International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2019): 7.583

SAP Cloud Integration - An Overview, Best Practices, and Implementation Steps - Part 2

Deepak Kumar[1], Maha Bhageshwara Raju Kesaboina[2]

Wilmington, USA
Email: *deepak3830[at]gmail.com*

Hyderabad, India
Email: *bhageshwar.raju[at]gmail.com*

Abstract: ***SAP CPI, also known as SAP Cloud Platform Integration is a cloud-based integration platform provided by SAP. It allows for connectivity, between applications, systems, and services both within and outside of an organization. With SAP CPI businesses can simplify the integration process. Optimize data flow and communication between software solutions. It facilitates the exchange of data across environments promoting interoperability and enhancing collaboration efficiency. Additionally, SAP CPI offers built-in integration iFlow and adapters to reduce the complexity of integration projects and speed up deployment timelines. Overall SAP CPI plays a role, in the SAP ecosystem by enabling the creation of agile business processes. In this paper, we will discuss SAP CPI's capabilities in connecting diverse applications, facilitating data flow, supporting different integration scenarios, and leveraging pre-built content and adapters. It will provide a step-by-step guide or overview of the fundamental steps involved in implementing SAP CPI. This paper will cover the initial setup, configuration, and the process of establishing connections between different systems. The aim is to offer a practical understanding for readers who might be considering or undergoing the implementation process. This paper will also address the monitoring aspect of SAP CPI, emphasizing the importance of keeping track of integrations and workflows. Insights into monitoring tools, key performance indicators, and best practices for ensuring the ongoing efficiency and reliability of integrated systems will be elaborated.***

Keywords: SAP, SAP CPI, SAP Cloud Integration, SAP On-premice to SAP cloud integration

1. Introduction

SAP CPI – SAP Cloud Platform Integration is a powerful cloud-based platform that enables seamless integration between cloud applications and other SAP or non-SAP cloud or on-premises applications. SAP CPI uses Apache Camel in its integration framework. Apache Camel is an open-source framework that offers a variety of pre-built components and patterns for creating enterprise integration solutions. SAP CPI utilizes Apache Camel to make it easier to develop integration flows and to improve the platform's flexibility and extensibility.

Integration Flow (Iflow) – An iFlow is a graphical representation of the complete integration process within SAP CPI. It outlines the steps messages must take as they travel from a source system to a target system, including data transformations and mappings along the way.

API stands for Application Programming Interface. It's a set of protocols, tools, and definitions that enable different software applications to communicate with each other. APIs determine how software components should interact, allowing developers to access specific features or data from a service, library, or application without having to comprehend its internal workings.

Groovy Script – A Groovy script is a script written in the Groovy programming language. Groovy is an object-oriented scripting language that is dynamically typed and seamlessly integrates with Java. It is often used as a scripting language for Java applications and provides concise and expressive syntax. Pre-requisite: To understand this paper thoroughly prerequisite is SAP Cloud Integration – An Overview, Best Practices, and Implementation Steps – PART 1.

2. Discussion

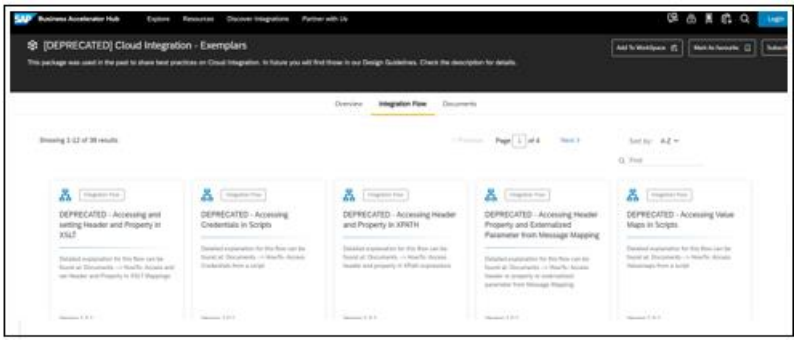
SAP CPI Standards and Best Practices - The purpose of this paper is to establish guidelines for integration developers who will be using the SAP Cloud Platform Integration Service to develop integrations for projects. This blog provides end-to-end CPI standards and best practices that build upon the guidelines and insights shared by experienced CPI developers worldwide. Additionally, it incorporates our real-world experience from numerous projects and the recent distribution of millions of API messages through CPI.

3. API Business Hub Package

The SAP API Business Hub Package is a collection of APIs, integration content, and resources available on the SAP API Business Hub. The SAP API Business Hub is a central repository provided by SAP that offers a catalog of APIs, pre-built integration packages, and other content. These packages are intended to help developers create integrations between SAP solutions and other systems more easily and quickly. The packages CPI Cloud Exemplar package SAP CPI Integration Design Guidelines and SAP CPI Troubleshooting Tips include not only detailed documentation or FAQs but also working samples and templates that help you:

- Understand how you can perform basic tasks
- Identify the common pitfalls while designing your flow
- Discover optimal ways of modeling an integration flow
- Determine techniques to achieve a better memory footprint

- Define what to keep in mind in order to create performant integration flows
- Solve commonly known errors with a ready solution



4. Development Guidelines

SAP CPI offers development in two different environments namely Eclipse and Web IDE.

Eclipse is a widely used integrated development environment (IDE) that is primarily Java-based but supports various programming languages through plugins. It provides a comprehensive platform for software development, offering tools and features for coding, debugging, and version control. Eclipse is open-source and has a large ecosystem of plugins and extensions that enhance its functionality. In the context of SAP Cloud Platform Integration (CPI), developers often use Eclipse with the SAP Tools for Eclipse (SAP TDI) plugin to facilitate the development and deployment of integration flows.

The SAP Cloud Platform Integration (CPI) Web UI refers to the web-based user interface provided by SAP for designing, configuring, monitoring, and managing integration processes within the SAP CPI environment. The CPI Web UI is accessible through a web browser and serves as the primary interface for users involved in integration development and operations.

Here are some guidelines to consider when designing the layout, for integration flows which can help simplify maintenance;

- Try to avoid overlapping sequence flows
- Keep the sequence and message flow connectors straight as possible avoiding any twists or turns.
- Make sure not to have process steps overlapping. If you have a lot of process steps consider expanding the canvas and arranging them neatly.
- For complex logic break them down into modules that are easier to understand. Move the logic of each module into its subprocess. Give the subprocess an appropriate name that describes its operation.
- Avoid mixing transformations in a script or subprocess. Each sub-process should only contain the logic for one function.
- Only assign the XML message to a header or property if necessary. Once you're done with it make sure to clear it.
- Always maintain a flow direction from left to right in your design. The sender should always be on the side. The receiver is on the right side.
- It is advisable to keep the number of steps, in an integration flow limited to 10. Utilize the integration process within those steps to break down integration flows as it helps reduce the total cost of ownership and facilitates easier maintenance.

Here are some recommended practices when multiple developers are working on projects, in SAP Cloud Integration. These practices ensure that developers do not make changes to packages and artifacts created by developers.

- To indicate that a package is still a work in progress enter "WIP" in the Version field. In the Owned By text box specify your name or team name to indicate who should be contacted by teams or developers if they want to make changes to this package's artifact.
- These steps will assist developers in coordinating and communicating about editing the artifacts within the package. After completing the edits save the package as a version. This will update the version number, from WIP to the number.

5. Performance Guidelines

It is advisable to avoid using Bulk Extracts whenever possible. When designing processes it is important to utilize change pointers and deltas of transferring data in bulk. We should only transmit the data by avoiding transmitting unnecessary information. Transmitting huge amounts of data can have an impact, on the performance of CPI systems, external partner systems, networks and ultimately affect end users. Therefore when designing interfaces it is crucial to optimize the transfer of data. Additionally consider exploring tools such, as SAP Data Services, CPI Data Services or Smart Data Integration if you need to extract data from source systems and transform it before loading it into target systems.

API(S) generally are designed for low volumes. We should use other adapters like JMS, Files, JDBC, etc for large volumes or CPI Data Services/Smart Data Integration if we have to extract, transform, and distribute large amounts of data between many systems. The following are the performance guidelines to optimize IFLOW when you are integrating systems using API endpoints.

Session Reuse: Creating a login session is a task that requires a lot of resources. Therefore, it is advisable to reuse login sessions instead of creating a new session for each HTTP transaction or each page of paginated data. When making API

calls, a user session is created, which is also a resource-intensive process. If there are a large number of API calls, it can cause stress on the server and significantly impact response time. To avoid this, you can enable a session on the integration flow to reuse the session. You can find more information about this in the blog post linked below.

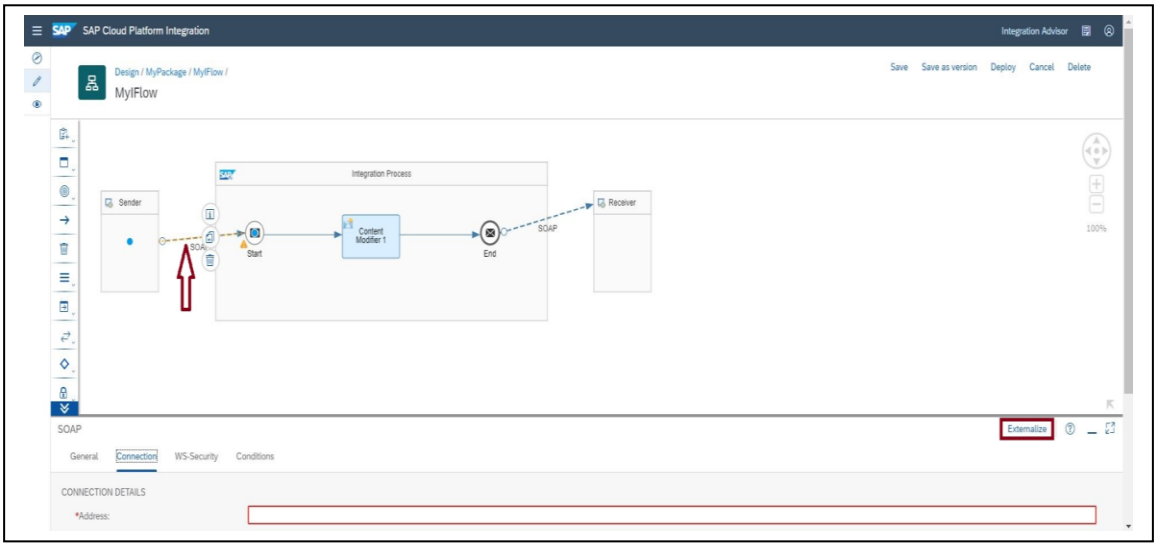
API Retry Mechanism: Implement your retry logic properly, Retry logic can help to recover transactions that failed due to internet connectivity or backend server issues, but retry must be done with care based on the type of HTTP error. Only return a maximum number 5 of times before abandoning your task.

OData API Performance Optimization Recommendations: It's important to optimize the size of your batch requests when using the OData API, as it can only return up to 1000 records on a single page. To achieve the best performance, your batch sizes should be as large as possible. However, for more complex transactions, you may need to reduce the size to avoid HTTP timeouts. Instead of pulling many records one at a time, you can use batch or $filter to retrieve multiple records at once. When using the $expand statement to request master-detail data, it's important to avoid using it to join a large number of tables, as this can result in poor performance. SAP recommends fetching the master data by batch first, and then using a content enricher or expand to get the details. It's also important to avoid querying properties or expanding entities that you don't need or use.

Splitter: We should consider improving interface performance by exploring options for parallelizing the process within the CPI tenant. To activate parallel processing, you can use the "Splitter" step, which splits the content into packages containing 1000 entries per package. You can enable parallel processing by selecting the checkbox at this step. Additionally, the splitter step has concurrency that can restrict the number of concurrent parallel processes that CPI can trigger in the SAP destination systems.

6. Externalizing parameters

Integration Developers build their integration flows using the SAP Cloud Platform Integration tools on their development systems and once the development is complete, they move them to the test and production systems. During this development phase, they realize that the same integration flow may not work across different systems and would require changes in the configurations of adapters or flow steps. To overcome this situation, they use the externalization feature offered by SAP Cloud



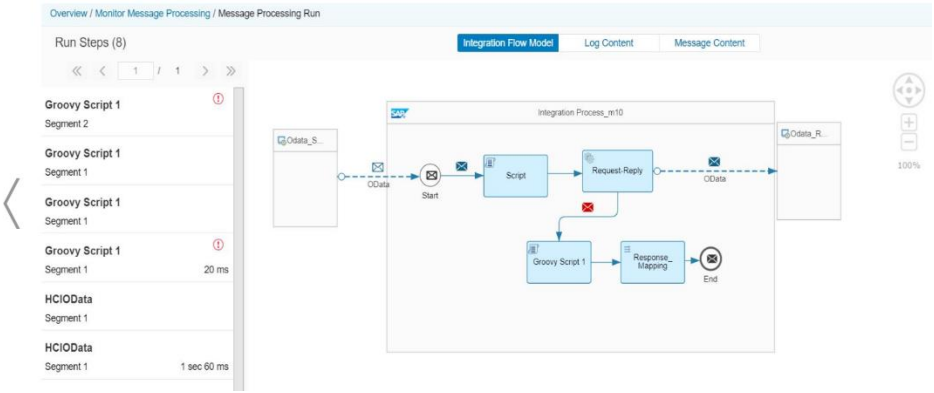
Platform Integration tools.

The configuration view allows to configuration of the externalized parameters of adapters and/or flow steps without editing the integration flow. The value configured from the configuration view is called the configured value of a parameter. The Configured value of a parameter is centric to the tenant/landscape. Below are the recommended steps for externalization:

- Externalize all the fields of integration flow that you envisioned and provide the appropriate Default values.
- Ensure not to provide the tenant/landscape-specific value as a Default value of parameters.
- Validate the Default value of parameters through validation checks. Save of integration flow will run validation checks.
- Always provide tenant/landscape-specific value in the Configure View.
- Before downloading the Integration flow or exporting the content package, always leverage the benefit of Externalized Parameter View to compare the Default and Configured value of parameters for quality assurance. Update the parameter's default value from the externalized parameters view or externalization editor for any correction.
- Download the integration flow with “Default Values Only” if you wish to import the integration flow in the target system which has a different tenant/landscape configuration.
- Download the integration flow from the source system with “Merged Configured and Default Values” if you wish to import the integration flow in the target system with the same tenant/landscape configuration.

7. Monitoring

SAP CPI offers pre-built monitoring features and thorough auditing of message processing at every stage of its life cycle. This helps support teams to swiftly resolve any problems that arise. It is advisable to enable payload tracing solely in testing systems and activate it in production systems based on the logging configuration of the IFLOW. This optimizes system performance unless it is necessary to have it on in the production environment. Troubleshooting always begins with the Monitor Message Processing feature and multiple search options are available to retrieve the message you are interested in. After having selected the specific message processing log, you see the first details of the message processing. In this case, we are investigating the error that occurred in the message processing. The Error Details are giving you a first technical hint on what happened. To further analyze this issue, click on the Log Level you are seeing in the detailed view of the message processing log. In this specific case, the message was sent in a log-level Trace.



On the left-hand side, you see the master list containing the Run Steps. A click on one of the elements will affect the views on the right-hand side, the Integration Flow Model, the Log Content, and Message Content views that are selectable by clicking on the respective tabs. The visible log steps are the logical steps that match the configured integration message flow model steps, in this case, ODataSender, Script, Groovy Script, and HCIOData. Other steps are not visible, since the occurring error stopped the message processing. If the integration flow model is still deployed it will be loaded and shown as the default page. The Run Steps table on the left side is the starting point for all views. The indicates the steps where an error occurred. The table is sorted by occurrence, the newest entries being at the top. Errors are most likely shown in one of the last steps and that is what we are looking for in this case.

8. Transport Mechanism & Naming Conventions

SAP provides below mechanisms to transport CPI objects:

- Manual Export and Import
- CTS+ (Content Transport Service)
- MTAR Download
- Transport Management Service

It is recommended to use CTS+(If the customer has a CTS+ System)/TMS transports for transporting objects from one environment into another environment. CTS+/TMS Transport should contain package name and version number and change the description for each transport for customers with complex integration landscape and who have solution managers in the to-be landscape. Customers can evaluate SCP TMS and FIGAF for small to medium-complexity integration landscape or if customers don't have a solution manager on the roadmap.

CPI Transport Naming Conventions : <CR Number> <CR Description><Package Name/Artefact Name> <Version Number>

9. Conclusion

The this discussion we have included an overview of the SAP CPI architecture, which involves understanding how different components within SAP CPI interact to facilitate seamless integration between various systems and applications. This paper covers aspects such as message processing, connectors, runtime, and the overall structure of the integration platform. This paper covers how SAP CPI enables users to connect different systems, orchestrate processes, and manage the flow of data. Integration capabilities may include pre-built connectors, data mapping, content-based routing, and support for various protocols. Also, we have provided a glimpse into the user interface of SAP CPI, including key screens such as the homepage, design tab, monitoring tab, and settings tab. This gives users an understanding of where they can perform different tasks related to designing, monitoring, and configuring integration processes. The upcoming sections will focus on implementation steps along with practices and a case study to offer valuable insights and guidance, for users working with SAP CPI.

Declarations

Ethics approval and consent to participate: Not Applicable
Consent for publication: All authors have consent to submit this paper to the Journal of Cloud Computing. Also, we confirm that this paper or any part of this paper was not submitted anywhere.

Availability of data and materials: Not Applicable

Competing interests: Not Applicable

Funding: Not Applicable

Acknowledgments: Thank you co-author Maha Bhageshwara Raju Kesaboina for his expertise and assistance throughout all aspects of our study and for your help in covering a few topics and reviewing the manuscript.

References

[1] "SAP Business Accelerator Hub," api.sap.com. https://api.sap.com/integrations/packages
[2] Sookriti_Mishra, "My adventure in learning CPI: Part 1 | All about SAP Cloud.," SAP Community, Sep. 10, 2019. https://community.sap.com/t5/technology-blogs-by-members/my-adventure-in-learning-cpi-part-1-all-about-sap-cloud/ba-p/13393526
[3] Sookriti_Mishra, "My adventure in learning CPI: Part 2 | Deployment Models," SAP Community, Sep. 15, 2019. https://community.sap.com/t5/technology-blogs-by-members/my-adventure-in-learning-cpi-part-2-deployment-models/ba-p/13394902
[4] Sookriti_Mishra, "My adventure in learning CPI: Part 3 | Cloud Security," SAP Community, Sep. 15, 2019. https://community.sap.com/t5/technology-blogs-by-members/my-adventure-in-learning-cpi-part-3-cloud-security/ba-p/13398951
[5] "SAP CPI | SAP Blogs," blogs.sap.com. https://blogs.sap.com/tag/sap-cpi/
[6] STALANKI, "Comprehensive SAP CPI Guide for Standards & Best Practices," SAP Community, Jan. 16, 2020. https://community.sap.com/t5/technology-blogs-by-members/comprehensive-sap-cpi-guide-for-standards-best-practices/ba-p/13457873
[7] "SAP Help Portal," help.sap.com. https://help.sap.com/docs/cloud-integration/sap-cloud-integration/sap-cloud-integration
[8] "Home - Apache Camel," camel.apache.org. https://camel.apache.org/
[9] "SAP Help Portal," help.sap.com. https://help.sap.com/docs/cloud-integration/sap-cloud-integration/integration-capabilitiesengswee, "CPI's Groovy meets Spock – To boldly test where none has tested before," SAP Community, Jan. 24, 2018. https://community.sap.com/t5/technology-blogs-by-members/cpi-s-groovy-meets-spock-to-boldly-test-where-none-has-tested-before/ba-p/13367099
[10] "Steps to learn SAP CPI, what you must know before your first project | SAP Blogs," blogs.sap.com. https://blogs.sap.com/2019/12/06/steps-to-learn-sap-cpi-what-you-must-know-before-your-first-project/
[11] "The Apache Groovy programming language - Syntax," groovy-lang.org. https://groovy-lang.org/syntax.html#_unicode_escape_sequence
[12] "Comprehensive SAP CPI Guide for Standards & Best Practices | SAP Blogs," blogs.sap.com. https://blogs.sap.com/2020/01/16/comprehensive-sap-cpi-development-standards-best-practices/
[13] "Integration Developer's Corner | SAP Blogs," blogs.sap.com. https://blogs.sap.com/2017/07/17/introduction-to-developers-corner/
[14] "SAP Help Portal," help.sap.com. https://help.sap.com/docs/cloud-integration/sap-cloud-integration/integration-flow-design-guidelines
[15] "Externalizing parameters using SAP Cloud Platform Integration's Web Application | SAP Blogs," blogs.sap.com. https://blogs.sap.com/2017/06/20/externalizing-parameters-using-sap-cloud-platform-integrations-web-application/
[16] "Enrichments of Externalization Feature in SAP Cloud Integration | SAP Blogs," blogs.sap.com. https://blogs.sap.com/2020/01/21/enrichments-of-externalization-feature-in-sap-cloud-platform-integration/?