AIP Console - 2.x - Introduction

Introduction

Console 2.x brings improvements and changes, mainly designed to improve the overall flexibility of deployment and application analysis. The main changes are in the architecture and deployment areas, as listed below.

- **Console deployment**:
  - the **front-end Console** is now provided as a Linux based Docker container. This means cross-platform deployment and upgrade has been simplified (Docker is compatible with both Linux and Microsoft Windows)
  - the Console **authentication provider** has been totally restructured and now uses the open-source OAuth2 compatible **Keycloak** system. Keycloak provides local authentication, and can also interact with other enterprise authentication systems such as LDAP and SAML. This change greatly simplifies the configuration of the authentication method you choose. Keycloak is also provided as a Linux based Docker container.
  - the method for **storing settings, options and information about Nodes** has been changed from a flat-file H2 database to a PostgreSQL database (the **Node Database**), also provided as a Linux based Docker container. This database can also be used to host schemas to store analysis and snapshot data if necessary (additional standalone CAST Storage Services/PostgreSQL instances can also be used as in v 1.x and are recommended).
- **Node instance deployment** is very similar to v. 1.x: an installation of AIP Core and an Node Service are required and must be installed on a compatible Microsoft Windows server. One significant improvement, however, is that Node instances are now considered to be stateless (the applications are not attached to a specific Node instance). All Node instances register themselves in Console and use a common configuration provided by Console (stored in the PostgreSQL database provided as a Docker container - the **Node Database**). Thus, all Node instances connect to the same PostgreSQL database (the **Node Database**) to fetch settings and options and all use the same locations for delivery, deploy and common data (these locations must now always be deployed as shared folders). Node instances also share the connection settings to additional CAST Storage Services/PostgreSQL databases defined in Console for analysis/snapshot requirements (these are no longer defined when the Node service is installed as in v. 1.x). In other words, if multiple CAST Storage Services/PostgreSQL instance are defined in Console, all Node instances will be able to connect to and use all CAST Storage Services/PostgreSQL instances.
- **Application management** is similar to v 1.x, however, as outlined above, Applications are no longer tied to one single Node instance. All required storage locations such as deploy/delivery must now be configured as shared folders. Console will also automatically operate in load-balancing mode where the least used Node instance is selected from the pool of Node instances to perform the next job (analysis/snapshot etc.) - note that by default where multiple Node instances are available, Console will choose the Node instance running the most recent release of AIP Core. Therefore when creating a new Application, an Node instance is no longer defined, however, it is now necessary to define a specific CAST Storage Service/PostgreSQL instance for your Application storage requirements at this time (the application remains "tied" to this storage instance).

See **AIP Console - 2.x - Main updates and new features** for more information.

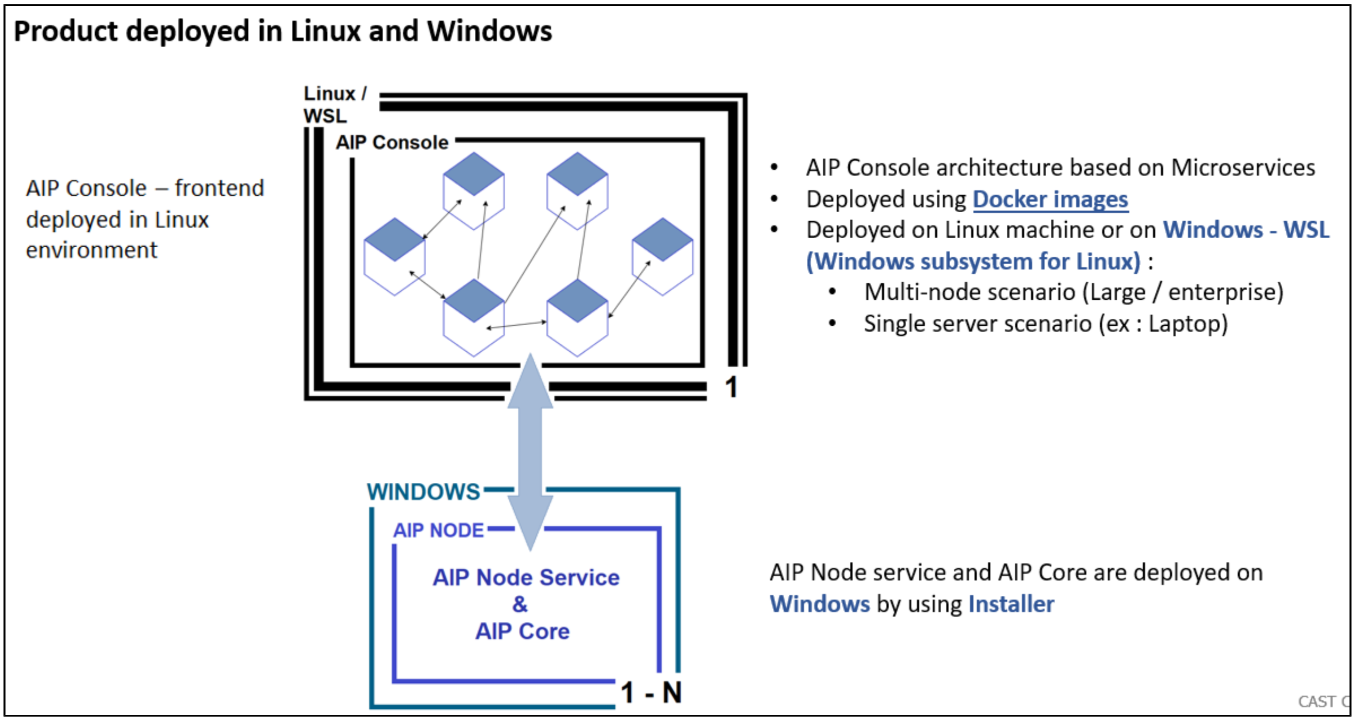
Benefits of Console 2.x

- Central configuration and stateless AIP Node instances allows applications to be detached from an Node instance. This makes it possible to add Node instances on request, removes complex routing rules, and allows to load balance the analysis/snapshot processes gracefully.
- All the front-end components are deployed as Linux based Docker containers to speed up and simplify deployment across both Linux and Microsoft Windows.
- No synchronization between Console and Node instances required. Node instances use the same PostgreSQL database (Node Database) which contains persistence information about all the applications.
- Node and Architecture Studio instances share the common data folder deployed as a shared folder, therefore there is no need to upload /download the files between the services as is the case in v 1.x.
- All the services have bidirectional access to each other through the Service Registry (deployed as a Linux based Docker container).
- No need to manually add Node instances or other services. Node instances register themselves in the Service Registry and become automatically available.
- Use of the OAuth2 compatible **Keycloak** system adds JWT token-based authentication instead of basic authentication provided in v1, removes the need to have custom tokens for Node instances and properly secures all the services.
- Embedded Dashboards are provided out of the box and require no additional manual configuration.

Current deployment limitations

- No in place upgrade from Console 1.x (same for Node instances).
- No import of Applications currently managed in CAST Management Studio / Console v 1.x - (now supported from Console 1.28/2.4 - see **Console V2 Migration Tool**)

Architecture



Product deployed in Linux and Windows

AIP Console – frontend deployed in Linux environment

Linux / WSL

AIP Console

1

- AIP Console architecture based on Microservices
- Deployed using Docker images
- Deployed on Linux machine or on Windows - WSL (Windows subsystem for Linux) :
  - Multi-node scenario (Large / enterprise)
  - Single server scenario (ex : Laptop)

WINDOWS

AIP NODE

AIP Node Service & AIP Core

1 - N

AIP Node service and AIP Core are deployed on Windows by using Installer

CAST C

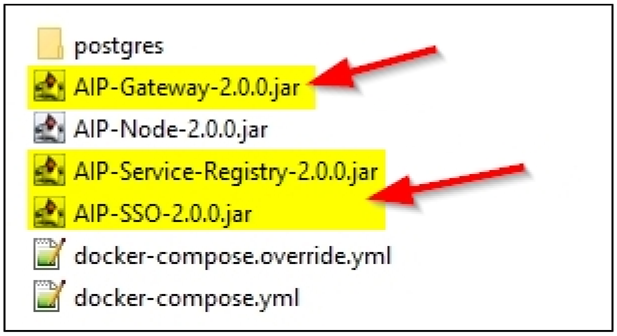
Docker containers provided by CAST

All Docker containers are Linux based.

Container	Default port	Description
aip-gateway	8081	This is the entry point to Console. It receives registered services from the Service Registry and forwards incoming requests to the required services. It also acts as a load balancer, so it can transparently handle multiple registered service instances, based on the chosen load balancing strategy.
aip-service-registry	8088, 2281	Used to register the various required services and monitor their health.
keycloak (OAuth2)	8086	The OAuth2 server (**Keycloak)** - provides authentication services for Console.
dashboards	8087	The embedded Health and Engineering Dashboards (**available from 2.0.0-beta2**).
postgres	2285	The Node Database: used primarily to store information about the Node instances. It can also be used to store Application analysis/snapshot data if required (but CAST recommends dedicated CAST Storage Service/PostgreSQL instances).

Java JAR installers

In **2.0.0-funcrel**, CAST provides Java JAR installers (along side the elements required for a Docker installation) as part of the download media available on CAST Extend (**https://extend.castsoftware.com/#/extension?id=com.castsoftware.aip.console&version=latest**):



postgres
AIP-Gateway-2.0.0.jar
AIP-Node-2.0.0.jar
AIP-Service-Registry-2.0.0.jar
AIP-SSO-2.0.0.jar
docker-compose.override.yml
docker-compose.yml

These installers are an **alternative** to a deployment on Docker, however, they currently contain some limitations/constraints and require some additional manual configuration post installation. Therefore CAST highly recommends that **Docker** is used for an enterprise deployment scenario wherever possible.

See **2.x - Enterprise mode - Installation of AIP Console front-end via Java JAR installers** for more information about this.

Prerequisites

See **Prerequisites** (for CAST Dashboard deployment) or **Prerequisites for CAST Imaging** (CAST Imaging deployment).

Installation and configuration instructions

See the following:

- **AIP Console - front-end installation**
- **AIP Node service - back-end installation**
- **Initial configuration steps - v. 2.x**

Upgrade process

See the following:

- **Console upgrade**
- **Node service upgrade**