

# Kelverion Automation

## Ticket and DevOps Synchronisation Solution

### User Guide

Version 1.2

## Table of Contents

<b>1. Overview.....</b>	<b>3</b>
1.1. Ticket and DevOps Synchronisation Solution Operation .....	4
<b>2. Pre-Installation Information .....</b>	<b>6</b>
2.1. Kelverion Ticket and DevOps Synchronisation Solution Package Contents .....	6
2.2. Integration Packs Required .....	6
2.3. Persistent Data Store.....	6
<b>3. PDS Creation .....</b>	<b>8</b>
3.1. PDS Creation Steps .....	8
<b>4. Solution Installation Steps.....</b>	<b>9</b>
4.1. Runbook Solution Installation.....	9
<b>5. Solution Configuration .....</b>	<b>10</b>
5.1. System Center Orchestrator Configuration.....	10
5.1.1. Kelverion SQL Server IP configuration .....	10
5.1.2. Kelverion Data Manipulation IP configuration.....	10
5.1.3. Kelverion ServiceNow IP configuration .....	11
5.1.4. Kelverion Azure DevOps IP configuration.....	11
<b>6. Customising the Ticket and DevOps Synchronisation Solution.....</b>	<b>12</b>
6.1. Default 'Out of the box' behaviour .....	12
6.2. Triggering Runbooks.....	12
6.3. Configuring the Solution to target 'Azure DevOps' .....	12
6.4. Configuring the solution for a different Service Desk.....	13
<b>7. Installing Temporary License of Kelverion Integration Packs .....</b>	<b>15</b>
<b>8. Upgrade Warning .....</b>	<b>16</b>
<b>9. Notes .....</b>	<b>17</b>

## 1. Overview

The existence of multiple Service Desks from different vendors is now commonplace across many organisations. However, incorporating reliable communication and transfer of information across such Service Desks in an Enterprise setting is a time consuming, labour intensive and an error prone process.

It typically involves the user raising a call into a Helpdesk (A) to log an incident in a Service Desk (A). The incident is then processed and after a period of time the information is then manually replicated by another team using a different Help Desk (B). Only now can the information be actioned by an engineer exposed to Service Desk (B). At any point during the life span of this incident a sequence of unpredictable scenarios can play out. I.e. the 'urgency' may change in Service Desk (A) while the engineer is investigating. The 'category' in Service Desk (B) may change, resulting in the 'Assigned Engineer' also changing. The ticket may even become 'Closed' or 'Resolved' on either side. All the time during the lifespan of this incident, this information needs to be relayed and transferred to the opposing Service Desk. Not only is this very labour intensive but it is also prone to time delays and human error. The human error element most likely arises during the process of 'identifying what has changed in the incident' in the respective Service Desk.

Up to now automating this process has become a minefield of challenges. System Center Orchestrator provides the platform from which to succeed.

However, even for an experienced Orchestrator developer, there are two significant challenges with envisioning a Helpdesk to Helpdesk or Help Desk to Bug Tracking solution. The first challenge is being able to interface and parse data between the two systems bi-directionally. The second challenge is more complex and entails being able to identify and dynamically update or communicate only the changes that have occurred to the Incident in the other Service Desk.

The Kelverion 'Ticket and DevOps Synchronisation Solution' meets these challenges and delivers. A user can simply raise the incident in ServiceNow and the incident will be created in Service Desk B (e.g. Azure DevOps or Atlassian Jira). Subsequent changes to the incident by either Service Desk will be automatically relayed in isolation to the opposing Service Desk via Orchestrator. The relayed information will be in the form of communication to the respective Work Notes (ServiceNow). The operator is then equipped to update the incident appropriately and accurately.

The Kelverion Ticket and DevOps Synchronisation Solution leverages the Persistent Data Store (PDS) design philosophy and the Kelverion Orchestrator Integration Packs to provide a scalable and robust solution.

Each companies Service Desk and Bug Tracking systems are configured slightly differently and the Incident processes vary so the solution is provided as a flexible

working framework which can be tailored to each specific customer implementation. The benefit of the solution is that the design and operational workflow is in place and only requires customising to accommodate the actual fields present in the Service Desk Incident Ticket or Bug Tracking system Issue Ticket.

This solution is available as a self-installation package for customers proficient with System Center.

For customers who are less familiar with System Center, Kelverion or our partners can provide a complete installation and configuration solution where we work with you to customise the solution for your environment.

This document provides the guidance on how to setup and configure this solution in your environment. It is aimed at experienced System Center users. Users should also reference Microsoft supplied documentation for the System Center tools and Kelverion Integration Pack guides.

### 1.1. Ticket and DevOps Synchronisation Solution Operation

The Automation Solution utilises a common Persistent Data Store (PDS). From a high level, the solution operates as follows:

1. An IT user raises an Incident in ServiceNow with an Urgency, classification, short description and 'Support Group'
2. Orchestrator raises a new Work Item in Azure DevOps with the enrichment of the data retrieved from step 1 (e.g. urgency, classification, short description) as well as updating the work notes section (e.g. 'Entered by: J Bloggs'). Once the record is inserted, the raw data is recorded in the PDS allowing for comparison later.
3. The IT user updates incident in ServiceNow. The Runbooks identify this an update in the PDS. The Azure DevOps work item is updated with a description of the changes. This will typically be things such as changes to 'assigned to'.

OR/AND

The Azure DevOps work item is updated and an entry is created in the PDS. The PDS data is compared with the original 'work item' or the most recent update to the work item, whichever is newer. The runbooks identify where text() has previously existed for a value e.g. Urgency: 'Low' (was) 'High' (now).

4. The ServiceNow incident is marked as 'Resolved' or 'Closed' by the ServiceNow Operator. A record is updated in the PDS. The Azure DevOps work item is updated with the resolved notes from ServiceNow.

OR/AND

The Azure DevOps work item is 'Resolved' or 'Closed'. The record is updated in the PDS. The ServiceNow incident has its 'Work Notes' updated with the resolve notes from the Azure DevOps work item.

The solution is based around our experience of building this type of automated offering for many customers. The Automation Solution provides the foundations out of the box and allows extension and modification to tailor the solution to exact customer requirements.

By using the Persistent Data Store approach any complex decision logic can be handled using the database and Orchestrator runbooks. You can extend the runbooks yourself or we can provide consultancy to help you with the design update and runbook modifications.

## 2. Pre-Installation Information

### 2.1. Kelverion Ticket and DevOps Synchronisation Solution Package Contents

Kelverion Ticket and DevOps Synchronisation Solution contains the following elements:

- Solution Runbook export file
- Data Manipulation XML file
- SQL script for the Persistent Data Store

### 2.2. Integration Packs Required

The solutions requires the following Integration Packs:

#### **Kelverion**

- SQL Server Integration Pack
- ServiceNow

Before importing any Runbooks please ensure these Integration Packs are installed in Orchestrator. If you do not already have Kelverion Integration Packs, they can be downloaded for evaluation from our website.

### 2.3. Persistent Data Store

The Persistent Data Store (PDS) is a SQL Server database that is used by this Solution to allow all of the actions that the Runbooks take to be carried out in a robust way. The use of the database at each “step” allows us to design the Runbooks such that each Runbook is simple and can be considered a discrete unit. In programming terms it allows the Runbooks to be modular.

In your environment there may be a number of constraints that control the creation of a new database. For example, the location of the log and data files, the recovery options that should be used, and the collation of the server. These requirements are typically specified by the DBA responsible for your database server. These options do not affect the Runbooks so please use the appropriate options for your environment.

#### **Location**

Typically, the PDS is created on the same database instance as is used for the Orchestrator database. There is no specific requirement that this must be the case. In environments where there is very high load you may find that creating the PDS on a different database instance advantageous.

#### **Database version**

The Runbooks provided, have been tested against SQL 2019 \ 2022 with the latest patches and updates applied. You may need to modify the SQL Script to get it to operate in your environment or to install it on older versions of SQL Server.

### **Collation**

The Runbooks have all been developed on systems using **Case Insensitive** collations, the specific collation setting used for your environment must be case insensitive other than that though the setting can be chosen as appropriate for your environment.

### **Sizing**

The minimum recommended size of the PDS is 1GB.

The amount of space required will depend on the two following factors:

- Number of requests processed
- Housekeeping frequency

### 3. PDS Creation

Each Kelverion Automation Solution uses a set of common tables within the PDS Database and a set of tables specific to itself.

As part of each solution package you are provided with a SQL script which will generate the PDS database tables required for the solution. When the SQL scripts are executed they check for the existence of each table they required in the PDS database. If this is a new installation they will create both the Common Tables and their Solution Specific database tables. If you are already using a Kelverion Automation Solution then the script will detect that some of the tables this solution requires already exist and the script skips these table creation steps and creates only the tables which do not exist in your installation.

#### 3.1. PDS Creation Steps

1. Create a New Database on your SQL Server called PDS\_Live or connect to your existing PDS\_LIVE database
2. Then execute the SQL Script provided within the PDS\_Live database you created.
3. Once the PDS\_Live database is created you must ensure the Orchestrator Runbook Server Service Account has as a minimum Read and Write Access permissions to the PDS\_Live database.



## 4. Solution Installation Steps

The installation steps below assume that System Center Orchestrator is installed. These steps should be followed by an experienced user.

### 4.1. Runbook Solution Installation

The Runbook Solution installation steps are as follows:

1. Install the required Kelverion Integration Packs. Follow the guidance given as per respective IP User Guides.
2. Import the Ticket and DevOps Synchronisation Solution Runbook Set.
3. Place the 'Datamanip\_TicketSync.xml' and the 'Diff.xml' in a new directory called 'C:\Orchestrator\TicketSync\' on each runbook server.

## 5. Solution Configuration

To use the solution, you have to do a series of simple configuration steps to make the Runbooks operate in your environment. These configuration settings are made from the Orchestrator Options menu for each of the products listed below.

### 5.1. System Center Orchestrator Configuration

This solution depends on the following Integration Packs. These will need to be configured in the Orchestrator Runbook Designer Console under the Options Menu and the relevant entry. It is important that the names specified in the configuration are accurate as the Runbooks are looking for them specifically. Where different configurations are required to the same system then additional entries can be made.

#### 5.1.1. Kelverion SQL Server IP configuration

This integration pack connects the Runbooks to the SQL Server that hosts the Persistent Data Store. Use the table below as a guide on how to configure the Integration Pack.

Item	Configuration
Name	PDS_LIVE
Type	SQL Server Options
Server Name	*****
Database Name	PDS_LIVE
Authentication Scheme	Windows Authentication
Username	
Password	
Connect Timeout	15
Enable Column Encryption	False

#### 5.1.2. Kelverion Data Manipulation IP configuration

This integration pack allows the runbooks to deal with the manipulation of text, that allows to more easily pass data through the activities in the Runbooks.

Item	Configuration
Name	TicketSync
Type	XML Specification
Specification File	C:\Orchestrator\TicketSync\Datamanip_TicketSync.xml

**5.1.3. Kelverion ServiceNow IP configuration**

This integration pack connects the Runbooks to ServiceNow.

Item	Configuration
Name	ServiceNow
Type	ServiceNow Configuration
ServiceNow URL	https://*****.service-now.com
User Name	*****
Password	*****
User Date Format	Yyyy-MM-dd
User Time Format	HH:mm:ss
Proxy Server URL	
Proxy User Name	
Proxy Password	
Proxy Domain	
Client ID	
Client Secret	
Access Token URL	
Refresh Token	
Skip Certificate Validation	False

**5.1.4. Kelverion Azure DevOps IP configuration**

This integration pack connects the Runbooks to Azure DevOps.

Item	Configuration
Name	DevOps
Type	Azure DevOps
Server URL	https://dev.azure.com/*****/
User Name	*****
Password	*****
Custom Config File	
Personal Access Token	
Skip Certificate Validation	False

## 6. Customising the Ticket and DevOps Synchronisation Solution

### 6.1. Default 'Out of the box' behaviour

A customer may wish to use a different Service Desk product to drive or target the Ticket and DevOps Synchronisation Solution.

The Runbook sets are designed as a solution which provides the foundations out of the box and allows extension and modification to tailor the solution to exact customer requirements. It is expected therefore that a customer may wish to extend or customise the solution to talk to different Service Desks.

Out of the box, the Solution is configured to target Azure DevOps. Using the Variable 'TargetServiceDesk'.

### 6.2. Triggering Runbooks

The runbooks are triggered on 'New' ServiceNow Incidents where the 'short description' starts with **TASK**.

20.1.1-Monitor SNow New

**Monitor Records Properties**

**Filters**  
Define the filters used by the activity.

General  
Properties  
**Filters**  
Run Behavior

Name	Relation	Value
State	Equals	New
Active	Equals	True
Short description	Starts with	TASK

### 6.3. Configuring the Solution to target 'Azure DevOps'

Out of the box, the Runbooks are provided to process Azure DevOps workitems. The runbook '**33.1.1-Create Work Item**' will need to be adjusted to match the required target project.

**Configuration**

Name:  ...

**Properties**

Collection	kelverion-uk-dev
Project	Azure Automation
Work Item Type	Issue
Created By	Azure Automation Build Service (kelverion-uk-dev)
Reason	Added to backlog
State	To Do
Title	{short description from "Parse Text -Incident data"

Optional Properties...

The current configured project will differ in the customer environment.

Any custom fields will need to be adjusted in the data manipulation XML file used in the Compose Text activity.

**Configuration**

Name:  ...

**Properties**

Compose Configuration	AzureDevOps WorkItem
AreaPath	{Area Path from "Get Work Item"}
State	{State from "Get Work Item"}
Title	{Title from "Get Work Item"}
AcceptanceCriteria	{Acceptance Criteria from "Get Work Item"}
AcceptedBy	{Accepted By from "Get Work Item"}
AcceptedDate	{Accepted Date from "Get Work Item"}

Optional Properties...

#### 6.4. Configuring the solution for a different Service Desk

You will need to ensure that there is a suitable integration pack available for the Service Desk that you need to connect to.

The runbooks will need to be updated to change the chosen activities to the new Service Desk.

The Data Manipulation file that is utilized in the Parse \ Compose Text activities will need to be updated. There are already some example configurations in the file `Datamanip_TicketSync.xml` for the following ServiceDesks:

- Azure DevOps
- ServiceNow
- Microsoft System Center Service Manager

- Atlassian Jira

```

<ka:DataManipulation xmlns:ka="http://www.kelverion.com">
  <ka:ParseText name="AzureDevOps WorkItem">
  <ka:ComposeText name="AzureDevOps WorkItem">
  <ka:ParseText name="Service Manager Incident">
  <ka:ComposeText name="Service Manager incident">
  <ka:ParseText name="ServiceNow Incident">
  <ka:ComposeText name="ServiceNow Incident">

  <ka:ParseText name="Jira Issue">
  <ka:ComposeText name="Jira Issue">

  <ka:ParseText name="SCSM Action Log">
  <ka:ComposeText name="SCSM Action Log">
  <ka:ParseText name="SCSM Comment Log">
  <ka:ComposeText name="SCSM Comment Log">
  <ka:ParseText name="AD User">
  <ka:ComposeText name="AD User">
  <ka:ParseText name="Windows Computer">
  <ka:ComposeText name="Windows Computer">
</ka:DataManipulation>

```

## 7. Installing Temporary License of Kelverion Integration Packs

To run the solution you will need a full or evaluation licence key for Kelverion Integration Packs.

The licence files need to be copied into a folder called C:\Program Files\Kelverion Automation\Licenses. If this folder does not already exist on your system please first create the folder C:\Program Files\Kelverion Automation\Licenses and then copy the attached files into it.

The license key is regularly updated as it includes a specific license end date after which the product will no longer work. If you have a license or date format error on trying to run this product please contact [info@kelverion.com](mailto:info@kelverion.com) detailing date of download and error details.

To purchase a license please contact your Kelverion representative, reseller or email [info@kelverion.com](mailto:info@kelverion.com)

## 8. Upgrade Warning

The runbooks provided in this Automation Solution are provided for installation in a clean Orchestrator environment. If you have deployed any previous versions of this Automation Solution, then installing this version will overwrite any changes you have made to the currently deployed Runbooks.

You can either delete you existing Runbook deployment and then install this new Automation Solution set or manually upgrade your existing deployment.



## 9. Notes

Kelverion Automation Ltd  
Compass House,  
Vision Park,  
Chivers Way,  
Histon  
Cambridge CB24 9AD  
Email: [info@kelverion.com](mailto:info@kelverion.com)  
Web: [www.kelverion.com](http://www.kelverion.com)