Change Log

| **Revision** | **Submission Date** | **Affected Sections or Pages** | **Change Summary** |
| --- | --- | --- | --- |
| Initial | 12-10-2019 | All | Initial issue of document. |

Table of Contents

1 Document Overview 1

1.1 Identification 1

1.2 Purpose 1

1.3 Terminology and Notation 1

1.4 References 2

2 System Requirements 2

2.1 Software Requirements 2

2.2 Supported Browsers 3

2.3 Hardware Requirements 3

2.4 TCP Port Requirements 3

3 Administration 4

3.1 Environment Variables 4

3.2 Network Communications 4

3.3 Administration Procedures 5

4 Product Support 5

4.1 Defect Reporting Procedure 5

4.2 Points of Contact 5

A. Build Instructions 5

B. Installation Instructions 6

Table of Tables

[Table 1: Applicable JPL Rules Documents 1](#_Toc484437008)

[Table 2: Applicable MGSS Documents 1](#_Toc484437009)

[Table 3: Environment Variables 2](#_Toc484437010)

[Table 4: TCP/UDP Port Information 3](#_Toc484437011)

[Table 5: Points of Contact 3](#_Toc484437012)

# 

# Document Overview

## Identification

| **Property** | **Value** |
| --- | --- |
| Configuration ID (CI) |  |
| Element |  |
| Program Set |  |
| Version |  |

## Purpose

This document will guide a user through installing AERIE. AERIE is a new software system being developed by the MPSA element of Multi-mission Ground System and Services (MGSS), a subsystem of AMMOS (Advanced Multi-mission Operations System). AERIE will support activity planning, sequencing, and spacecraft analysis of mission operations. This guide will be updated as features are added.

## Terminology and Notation

|  |  |
| --- | --- |
| Term | Meaning |
| AMMOS | Advanced Multi-Mission Operations System |
| MGSS | Multi-Mission Ground Systems and Services |
| MPSA | Mission Planning, Sequencing and Analysis |
| YAML | Markup language |
| Docker | Software platform for building applications based on containers |
| Artifactory | Repository manager |
| CLI | Command Line Interface |
|  |  |

## References

Table 1: Applicable JPL Rules Documents

| **Title** | **DocID** |
| --- | --- |
| Software Development | 57653 |

Table 2: Applicable MGSS Documents

| **Title** | **Document Number** |
| --- | --- |
| MGSS Implementation and Task Requirements | DOC-001455 |
| Aerie Software Requirements Document (Aerie SRD) | DOC-002388 |
| Aerie Concept of Operations | DOC-002387 |
| Aerie Users Guide | TBD |
| Aerie Software Design Document (SDD) | TBD |

# System Requirements

## Software Requirements

|  |  |
| --- | --- |
| Name | Version |
| Docker | 19.x |
| \*NodeJS | 10.16.0 LTS |
| \*NPM | 6.10.0 |
| \*Open JDK | 11.x |
| \*Git | 2.x |

\*For build purposes only. Not needed for installing the application.

## Supported Browsers

|  |  |
| --- | --- |
| Name | Version |
| Chrome | Latest |
| Firefox | Latest |

## Hardware Requirements

|  |  |
| --- | --- |
| Hardware | Details |
| CPU | 2 gigahertz (GHz) frequency or above |
| RAM | 4 GB at minimum |
| Display Resolution | 2560-by-1600, recommended |
| Internet Connection | High-speed connection, at least 10Mbps |

## TCP Port Requirements

|  |  |
| --- | --- |
| Service | Port |
| Aerie UI | 8080 |
| Adaptation | 27182 |
| Plan | 27183 |

# Administration

This product is using Docker containers to run the application. There are total of five Docker containers that are internally bridged (connected) to run the application. Containers can be restarted in case of any issues using Docker CLI. Only port 8080 from the UI container is exposed to outside.

## Environment Variables

Identify the environment variables required by the software in the test and deployment environments and how to edit them. If the environment variables in the test and deployment environments are different or have different values, create two tables – one for the deployment environment and the other for the test environment. These variables should document global variables (e.g., defined in a shell) and local variables (e.g., defined in a script or configuration file). See Appendix B of AMMOS Environment Variables (MGSS DOC-001651) for more information. \*\*\*\*\* Patrick to provide the environment variables. \*\*\*\*\*

Table 3: Environment Variables

| **Variable Name** | **Default Value** | **Description** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Network Communications

The following ports are required for the application to run. The ports can be changed inside docker-compose.yaml file. Note in docker-compose file, there is an entry for each docker container which is called Port. The number that is needed to change is the first port number. The second number represents the port number within the container itself. An example of this would be ports: ["**8080**:80"]. The number that needs to be changed is the first port which is 8080.

Table 4: TCP/UDP Port Information

| **Port Type** | **Default Port #** | **Description** |
| --- | --- | --- |
| TCP | 8080 | UI Port |
| TCP | 27182 | Adaptation |
| TCP | 27183 | Plan |

## Administration Procedures

Aerie is architected and orchestrated in Docker environment. Each of the software components are packaged and run in an isolated Docker Container independently from another. Currently, we have five docker containers for Aerie.

* Aerie-UI: Hosts the web application and communicated via REST API with Adaptation and Plan Docker Containers.
* Adaptation: Handles all the logic and functionality for the model Adaptation for activity planning.
* Adaptation-mongo: Holds the data for Adaptation container.
* Plan: Handles all the logic and functionality for activity planning.
* Plan-mongo: Holds the data for Plan container.

The main three Docker Containers, Aerie-UI, Adaptation and Plan communicate to each other via rest API with the ports specified in the Docker Compose file. The database containers, Adaptation-mongo and Plan-mongo are isolated to only with the respected container, Adaptation or Plan.

# Product Support

## Defect Reporting Procedure

All defect reports should go to aerie\_support@jpl.nasa.gov.

## Points of Contact

Table 5: Points of Contact

| **Type** | **Point of Contact** |
| --- | --- |
| Adaptation | Kenneally, Patrick W, Development Lead |
| Administration | Kenneally, Patrick W, Development Lead |
| General Help | Alper Ramaswamy, Emine Basak, Product Lead |

1. Build Instructions

There is no need to build the application to be able to run Aerie. The application has been built and pushed to JPL Artifactory. Installation instructions are described in section B below.

In case there was a need to rebuild the docker images for the application, follow the instructions below.

To Build Aerie Docker Images from Source Code:

1. Install docker and docker-compose for your operating system. Link: help docs
2. Navigate to root of the downloaded source code
3. Get Aerie application source code from JPL Artifactory
4. Run docker-compose-local up to build the Docker Images locally and start Aerie
5. Navigate to localhost:8080 in your preferred browser to use Aerie
6. Installation Instructions

The Aerie project uses Docker Compose to configure and run the entire project. Docker Compose utilizes YAML configuration files which are located in the root of the project. The YAML configs include everything that is required to run Aerie with Docker and Docker Compose.

*“Compose is a tool for defining and running multi-container Docker applications. With Compose, you use a Compose file to configure your application’s services. Then, using a single command, you create and start all the services from your configuration.”*

In most cases, Docker Compose will pull down pre-built Docker containers for each of the servers which make up the AMPSA application. These containers are built at runtime, but will eventually be retrieved from our local JPL installation of DockerHub which is hosted on Artifactory. Artifactory is a repository that hosts project artifacts such as build results and Docker images.

Once the necessary containers are pulled down, Docker Compose will orchestrate the network of services and start them up.

Important note: This demo is NOT designed for production use or performance testing. It has not been optimized in any way. It WILL be slow and should not be considered for release.

To Install Aerie:

1. Install docker and docker-compose for your operating system
2. Run docker login cae-artifactory.jpl.nasa.gov:16001 from your command line or terminal
3. Get Aerie application from JPL Artifactory
4. Navigate to the downloaded folder
5. From the root of the project path run docker-compose up to start Aerie
6. Navigate to localhost:8080 in your preferred browser to use Aerie