1 S-CD-13 Containerized Development and Deployment

Artifact ID	Artifact Title		
S-CD-13	Containerized Development and Deployment		
Team		Revision	Artifact Date
BYU Mars Rover		0	22 Oct 2024
Prepared by:		Checked by:	
Nelson Durrant		Alyssa Fielding	

1.1 Revision History

Revision	Date	Made by	Checked by	Description
0	22 Oct 2024	Nelson Durrant	Alyssa Fielding	Initial upload

1.2 Purpose

This artifact describes the reasoning process behind using Docker to implement a containerized development and deployment stack on the rover.

1.3 Concept Definition

Motivation

As of October 2024, the rover uses a Orin NX to run the team's software stack, which requires a large number of separate dependencies and libraries to be installed to work. There is an extensive detailed process to set up and configure these installations on the Orin NX included in the team wiki, but these instructions are computer-specific and would have to be extensively modified to be used to configure different hardware, including development and personal computers. Modifying our software stack to use a containerized development and deployment approach will allow the team to (1) effectively develop for the rover's runtime environment, (2) easily deploy and test code, and (3) document environment dependencies efficiently.

Effective Development

Containerization is a proven method for ensuring that any two operating system environments have matching dependency and library installations, and can be used to avoid the "but it works on my computer" problem in software development. Using Docker to define (and combine) both the runtime and development environments for the rover will allow team members developing code to ensure that their contributions build and run in the exact environment the rover will use during operation and competition. This will save time debugging problems due to operating system environment differences as well as free up the rover hardware for use in other tasks besides verifying newly-developed code.

Simple Deployment

Containerization also ensures simple and easy environment deployment across a large variety of different hardware configurations. Team members simply need to install Docker on the desired computer and pull the environment image from the cloud to get the rover environment up and ready for testing and development. This greatly simplifies the process of flashing a new computer and manually installing dependencies, or translating installation documentation written for one computer to documentation that works on another machine. Containerizing the rover's runtime environment will also allow us to easily upgrade the rover's hardware or computer in the future without running into software compatibility friction – all the team will need to do is simply install Docker and pull the standardized environment onto the new hardware.

Built-in Documentation

Using Docker will also ensure that the dependencies installed into our standardized environment are tracked and documented. In Docker, a Dockerfile is used to outline the set of command line instructions that have been run to create any given image. This ensures that the exact environment used for development, testing, and production is consistent and reproducible, and must be up to date and defined for the containerized environment to build correctly. If the Dockerfile is wrong, the code won't compile or run on any new computers. By committing this Dockerfile to version control (like git), our team will be able to easily track changes to the standardized environment over time and revert to previous versions if necessary.