unreliable**

App-specific data is not migrated at all

Some **out-of-the-box** Custom Fields are unsupported

Will **only** migrate from Server to Cloud

Atlassian Tooling: Working with Atlassian

Krisz Kovacs (Adaptavist UK)

December 2019 (**during Beta**)

Atlassian HQ in **Sydney, Australia**

Improving Cloud Migration Tooling

Atlassian Tooling: Working with Atlassian

Public Administration API **not** available in short-term

Improvements to **logging** based on real-world experience

Better understanding of **real** customer needs in this space

Building **stronger relationships** between developers of JCMA and teams working with end-users

Introduction to Arm

Introduction to Arm: Who are Arm?

Arm Holdings Limited

Semiconductor and Software Design

Founded in 1990

Headquartered in Cambridge, UK

Over 6,000 global employees

Introduction to Arm: Where were Arm?

Introduction to Arm: Where were Arm?

Introduction to Arm: Where were Arm?

The Project

Adaptavist

The Project: Project Scale

Adaptavist team of ~**4-6 FTE**

Around **10%** of teams successfully migrated

12 Months into Project
12+ Months remaining

The Project: Requirements and Limitations

Hundreds of teams to migrate

Both Server and Cloud Sources

Significant App data to migrate

Migration needs to occur in batches

Performant upon migration

Process needs to complete with minimal downtime

The Project: JCMA fit to Requirements

Hundreds of teams to migrate

Both Server and Cloud Sources

Significant App data to migrate

Migration needs to occur in batches

Performant upon migration

Process needs to complete with minimal downtime

The Project: Cloud Migration Options Redux

Until recently, there have been just 3 mechanisms by which **data** can be imported into Atlassian Jira Cloud environments:

XML (Site) Import:

Entire Jira site

Does not manage App data

CSV Import:

Simplistic Mechanism

Imports Single Project

Does not import issue history

Does not manage App data

Does not migrate project configuration

JSON Import:

Complex Mechanism

Imports Single Project

Imports issue history

Does not manage App data

Does not migrate project configuration

Heavily extensible

The Project: Cloud Migration Options Redux

Until recently, there have been just 3 mechanisms by which **data** can be imported into Atlassian Jira Cloud environments:

XML (Site) Import:

Entire Jira site

Does not manage App data

CSV Import:

Simplistic Mechanism

Imports Single Project

Does not import issue history

Does not manage App data

Does not migrate project configuration

JSON Import:

Complex Mechanism

Imports Single Project

Imports issue history

Does not manage App data

Does not migrate project configuration

Heavily extensible

The Project: Cloud Migration Options Redux

Until recently, there have been just 3 mechanisms by which **data** can be imported into Atlassian Jira Cloud environments:

XML (Site) Import:

Entire Jira site

Does not manage App data

CSV Import:

Simplistic Mechanism

Imports Single Project

Does not import issue history

Does not manage App data

Does not migrate project configuration

JSON Import:

Complex Mechanism

Imports Single Project

Imports issue history

Does not manage App data

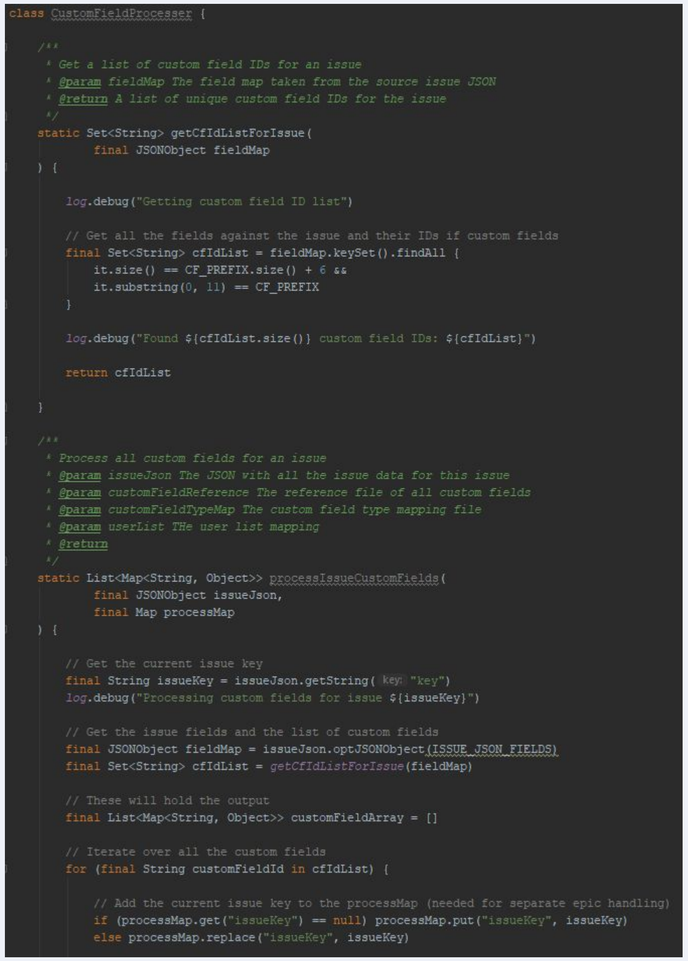
Does not migrate project configuration

Heavily extensible

The Project: Technical Approach

JCMA	Configuration (Server sources)
JSON	Issue Data, Issue History, Users, Links
Extension	App Data Mapping, User Mapping, Attachments Streaming

The Project: Adaptavist Tooling



```
class CustomFieldProcesser {

    /**
     * Get a list of custom field IDs for an issue
     * @param fieldMap The field map taken from the source issue JSON
     * @return A list of unique custom field IDs for the issue
     */
    static Set<String> getCfIdListForIssue(
            final JSONObject fieldMap
    ) {

        log.debug("Getting custom field ID list")

        // Get all the fields against the issue and their IDs if custom fields
        final Set<String> cfIdList = fieldMap.keySet().findAll {
            it.size() == CF_PREFIX.size() + 6 &&
            it.substring(0, 11) == CF_PREFIX
        }

        log.debug("Found ${cfIdList.size()} custom field IDs: ${cfIdList}")

        return cfIdList

    }

    /**
     * Process all custom fields for an issue
     * @param issueJson The JSON with all the issue data for this issue
     * @param customFieldReference The reference file of all custom fields
     * @param customFieldTypeMap The custom field type mapping file
     * @param userList THe user list mapping
     * @return
     */
    static List<Map<String, Object>> processIssueCustomFields(
            final JSONObject issueJson,
            final Map processMap
    ) {

        // Get the current issue key
        final String issueKey = issueJson.getString( key: "key")
        log.debug("Processing custom fields for issue ${issueKey}")

        // Get the issue fields and the list of custom fields
        final JSONObject fieldMap = issueJson.optJSONObject(ISSUE_JSON_FIELDS)
        final Set<String> cfIdList = getCfIdListForIssue(fieldMap)

        // These will hold the output
        final List<Map<String, Object>> customFieldArray = []

        // Iterate over all the custom fields
        for (final String customFieldId in cfIdList) {

            // Add the current issue key to the processMap (needed for separate epic handling)
            if (processMap.get("issueKey") == null) processMap.put("issueKey", issueKey)
            else processMap.replace("issueKey", issueKey)
```

Almost 10,000 lines of Java code

Thousands of resource-hours

Mapping for Custom Fields

Mapping for Usernames

Full automation of complex elements

The Project: Custom Field Mapping

Data from source may not match required import format

Requires custom mapping code

Have now been written for many typical Apps

```
com.atlassian.jira.toolkit:participants: mapParticipants
com.valiantys.jira.plugins.SQLFeed:nfeed-standard-customfield-type: mapCustomFieldString
com.tempoplugin.tempo-teams:team.role.customfield: mapString
com.tempoplugin.tempo-accounts:accounts.customfield: mapDict
```

The Project: User Mapping

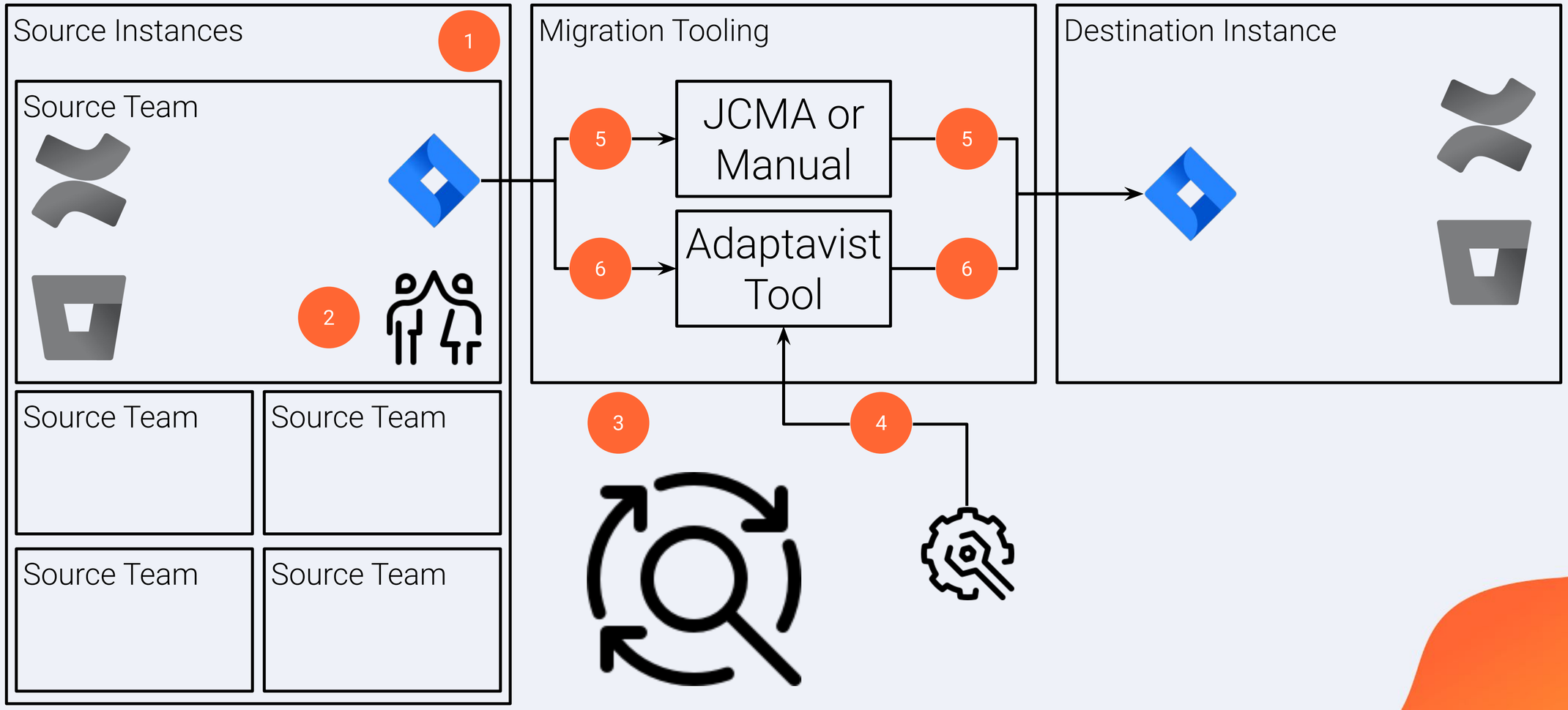
Complicated by Atlassian ID and GDPR limitations

All users must be included

Target Account ID is not simple to get

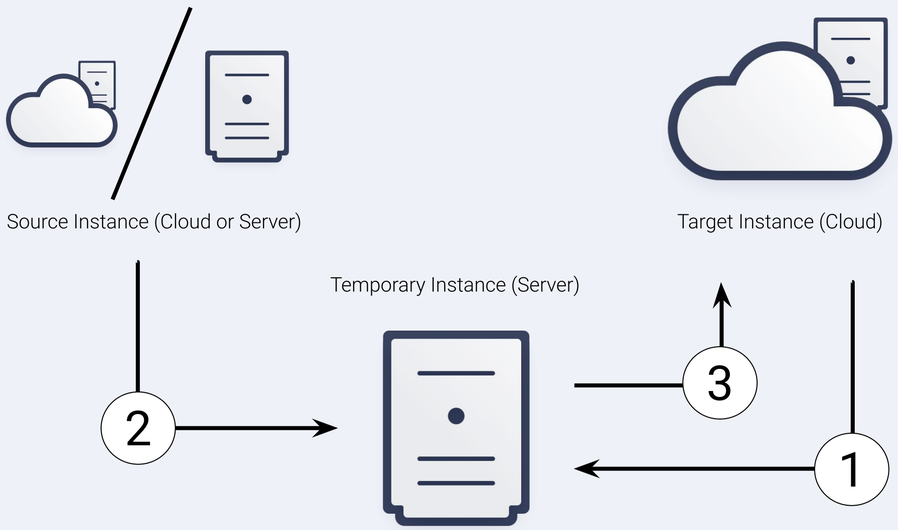

```
{
   "sourceId": "jbloggs1",
   "targetAccountId": "5db3021e0e6b1e3c3959d644",
   "sourceEmail": "jbloggs@example.com",
   "targetUsername": "4729a2ab-8c4e-4af0-9e9f-422593437b98",
   "sourceUsername": "jbloggs1",
   "idMappedFlag": true,
   "usernameMappedFlag": true
},
{
   "sourceId": "lremani",
   "targetAccountId": "5db2c259454fb20d96acdff3",
   "sourceEmail": "lremani@adaptavist.com",
   "targetUsername": "lremani.avst",
   "sourceUsername": "lremani",
   "idMappedFlag": false,
   "usernameMappedFlag": false
},
```

The Project: Approach



Summary

Summary: The Past



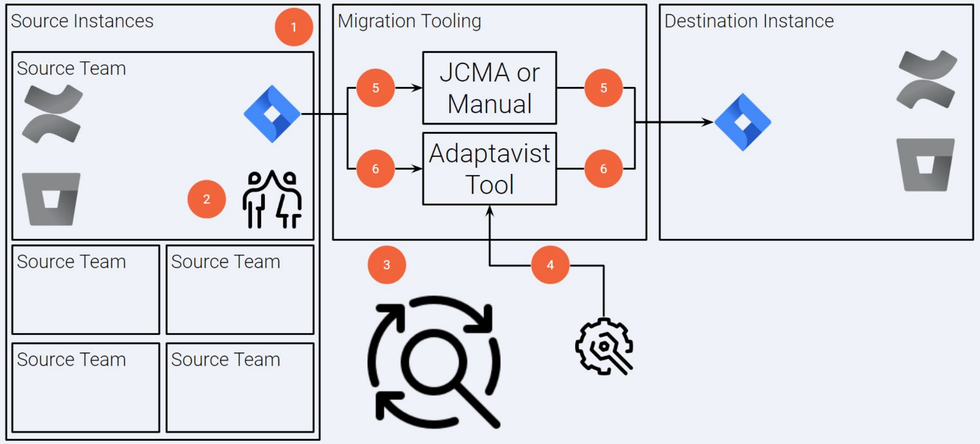
Complex process

Significant manual effort

Huge downtime windows

Typically non-viable approach

Summary: The Present



Simpler process

Reduced manual effort

Minimal downtime windows

Actually viable for Enterprises

Summary: The Future

Summary: The Future

Public Administration API **not** available in short-term

Improvements to App data **reporting**

Better understanding of **real** customer needs in this space

Dedicated Atlassian support team for this project

Summary: The Future

Public Administration API **not** available in short-term

Improvements to App data **reporting**

Better understanding of **real** customer needs in this space

Dedicated Atlassian support team for this project

Summary: The Future

Thank you for listening!

Adaptavist

slack
OFFICIAL PARTNER

ATLASSIAN
Platinum
Top Vendor

ATLASSIAN
Platinum
Solution Partner
ENTERPRISE