One ACE Deployment Solution

Table of Contents

[Overview 2](#_Toc169530887)

[Deployment Diagram 2](#_Toc169530888)

[Deployment Steps 2](#_Toc169530889)

[Initial Setup 2](#_Toc169530890)

[2 Pipeline integration 3](#_Toc169530891)

[2.1 Pull Latest Code 3](#_Toc169530892)

[2.2 Merge Code 3](#_Toc169530893)

[3 Package and Docker Image Creation 3](#_Toc169530894)

[3.1 Create Deployable Unit 3](#_Toc169530895)

[3.2 Verify Package 3](#_Toc169530896)

[4 Docker Image Creation 4](#_Toc169530897)

[4.1 Build Docker Image 4](#_Toc169530898)

[4.2 Verify Docker Image 4](#_Toc169530899)

[5 Environment Compatibility 4](#_Toc169530900)

[5.1 ACE Version Compatibility 4](#_Toc169530901)

[6 Deployment Stages 4](#_Toc169530902)

[6.1 Development Environment 4](#_Toc169530903)

[6.2 Progression to Higher Environments 4](#_Toc169530904)

# Overview

This document outlines the One ACE deployment solution for integrating two or more ACE instances (their Flows, APIs, etc) Ex integrating Sapiens Digital ACE, Core System ACE (ex-CoreSuite etc). Post integration a unified docker image will be created to realise One ACE use case. The deployment process uses a structured pipeline to manage the merging, packaging, and deployment of these components, ensuring seamless integration while adhering to naming conventions to avoid collisions.

## Deployment Diagram

This diagram depicts the deployment solution for integrating Sapiens Digital flows/APIs, Core System (ex-CoreSuite etc) flows/APIs and Customer Flows/APIs within a unified package.

A diagram of a software development process

Description automatically generated

# Deployment Steps

Refer the below mentioned deployment steps.

Initial Setup

1.1. Naming Conventions

* Establish distinct naming conventions for:
  + Digital Flows/APIs
  + Core System (ex-CoreSuite etc) Flows/APIs
  + Customer Flows/APIs
  + Variables
* Ensure that naming conventions are strictly followed to avoid name collisions and overwrites within the unified package.

1.2. Git Repository Access

* Configure the pipeline to access both Core System (ex-CoreSuite etc) and DigitalSuite Git repositories.
* Ensure the pipeline has the necessary permissions to pull code from these repositories.

# 2 Pipeline integration

## 2.1 Pull Latest Code

* The pipeline will pull the latest code from the Core System (CS) Git repository containing:
  + CS Flows
  + CS APIs
* The pipeline will also pull the latest code from the Digital Git repository containing:
  + Digital Flows
  + Digital APIs

## 2.2 Merge Code

* Merge the code into a single package, ensuring that naming conventions are adhered to, preventing any naming conflicts.

# 3 Package and Docker Image Creation

## 3.1 Create Package and Docker Image

* Package the merged code into a deployable unit.
* Ensure that the package contains all the necessary flows, APIs, and configurations.
* Create a Docker image from the verified package.
* Ensure the Docker image contains all required dependencies and configurations for the flows and APIs.

## 3.2 Verify Package and docker

* Conduct integrity checks and functional verification of the package.
* Ensure that the package meets all quality standards and is ready for deployment.
* Test the Docker image to ensure it functions as expected.
* Conduct tests to validate the integration of different flows and APIs within the Docker container.

# 5 Environment Compatibility

## 5.1 ACE Version Compatibility

* Check for compatibility between different ACE5 versions used by Digital and CS flows.
* Address any compatibility issues identified during the checks.

# 6 Deployment Stages

## 6.1 Development Environment

* Deploy the Docker image to the Development environment.
* Conduct thorough testing to ensure the deployment is successful and the integrated package functions as expected.

## 6.2 Progression to Higher Environments

* Once validated in Development, promote the Docker image sequentially to:
  + QA Environment
  + UAT Environment
  + Production Environment etc
* Conduct testing and validation at each stage to ensure seamless integration and functionality.

We ensure a structured and efficient process for integrating and deploying Digital, CoreSuite (CS), and Customer Flows/APIs within a single package, while maintaining clear naming conventions and addressing compatibility concerns.