Kelverion

System Center Orchestrator Best Practices Guide

Published by

Greg Charman

VP of Services and Solutions

Contents

Introduction

This white paper will assist anyone who has been tasked with taking the System Center suite of solutions and using Orchestrator Runbooks to better integrate, automate or orchestrate IT systems and processes to support business practices and ultimately achieve cost savings by driving greater efficiencies across the organisation.

The insights contained in this document reflect over 30 years' experience from the Kelverion team in assisting customers to achieve success in the design, building and deployment of Runbooks using System Center Orchestrator.

Having access to Orchestrator is akin to receiving a box of LEGO® bricks with no instructions as to what is to be built.

Therefore it is important to read and implement the recommended best practices stated within this guide before you start to build your automation processes. Failing to do so can result in errors, poor service, inefficient processes being automated, increased costs and contributing to a lack of IT process governance and control.

This paper will focus on Orchestrator Runbook development as an integral part of IT Process projects to ensure that IT professionals can create real value from deploying System Center solutions and, where it is appropriate, integrating System Center with other IT Systems Management or Service Management solutions.

Regardless of the scenarios that require a company to implement IT Process Automation, no two organisations are the same. Therefore, the success criteria of implementing IT Process Automation must firstly be determined by those in charge to establish clear goals to achieve and, secondly, to guarantee minimal disruption to the business through the transition from inefficient to efficient processes made more effective through automation.

Some IT Professionals struggle when using System Center Orchestrator as it not a simple IT solution to be installed and learnt on the job. It is highly customisable to support the business processes you wish to automate which can become a double edged sword for IT staff.

For example, it is simple to install Orchestrator and create simple workflows. However, very quickly, you can encounter problems when designing more complex activities and representing them in Orchestrator which involves multiple branches and loops.

From our experience, there are common mistakes to be avoided when implementing the solution and this guide will counter frustrations in your efforts by learning about our best practices and adopting them for greater success in your IT Process Automation projects

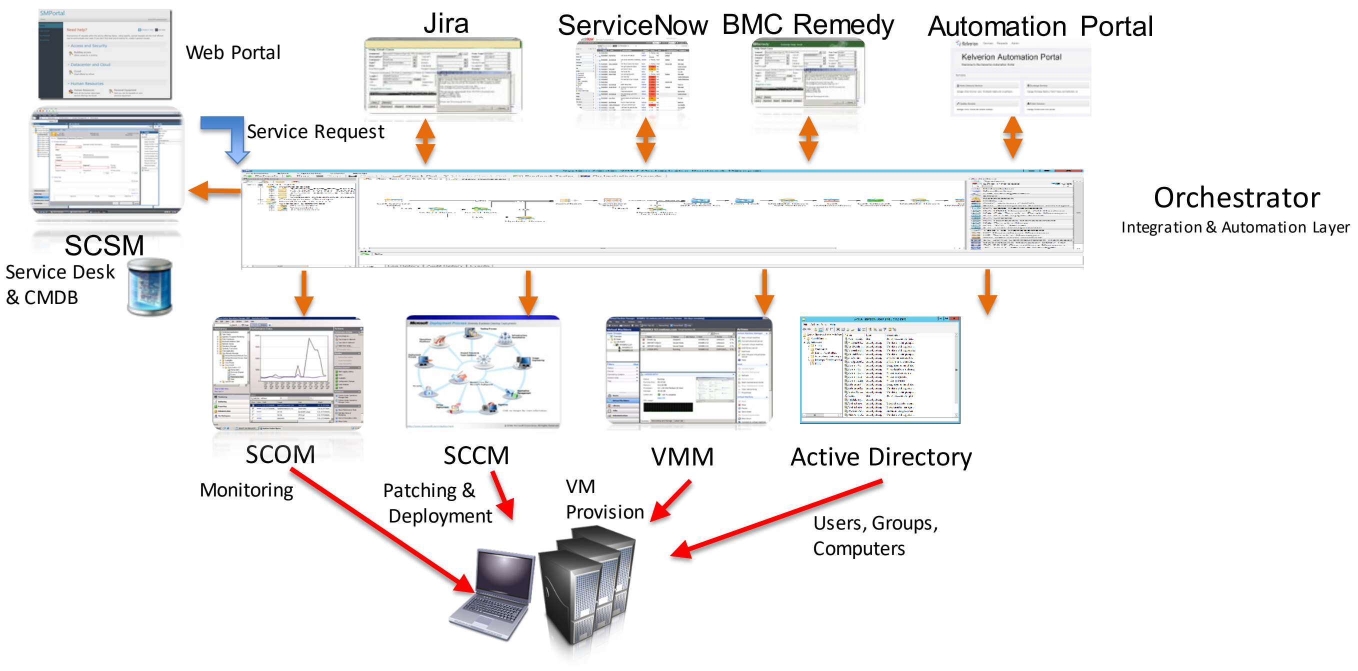
Understanding Orchestrator

Before reading the content of this best practice guide from Kelverion, it is essential for you to have a clear understanding of System Center Orchestrator and Runbooks.

What is System Center Orchestrator?

Orchestrator is one of the nine components of the Microsoft System Center Suite. An investment in the System Center Suite allows you to access the full capability of Orchestrator. Originally a third-party product named Opalis, which Microsoft acquired in 2009, 'Orchestrator provides a simplified way of building complex automation'.

Rather than spending time writing lines of arcane code, you can use Orchestrator to, for example, support event monitoring, alerts and actions by using a drag-and-drop process which only takes a matter of seconds to complete. This rapid construction and deployment of automated processes will result in IT service improvement and remove manual, resource-intensive or error prone tasks, allowing your team to focus on resolving more strategic activities and delivering new services.



What is a runbook?

The power of Orchestrator is in providing Runbooks and the individual 'activities' that make up a Runbook. Runbooks contain the 'instructions' for an automated 'task' or 'process'. The individual steps throughout a Runbook are called 'activities'. Within the Runbook, additional 'controls' provide information and instructions to govern the sequence of activities in the Runbook. Runbooks, activities and each Runbook control have configurable properties. You modify these properties to configure the behaviour that your Runbook requires.

In a Service Management scenario, rather than waiting for an end user to notice that a service has become unavailable or having a member of the support team raise a job, you can automate the process entirely through the use of an Orchestrator Runbook.

The alert is resolved, the job is logged and closed, and everything is completed without direct IT personnel intervention. Generally speaking, if you can come up with a procedure to resolve a specific known problem, then you can come up with a way to automate that resolution.

Fundamentally, the Runbook depicts the sequence of operations or work of a person, a group or mechanisms. Runbooks themselves can also be connected together to form more complex solutions to the more complicated processes in the organisation.

Benefits of System Center Orchestrator & runbooks

In summary, 'Orchestrator provides a solution for IT Operations Runbook Automation and provides orchestration, integration and automation of IT processes, enabling companies to define and standardise best practices and improve operational efficiency whilst reducing errors and costs.'

Integration

Orchestrator provides integration with third party service desks, systems management and administration tools as well as with databases, operating systems, their resources and system functions. There are a substantial number of Integration Packs available from Kelverion for non-Microsoft companies or offerings, such as Amazon, Citrix, CA Technologies, BMC, Jira, Nagios and ServiceNow.

Common pitfalls to avoid

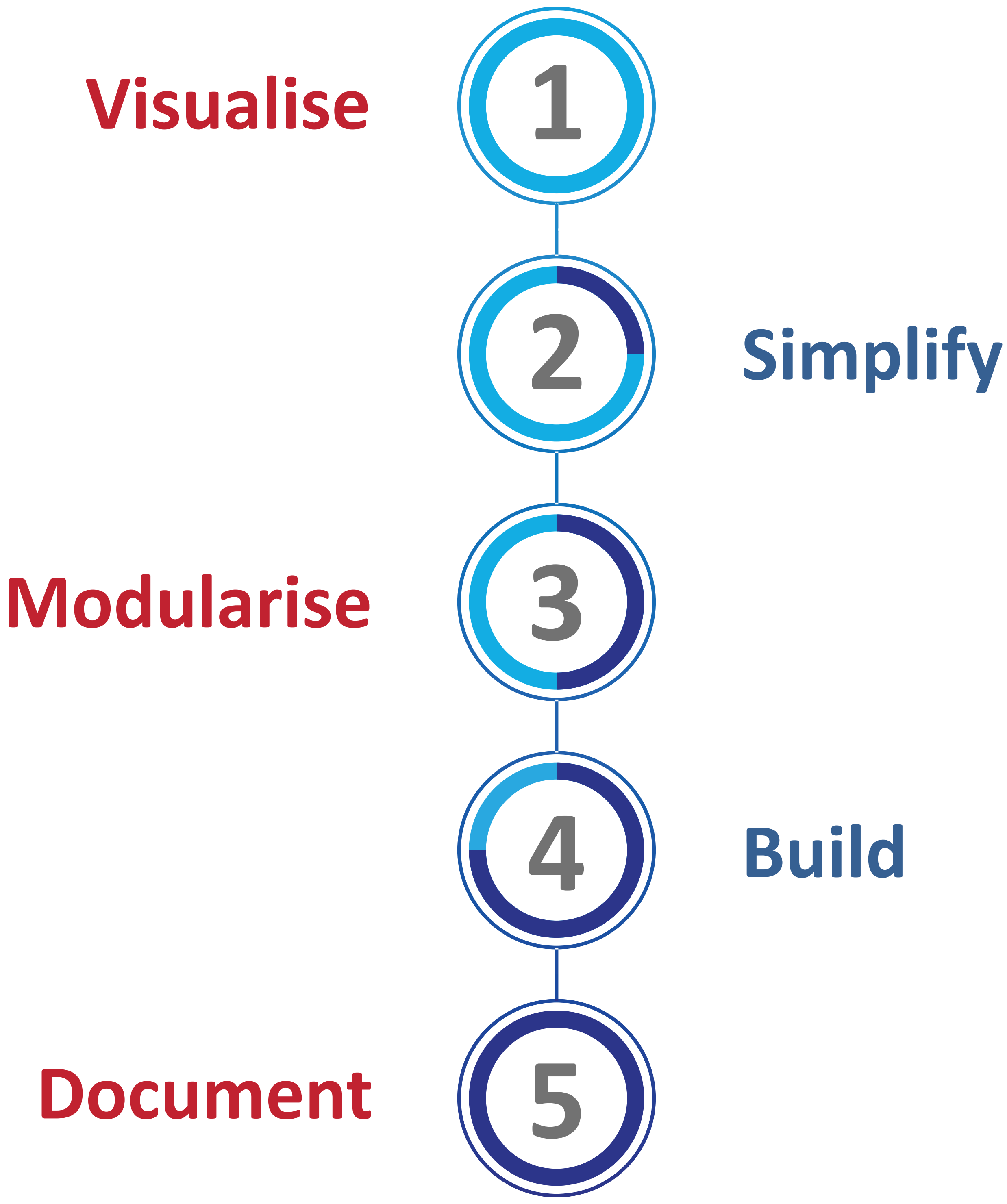
It is important to carefully design and plan your workflows first with knowledgeable experts on system design and those that understand the processes to be automated. It is not necessary to generate reams of documentation - a visualisation of the process design via the use of flow charts, use case and swim lanes is sufficient.

Do **NOT** throw away your paperwork documenting your process; this documentation will prove extremely useful in analysing the initial thought processes, future Runbook and process changes as well as any test issues and future personnel changes.

If you don't keep instructions on your automated system, how will you remember what you built and how it was designed?

The Best Practices Approach

To simplify and best structure your approach to IT Process Automation and the use of System Center Orchestrator, Kelverion advise our clients to follow these proven steps when building Runbooks:



This is the high level approach Kelverion always takes to gain a better understanding of our customers automation requirements

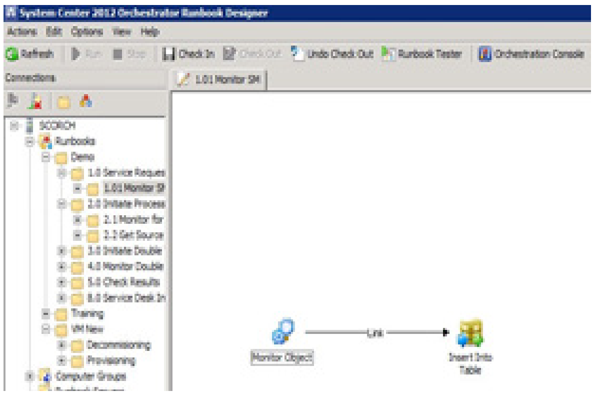
Top Tips for Designing Runbooks

Tip 1. Use a folder & Sub folder structure in your in your runbook layout

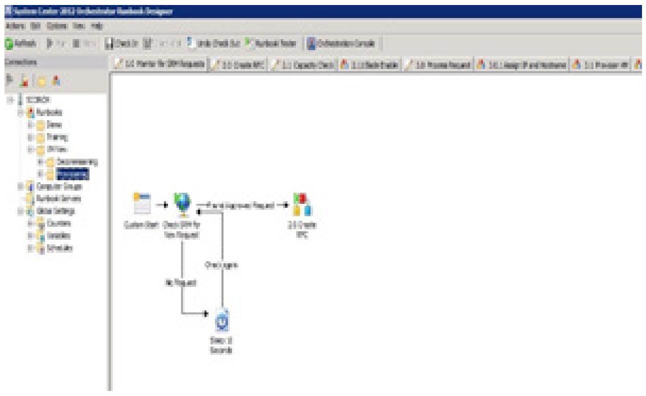
Best practice is to always use a '*Folder*' and '*Sub Folder*' structure for Runbooks (similar to a windows explorer structure). For ease of manageability purposes, only put one Runbook in a folder or sub folder.

why? This helps keep things simple. By having one Folder and one Runbook, you are making the process easy to view and understand.

Good Practice

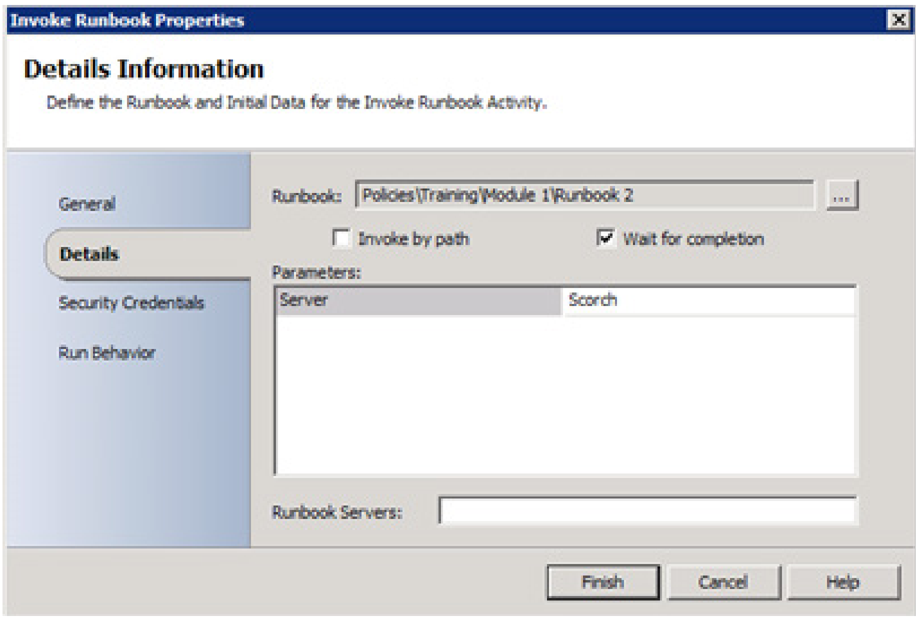


Poor Practice



Tip 2. Use folder numbering

To show the flow of execution and aid supportability, take time to give your folders numbers – for example 1.0, 2.0, 3.0 etc. This additional design feature can often be overlooked but its usefulness should not be under estimated.



Invoke Runbook Properties

Details Information

Define the Runbook and Initial Data for the Invoke Runbook Activity.

General | **Details** | Security Credentials | Run Behavior

Runbook: Policies\Training\Module 1\Runbook 2

☐ Invoke by path ☑ Wait for completion

Parameters:

Server	Scorch

Runbook Servers:

Finish | Cancel | Help

Tip 3. Create common task runbooks - Modularise

When building your workflow solution, an effective approach is to always identify functions which are commonly repeated.

why? This will allow you to take advantage of the ‘build once and use many’ methodology, saving time and increasing Runbook resilience.

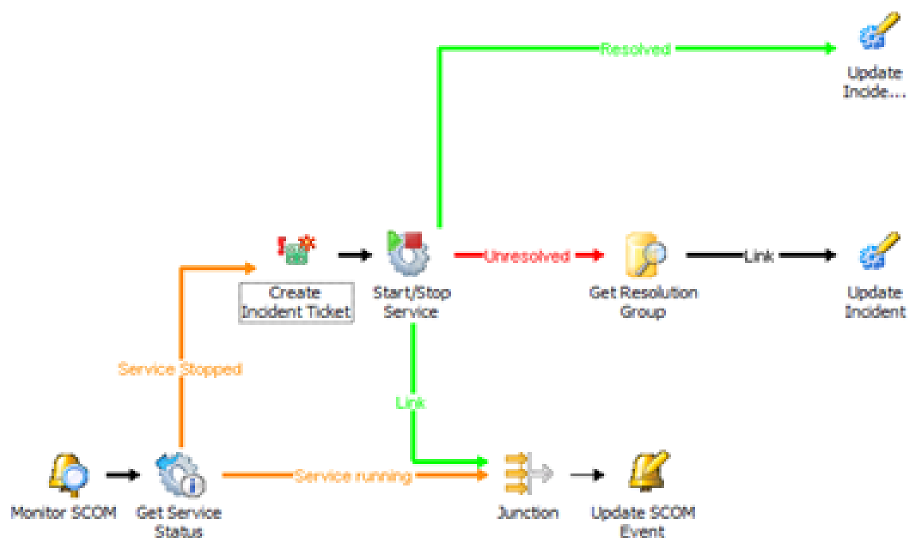
By breaking the various steps of a process into chunks, i.e. modularising, you can re-use these chunks whenever you need to do a task many times. For example, you could modularise common tasks such as ‘*send email*’, ‘*raise an incident ticket*’ and ‘*create a change request*’ etc. By building this type of functionality into your Runbooks, you will avoid having to update Runbooks in multiple locations which may become confusing and cumbersome to maintain. Orchestrator allows you very easily to call upon any other Runbook from within a Runbook to do work on behalf of the calling Runbook. This means you can build Runbooks which perform common functions and just call on them as required.

In building common task Runbooks, each task should be broken down into its own separate Runbook. Additionally, the design of the Runbook should be kept as generic as possible to enable re-usability. Orchestrator allows you to pass parameters into a Runbook via the ‘*Initialize Data Activity*’ and then the Runbook can do its operations and pass the results back to the calling Runbook via the ‘*Return Data Activity*’.

Tip 4. Link colours & labels

Any Link with Logic in it should always have a Label to explain its function and any Links with Decision Logic should always be colour coded so everyone can see a decision is made.

why? Quite simply, by introducing Link colours and Labels, it makes it much easier to read and understand.



Best Practice Key

🟩 Success Path

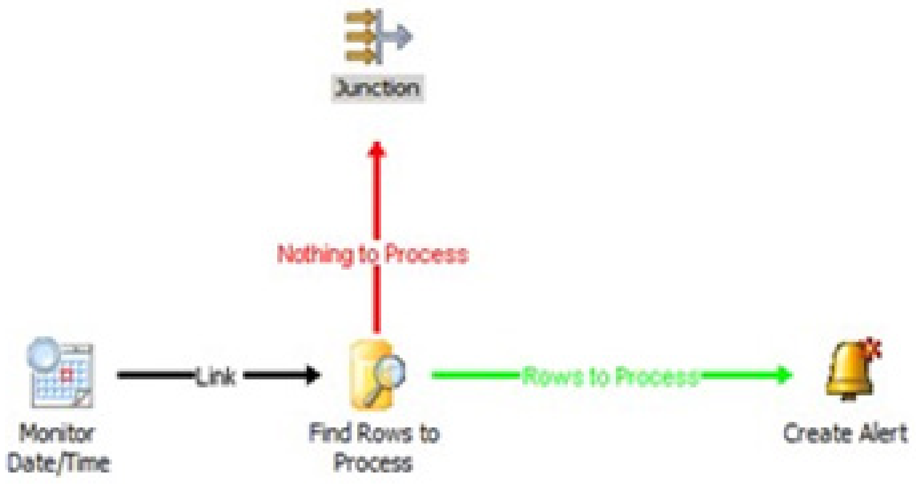
🟧 Decision Path

🟥 Fail Path

Tip 5. Add a branch for a 'Nothing to Process' batch

If you have a Runbook which looks for some state to be true and then goes off to check for this state, always add a branch to show the Runbook has a specific path to follow should there be nothing to process.

why? This makes your Runbook more easily understood by staff and helps test and debug the Runbook. By adhering to these guidelines, you are helping to make your Runbook fool proof.

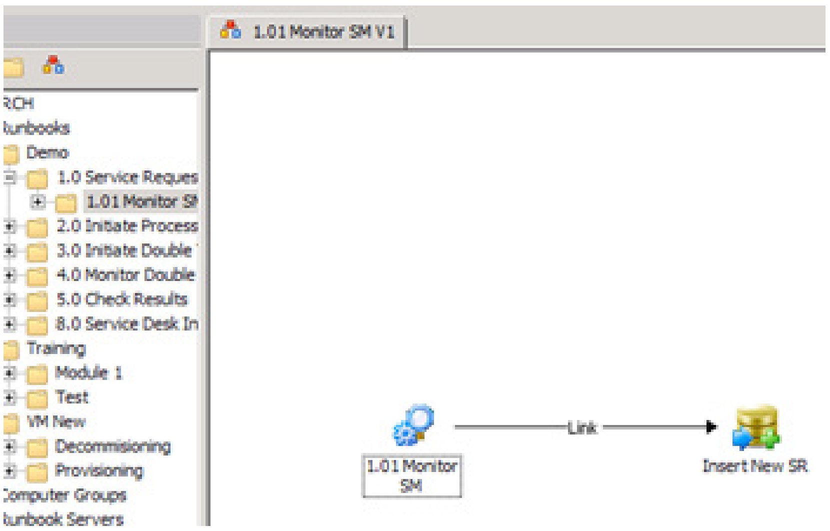


Tip 6. Give activities & runbooks meaningful names (avoid defaults)

To further support the administrator managing the Runbooks, always give Runbooks and Activities meaningful names and avoid sticking with the standard defaults provided by Orchestrator.

why? This will help others reviewing the Runbook to immediately view and understand what it is trying to do.

Every Runbook name should start with a number which reflects where it occurs in the end to end process flow and end in a version number. A best practice naming convention example would look like this: **1.01 Monitor SCOM V1**. This will provide the administrator with greater and clearer information on the process and when it stops.

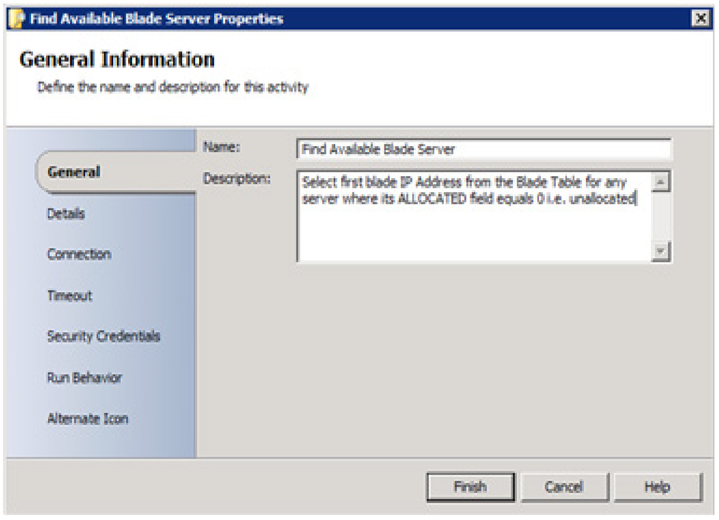


Furthermore, descriptions should be included for Activities to explain processes in more detail. Very often, this type of information is skipped but it a worthwhile inclusion particularly when new people take over the management of Runbooks who were originally designed by others who have since moved or left the organisation.

Tip 7. Include activity descriptions

If you put Activity Level Looping on an Activity or are using a Select Rows Activity always write in the description field of the Activity of the logic in use.

why? By incorporating this feature, you are supporting re-use. At a later date, when you come back to review the Runbook, it will aid memory on exactly what the Activity is supposed to be doing.


Find Available Blade Server Properties

General Information

Define the name and description for this activity

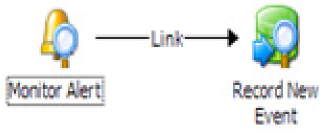
Name: Find Available Blade Server

Description: Select first blade IP Address from the Blade Table for any server where its ALLOCATED field equals 0 i.e. unallocated

Tip 8. Keep monitor runbooks very short

always keep the workflow very short for Runbooks which are monitoring an enterprise tool or the like. Grab the event and write it to a Microsoft SQL Server Database and use a Database Activity to process it.

why? By keeping the workflow very short, you will maximise performance. It is better to grab the event and write it to a Microsoft SQL Server Database and use a Database Activity to process it. This way, there is less chance of Orchestrator missing a new event because it is still busy processing the last set of events. It is highly likely that there are many tools being monitored therefore this design will better enable a more efficient method for monitoring, queuing and processing.

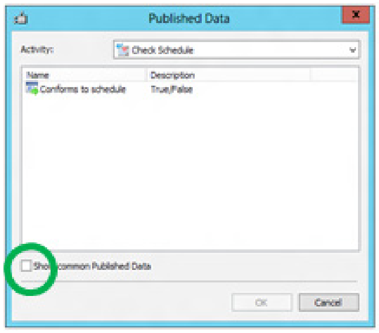


Tip 9. Create a robust error reporting routine

At some point, you will be required to create an error reporting routine from within Orchestrator to report Runbook errors back to an Event Management system. One of the best ways to do this is to create a 'Common Handler Runbook' for Error reporting which uses an Integration Pack, such as SCOM 2012 IP, to create an Alert directly in SCOM to inform users that there has been an error.

In the SCOM Alert, you always need to include details of the error.

why? This will make it useful and context sensitive to your users. This means you need details such as the Runbook Name, Activity that failed, Runbook Server etc. You can get this information easily by using the 'Common Published Data' available with every Activity. If you 'Right Click' and 'Subscribe' to the 'published data' of any 'Activity' you will see the 'Show common Published Data' Tick Box. By selecting this option, you will obtain lots of useful Published Data.



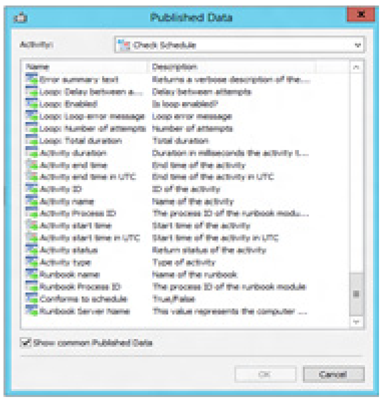
Published Data

Activity: Check Schedule

Name	Description
Conforms to schedule	True/False

☐ Show common Published Data

OK Cancel



Published Data

Activity: Check Schedule

Name	Description
Error summary text	Returns a verbose description of the...
Loop: Delay between a...	Delay between attempts
Loop: Enabled	Is loop enabled?
Loop: Loop error message	Loop error message
Loop: Number of attempts	Number of attempts
Loop: Total duration	Total duration
Activity duration	Duration in milliseconds the activity t...
Activity end time	End time of the activity
Activity end time in UTC	End time of the activity in UTC
Activity ID	ID of the activity
Activity name	Name of the activity
Activity Process ID	The process ID of the runbook modu...
Activity start time	Start time of the activity
Activity start time in UTC	Start time of the activity in UTC
Activity status	Return status of the activity
Activity type	Type of activity
Runbook name	Name of the runbook
Runbook Process ID	The process ID of the runbook module
Conforms to schedule	True/False
Runbook Server Name	This value represents the computer ...

☑ Show common Published Data

OK Cancel

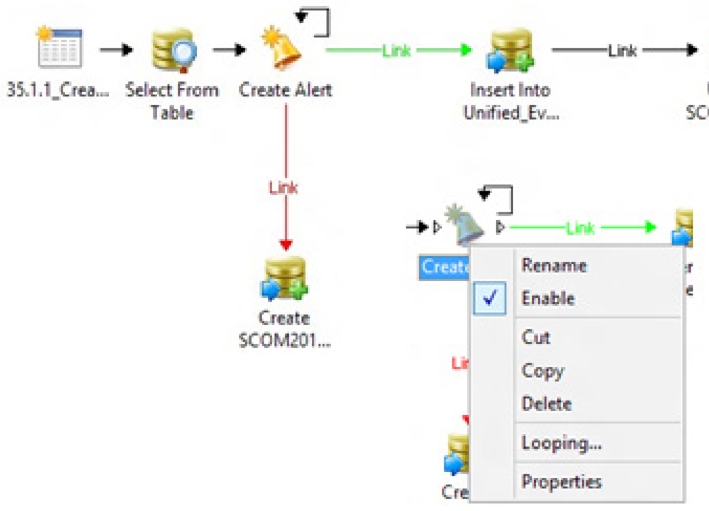
Tip 10. Use loops for resiliency

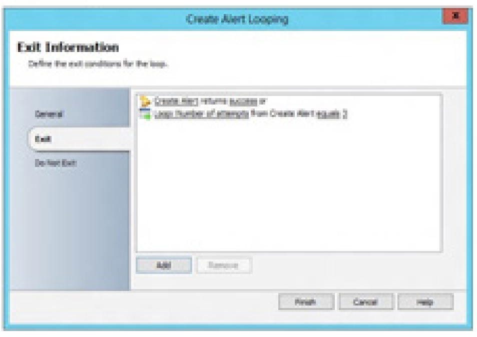
In Orchestrator, it is a good idea, when interacting with a System via an Integration Pack, to always try a few times to complete your task before throwing an error in your Runbooks.

why? It is common knowledge that API (Application Programming Interface) to Enterprise Management systems can be notoriously temperamental.

For example, when creating a SCOM Alert, try three times to create an individual Alert before throwing an error. The moto here is if you don't succeed, try, try and then try again.

You can right click on an activity, select "Looping" and then setup your looping criteria. Loop: Number of attempts is an item of Common Published Data.





Create Alert Looping

Exit Information

Define the exit conditions for the loop.

General
Exit
Do Not Exit

Create Alert returns success or
Loop: Number of attempts from Create Alert equals 3

Add Remove

Finish Cancel Help

Tip 11. Don’t use counters, use variables sparingly & keeps connection names generic

Do not use Counters in Orchestrator as they are global Counters and can therefore be modified by any Runbook at any time.

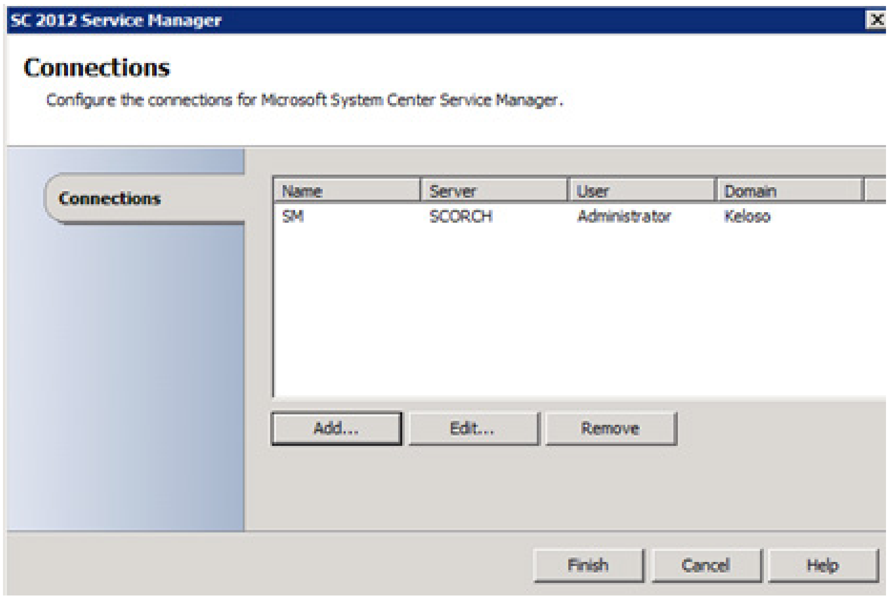
why? Basically, you cannot trust Counters as they are not reliable. You cannot rely on their value at any one point unless you are running in a Single Thread. This defeats a fundamental benefit of Orchestrator - parallel execution and multi-threading.

Variables in Orchestrator are also global. always use these sparingly; they can be ideal for database server names, table names etc. which will be used in many Runbooks. They can now be encrypted so are good for service account passwords.

why? When a Runbook is exported, all the Variables in Orchestrator are exported, not just the ones in use in the Runbook, therefore, it is very easy to contaminate your installation with variables gained from imported Runbooks. Use with caution.

Keep Connection Names in the options tab of Orchestrator generic i.e. BEM not kelvsr13-10.10.10.1-BEM.

why? When you export Runbooks from Test / Development Orchestrator into Pre- Production / Production Orchestrator, all the Connection settings are exported. If all instances use the same Connection name for their copy of the target tool when Orchestrator Imports the workflow, it will say “A connection with this name already exists, would you like to use the existing details or overwrite with the new details.” Confirm you want to use existing details and Orchestrator will automatically remap any Runbook using that connection to point to the new target system without any manual effort.


SC 2012 Service Manager

Connections

Configure the connections for Microsoft System Center Service Manager.

Name	Server	User	Domain
SM	SCORCH	Administrator	Keloso

Overall, the KISS - Keep it Short and Simple - principle applies here. Additionally, generic is better than specific.

Tip 12. Don't use text files for data repository

Text files can be read by many processes in parallel but can only be written to by one process at a time. Using a Text File means your Runbooks must run sequentially if they are to all write to the same text file.

You make Orchestrator a single threaded process. Orchestrator is a parallel processing tool.

Tip 13. Implement orchestrator with another SQL runtime database

Kelverion implementations always use a Runtime Database to drive a solution.

Why?

A. Data reliability and persistency.

If you do a straight monitor event, populate second Event management tool, you are relying on a daisy chain of Runbooks to complete the process, each passing the Event data from one Runbook to the next. If any one of the Runbooks fails then data on the whole event is lost as the information on the published databus is thrown away on failure. There is no way to recover or retry that event as the data is not persisted through the process on failure. Also, if many Events are picked up by a Monitor at the same time and passes through a daisy chain process, there is a good chance that the Monitor Runbook will still be processing the last lot of Events when the Monitor Activity reaches its time threshold and goes off to find the next lot of Events. This can result in strange behaviour in a Monitor Runbook's published data. The best solution is to keep your Monitor Runbook very short, Monitor Events and record direct to database, end Runbook. In this way, Orchestrator will have finished all the Old Events before the next monitor session.

B. Bi-directional event management interface.

If you want to implement a bi-directional interface (i.e. event is closed in second system, causing event in first system to be closed, and vice versa) you need a correlation between Event ID in first System to Event ID in second System. Most Event Management tools do not have a field in an Event to hold other Event Management Tool Event IDs. They might have incident Ticket IDs, but this is for the Number of the Service Desk Incident Ticket raised for this Event and is already in use for this. You can customise your Event Management systems to add an extra field but this is often a complex and costly activity. Therefore for ease of implementation, we recommend you export this mapping table into a database table.

C. Use a database solution for batch processing.

If you have a large number of Events to process continuously, then using a database driven solution provides performance improvement as you can batch process Events together. A number of Runbooks can each select a batch of Events and process these together in parallel, in effect multi-tasking. In a daisy chain process you are limited to processing Events in a more singular method as the next Runbook cannot start until the previous Runbook has finished.

D. Scalability.

Event Storm - In the scenario where there is a major failure and many Events are generated, there is a strong possibility that the Event Management System API used by the Monitor Activity becomes completely swamped and some Events are not passed across to the Monitor Activity. If you are not recording which Events Orchestrator has picked up you have no way to tell if any are missed. If you are persisting Events into a database then you can have a "sweeper" Runbook which periodically goes out to the first System and pulls back all the Events on the system created since the last time the "sweeper" ran. You then compare the Event IDs gathered against the persisted Events in the database. Should you find that an Event is not in the database and got missed, you can then feed this New Event back into the System as if it got picked up by the Monitor Activity.

Therefore, under Event load conditions, all Events will arrive at the second system; possibly a little later than they were generated, but at least they were not missed altogether.

D. Use a Database Solution to ensure Runbook Extensibility.

Once you have SCOM to BEM operating, you may choose to extend or change the solution by;

Turning the process on its head and go from BEM to SCOM

Add automatic Incident Ticket Creation via Orchestrator

Start to perform automatic Event diagnosis and then automatic remediation

Having a database driven solution allows you, very easily, to add capabilities to your Process flow, as you have a central source of data from which any Runbook can feed and record results.

If you have a daisy chain point-to-point process you are very limited as to what you can amend and change without having to rebuild your entire solution from scratch.

Why else might you want to use a runtime database?

- You want run time variable in a Runbook?
- Dynamic input data?
- Look up logic?
- Audit History?
- Struggling to Parse Data or Complex Decision Logic?
- Got XML/CSV as an input and got to manipulate the data?
- Need to produce a report on how often something happens or based on data from many sources?

Tip 14. Always deploy the Kelverion Integration Pack for SQL Server

Why? Automatically builds and executes the necessary commands without the user having to write or understand SQL therefore saving time, training investments and eliminating the chance of mistakes. It also simplifies Runbook design by automatically mapping table columns to Orchestrator input properties, filters and published data items.

Select Rows	**Insert Rows**	**Delete Rows**	**Update Rows**	**Run Procedure**	**Run Query**	**Monitor Rows**

When adding a database alongside Orchestrator there needs to be an efficient connection mechanism. The existing Orchestrator Activities create a connection for each transaction and close it after use, even if there is more data to process to the target table. In scenarios of heavy database interaction, this puts an unsustainable load on the database access authentication method and, for large updates, this can increase the workflow run time. An ability to maintain the connection is required to increase performance.

In order to improve ease of use, a key requirement was to remove the need to understand and build SQL statements that would be executed in an activity. Once executed, the Activity produces the published data items without any further processing steps or activities. This simplification of workflows is powerful for both users experienced in Orchestrator workflow development and those new to Orchestrator. The resulting benefits are to remove any potential errors in the SQL statements, faster workflow development and simpler workflows to maintain and support.

Tip 15. Document your solution - Use Kelverion Runbook Surveyor

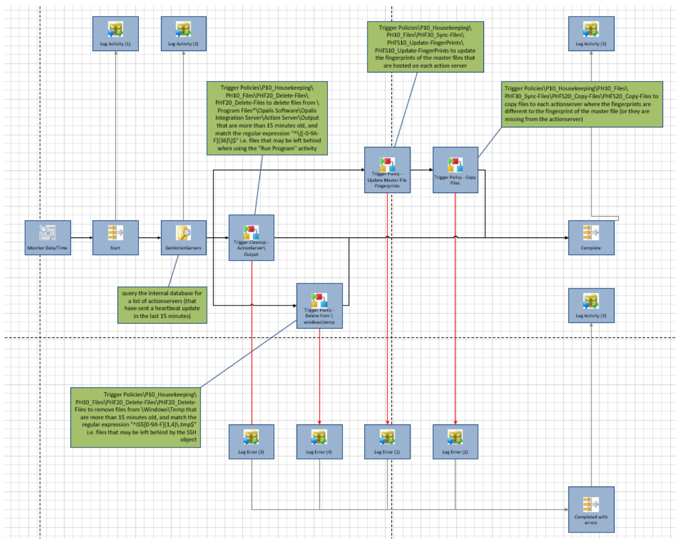
Once the solution is built, how do you document it?

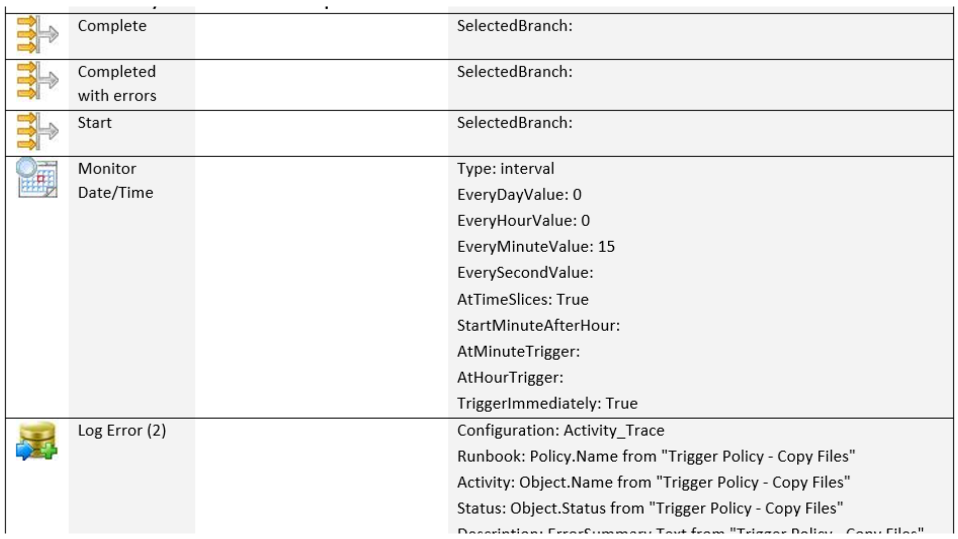
When you have created a series of Runbooks in System Center Orchestrator that delivers a business solution, you will be required, at some point, to document your Runbooks. Currently, this means taking multiple screenshots and laboriously recording each Activity configuration by hand so that, later on, someone can understand what you built and replicate the Runbooks if something catastrophic happens to your environment. This documentation process often takes longer than the actual Runbook design and creation phase.

What is required is an automated solution to recording Runbooks, and their configurations, and the Kelverion Runbook Surveyor is the answer to this problem.

The Runbook Surveyor enables users to automatically document their solution. When you have created your Runbooks, you can point the Runbook Surveyor at your Orchestrator environment and it will automatically produce a series of Visio Diagrams and a Word Document.

These outputs diagram and document all your Runbooks and show how each one is configured and connected to other Runbooks.


Log Activity (1)
Log Activity (2)
Trigger Policies\P10_Housekeeping\PH10_Files\PHF30_Sync-Files\PHFS10_Update-FingerPrints\PHFS10_Update-FingerPrints to update the fingerprints of the master files that are hosted on each action server
Log Activity (3)
Trigger Policies\P10_Housekeeping\PH10_Files\PHF20_Delete-Files\PHF20_Delete-Files to delete files from \Program Files*\Opalis Software\Opalis Integration Server\Action Server\Output that are more than 15 minutes old, and match the regular expression "^\{[-0-9A-F](36)\}$" i.e. files that may be left behind when using the "Run Program" activity
Trigger Policies\P10_Housekeeping\PH10_Files\PHF30_Sync-Files\PHFS20_Copy-Files\PHFS20_Copy-Files to copy files to each actionserver where the fingerprints are different to the fingerprint of the master file (or they are missing from the actionserver)
Trigger Policy - Update Master File Fingerprints
Trigger Policy - Copy Files
Monitor Date/Time
Start
GetActionServers
Trigger Cleanup - ActionServer\Output
Complete
query the internal database for a list of actionservers (that have sent a heartbeat update in the last 15 minutes)
Trigger Policy - Delete from \windows\temp
Log Activity (3)
Trigger Policies\P10_Housekeeping\PH10_Files\PHF20_Delete-Files\PHF20_Delete-Files to remove files from \Windows\Temp that are more than 15 minutes old, and match the regular expression "^ISS[0-9A-F]{1,4}\.tmp$" i.e. files that may be left behind by the SSH object
Log Error (3)
Log Error (4)
Log Error (1)
Log Error (2)
Completed with errors

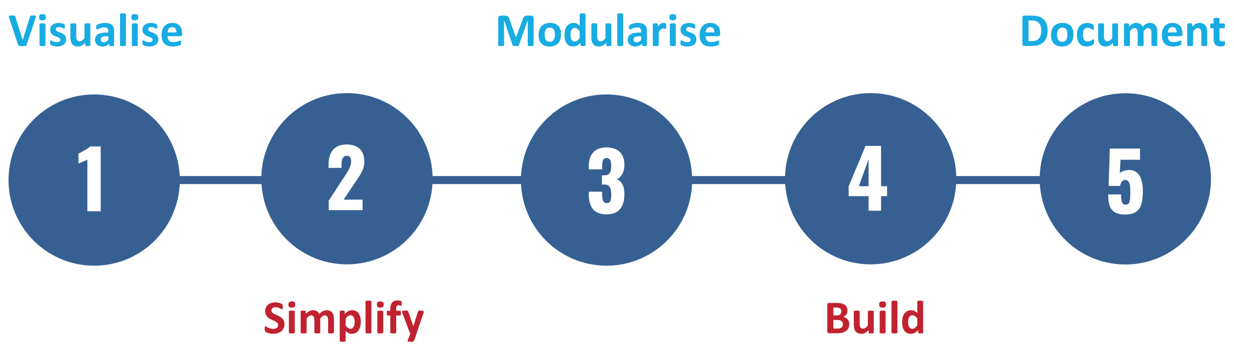


	Complete		SelectedBranch:
	Completed with errors		SelectedBranch:
	Start		SelectedBranch:
	Monitor Date/Time		Type: interval EveryDayValue: 0 EveryHourValue: 0 EveryMinuteValue: 15 EverySecondValue: AtTimeSlices: True StartMinuteAfterHour: AtMinuteTrigger: AtHourTrigger: TriggerImmediately: True
	Log Error (2)		Configuration: Activity_Trace Runbook: Policy.Name from "Trigger Policy - Copy Files" Activity: Object.Name from "Trigger Policy - Copy Files" Status: Object.Status from "Trigger Policy - Copy Files" Description: ErrorSummary.Text from "Trigger Policy - Copy Files"

Summary

In summary, IT professionals can avoid common pitfalls during the implementation of an Orchestrator solution by following the best practice outlined in this document. Follow our Top 15 Tips in designing an Orchestrator Runbook and ensure success and efficiency in your IT Process Automation projects.

The Kelverion best practice approach to IT Process Automation, and the use of System Center Orchestrator, involves following the below proven steps when building Runbooks.



Orchestrator is all the more powerful when implemented with its own run time database. Without a runtime database, you will limit what can be achieved. However, with a database, there are virtually no limits to Orchestrator's potential.

Failure to follow the steps in this paper could result in errors, poor service, inefficient processes being automated, increased costs and lack of IT process governance and control.

Remove the frustration from your IT Process Automation projects and follow Kelverion's best practice advice and approach.

Further Reading

For further information on System Center and Orchestrator, you can read more online at Microsoft TechNet:

https://technet.microsoft.com/en-us/library/hh237242.aspx

Working with Kelverion

Experts in Cloud, On-Premise and Hybrid automation, Kelverion provide solutions and integrations that remove the manual process tying up IT staff; transforming the productivity, efficiency and supportability of IT service automation. Our products utilise and enhance the power of Microsoft Azure and System Center Orchestrator.

If you would like more information on Kelverion products and services please visit our website or give one of our automation experts a call.

US +1 289 801 0559 UK +44 203 875 8035

www.kelverion.com info@kelverion.com

Kelverion