Annex A – Technical

Sim Manager v1.0

SYSC 5104 – Fall 2019

Prepared By:

Rob Barwell 101092722

**Contents**

[Overview 3](#_Toc26710108)

[Forward 3](#_Toc26710109)

[Assumptions 3](#_Toc26710110)

[Software Installation 3](#_Toc26710111)

[Overview 3](#_Toc26710112)

[Base Software Setup 3](#_Toc26710113)

[Container Setup 6](#_Toc26710114)

[Useful Docker Commands 6](#_Toc26710115)

[Docker Steps 7](#_Toc26710116)

[Running Software 8](#_Toc26710117)

[Initial Software Config 8](#_Toc26710118)

[Starting Containers 8](#_Toc26710119)

[Starting Development Environment 8](#_Toc26710120)

[Configuring Software 9](#_Toc26710121)

[Web Settings 9](#_Toc26710122)

[Java Microservices 10](#_Toc26710123)

# Overview

## Forward

The technical annex complements the primary paper and focuses solely on the technical implementation of Sim Manager. Please review the paper for a detailed overview of the different components and how they interact. This annex is split into 3 sections which cover installing the environment, running the software, and configuring the software.

Sim Manager was developed to be flexible both in terms of adding additional features in the future and implementation on different architectures / frameworks. This annex provides the reader an overview of a specific implementation of Sim Manager. However, the reader can choose to use any architecture / framework they choose. For example, instead of Apache httpd the reader could choose to use Internet Information Services (IIS) for the web server.

The current software architecture was primarily designed and tested on the Windows platform. All scripts can be adapted to any architecture and most were translated to Linux as part of the migration to containers.

## Assumptions

It is assumed the reader is familiar with the following technologies:

1. Containerization / Virtualization;
2. Java
3. C++
4. Full Stack Development (specifically PHP and JavaScript)
5. Libraries: D3, jQuery, Spring Boot (Lombok)
6. Cygwin
7. Databases
8. GIT
9. Maven

An extension of this work in the future could include adding the docker containers to docker hub and reduce the technical requirement for the user.

# Software Installation

## Overview

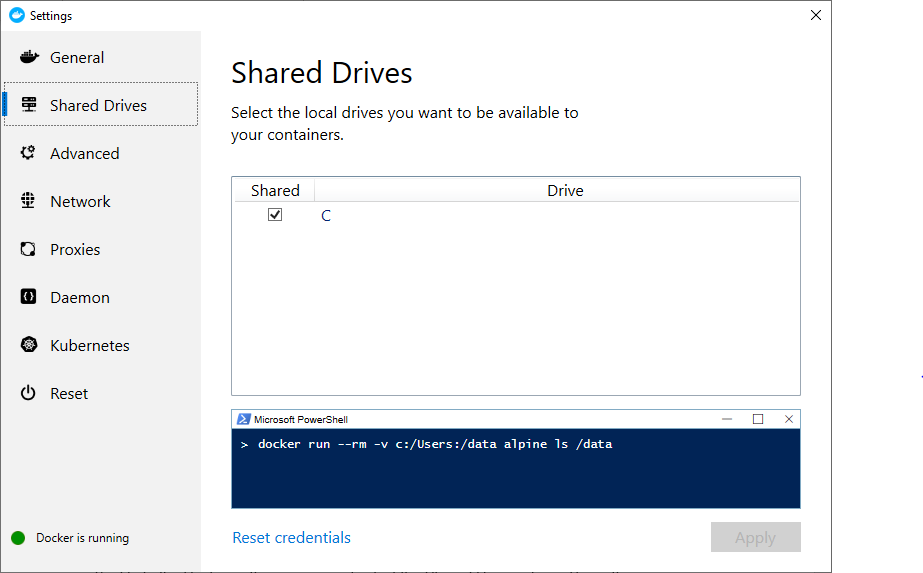
1. The software setup listed below is required to continue development of Sim Manager or deploy Sim Manager to containers. If the reader is only interested in developing Sim Manager they can choose to not install docker.

## Base Software Setup

1. Obtain the following installation files. Examples are provided of various tool sets, however the reader can substitute any they are comfortable with.

|  |  |  |
| --- | --- | --- |
| **Type** | **Example** | **Source** |
| Git | GitHub | <https://desktop.github.com/> |
| Java IDE with Maven | Spring Tool Suite | <https://spring.io/tools> |
| Lombok Library | Project Lombok | <https://projectlombok.org/> |
| Java Development Kit | Oracle JDK 13 | <https://www.oracle.com/technetwork/java/javase/downloads/index.html> |
| Container | Docker | <https://www.docker.com/products/docker-desktop> |
| Simulator | Cadmium | <https://github.com/SimulationEverywhere/Cadmium-Simulation-Environment> |
| Full Stack Development (Web, PHP, Database) | XAMPP | <https://www.apachefriends.org/index.html> |

1. Install software in the following order with default options:
   1. Oracle JDK 13;
   2. GitHub Desktop;
   3. Spring Tool Suite;
   4. Project Lombok;
   5. XAMPP;
   6. Docker; and
   7. Cygwin (per the Cadmium installation instructions).
2. Update your system PATH to include the following:
   1. The JAVA bin folder; and
   2. The Cygwin bin folder.
3. Add an environment variable called JAVA\_HOME and point to the base Java installation folder not the bin folder.
4. Once the software is installed clone the following Git repository:
   1. <https://github.com/robbarwell123/SYSC5104Project>
   2. Inside the [Simulator] folder create a [Cadmium] folder and clone the Cadmium code to that location.
   3. Copy files from the [Simulators/CadmiumAmend] folder and overwrite the associated files in the [Cadmium] folder. This fixes EIRational.hpp time and changes the Cadmium state logger to allow for JSON output.
5. Go to the Docker settings menu and select the Shared Drive option. The settings menu can be found by right clicking on the docker icon in the system tray . Provide docker access to your local drive to share with containers.



1. Web server configuration. Change the web server root to point to the [WebRoot] folder. In XAMPP this would be accomplished by modifying the httpd.conf file and adding the following lines.

Alias /Sim "C:/Dev/BuildDir/WebRoot"

<Directory "C:/Dev/BuildDir/WebRoot">

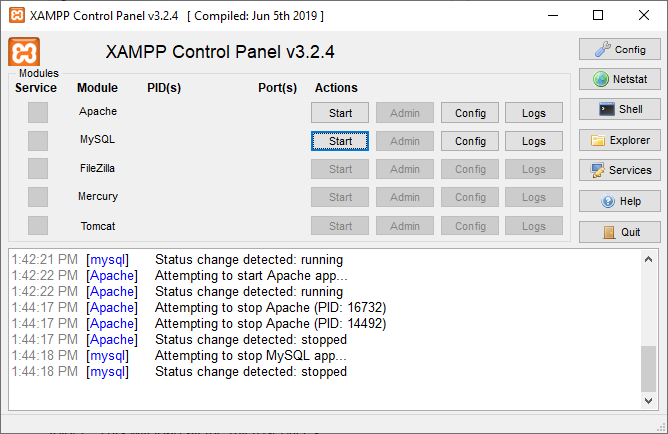
Options Indexes FollowSymLinks Includes ExecCGI

AllowOverride All

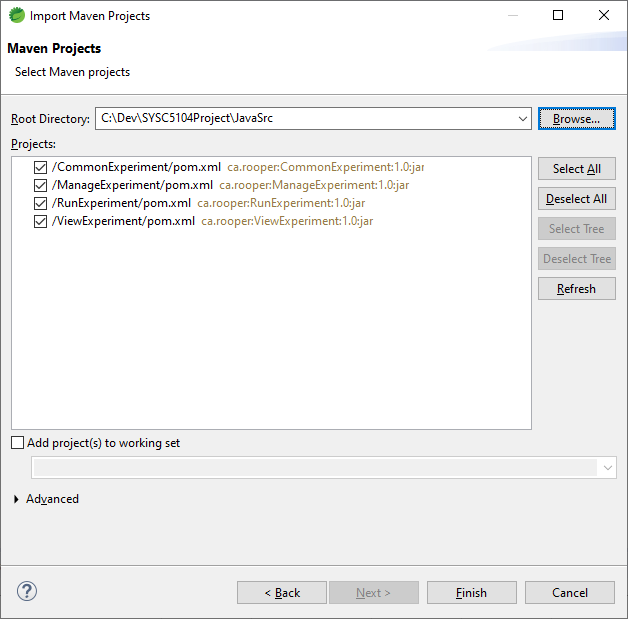
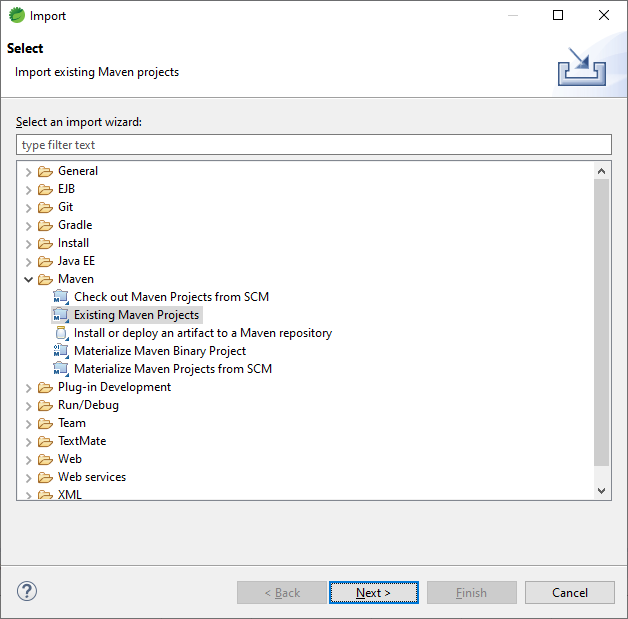
Require all granted

</Directory>

1. The database can be configured using phpMyAdmin. This is accessed by starting MySQL and Apache in the XAMPP control panel. To access phpMyAdmin select the “Admin” button next to MySQL once it has started. The following needs to be created:
   1. Database: sims
   2. Access to sims by user sims with password sims
   3. Use the create table script in [Scripts/Database/createtable.sql] to instantiate the experiments table in sims.



1. Open Sprint Tool Suite and select Import Projects from the left-hand pane. Under Maven in the tree list select Existing Maven Projects. Use the JavaSrc as the base folder. Select all microservices.



## Container Setup

### Useful Docker Commands

|  |  |
| --- | --- |
| **Command** | **Description** |
| docker logs {container name} | Displays the logs for a given container. Useful in trouble shooting. |
| docker [start / stop] {container name} | Starts and stops a given container. |
| docker rm {container name} | Used to remove a container. |

### Docker Steps

1. The microservices need to be compiled prior to creating the container. This is done using the following commands:
   1. Using the command line go to [JavaSrc/CommonExperiment] and run
      1. mvnw clean
      2. mvnw install
   2. For each of [JavaSrc/MangeExperiment], [JavaSrc/RunExperiment], and [JavaSrc/ViewExperiment] run the following
      1. mvnw clean
      2. mvnw package
2. Using the command line go to [Dockerfiles] and execute the following commands. Be sure to modify the commands based on the directory Sim Manager is installed in.
   1. Database
      1. docker build -f DatabaseDockerFile -t experimentdb c:/Dev/BuildDir/Scripts/Database
      2. docker run -p:3306:3306 --name mySQLDB -t experimentdb
   2. Sim Manager
      1. docker build -f WebServerDockerFile -t simmanager C:/Dev/BuildDir/WebRoot/SimManager
      2. docker run -p80:80 -v c:/Dev/BuildDir/Settings/cfg\_simmanager.js:/var/www/html/js/config/cfg\_simmanager.js --name SimManager -t simmanager
   3. Sim Visualizer
      1. docker build -f WebServerDockerFile -t simvisualizer C:/Dev/BuildDir/WebRoot/SimVisualizer
      2. docker run -p81:80 -v c:/Dev/BuildDir/Settings/cfg\_simvisualizer.js:/var/www/html/js/config/cfg\_simvisualizer.js --name SimVisualizer -t simvisualizer
   4. Manage Experiment
      1. docker build -f ManageExperimentDockerfile -t manageexperiment c:/Dev/BuildDir/JavaSrc/ManageExperiment/target
      2. docker run -v c:/Dev/BuildDir/Settings/PRODmanageexperiment.properties:/microservice/config/application.properties -v c:/Dev/BuildDir/Shared/experiments:/data --name ManageExperiment -p8091:8080 manageexperiment
   5. Run Experiment
      1. docker build -f RunExperimentDockerfile -t runexperiment c:/Dev/BuildDir
      2. docker run -v c:/Dev/BuildDir/Settings/PRODrunexperiment.properties:/microservice/config/application.properties -v c:/Dev/BuildDir/Shared:/common -v c:/Dev/BuildDir/Simulators/Cadmium:/cadmium -v c:/Dev/BuildDir/Scripts/Cadmium:/scripts/cadmium --name RunExperiment -p8092:8080 runexperiment
   6. View Experiment
      1. docker build -f ViewExperimentDockerfile -t viewexperiment c:/Dev/BuildDir/JavaSrc/ViewExperiment/target
      2. docker run -v c:/Dev/BuildDir/Settings/PRODviewexperiment.properties:/microservice/config/application.properties -v c:/Dev/BuildDir/Shared/experiments:/data --name ViewExperiment -p8093:8080 viewexperiment

# Running Software

Using the steps in the previous section will results in the following ports being assigned for the Sim Manager components:

|  |  |  |
| --- | --- | --- |
| **Component** | **Development Port** | **Container Port** |
| Sim Manager | 80 | 80 |
| Sim Visualizer | 80 | 81 |
| Manage Experiment | 8081 | 8091 |
| Run Experiment | 8082 | 8092 |
| View Experiment | 8083 | 8093 |
| Database | 3306 | 3306 |

## Initial Software Config

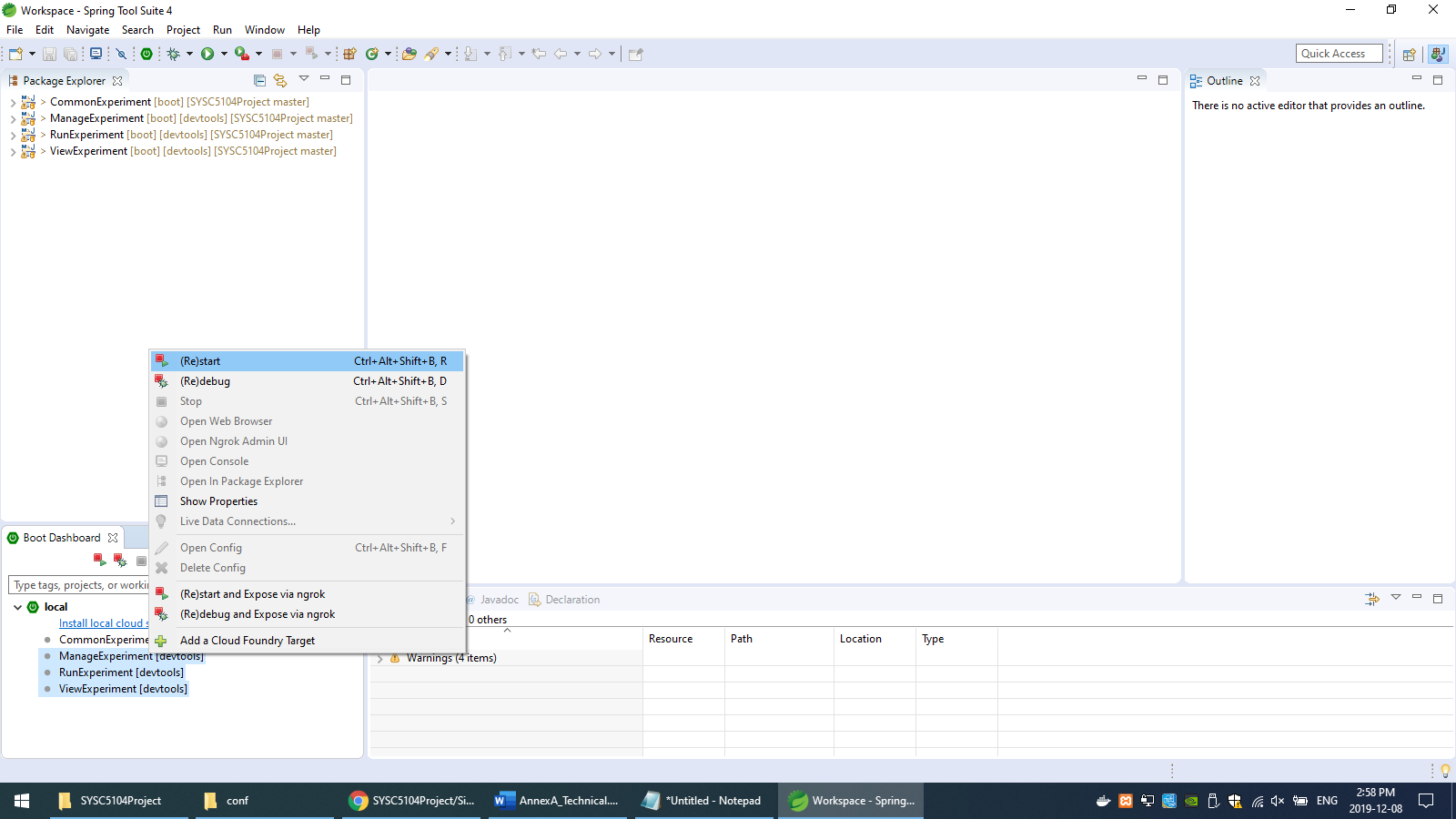
1. Before the software is used and after you add any new message types or other includes the user will need to update the make files in the [Shared] folder and re-issue the following commands when the services are running.
   1. http://localhost:8082/ResetCommon
   2. http://localhost:8092/ResetCommon

## Starting Containers

1. Containers were automatically launched when they were created in the previous section. To stop the containers use docker stop {container name}, to start the containers again after a system reboot use docker start {container name}. The container names are SimManger, SimVisualizer, ManageExperiment, RunExperiment, ViewExperiment, and mySQLDB.

## Starting Development Environment

1. Typically, the development and container environment can be used at the same time and useful for debugging. The only exception to this is the database container / database server. Only one of these should be running at any given time as they use the same ports. Additionally the database should be in-sync with the experiments file system This can be changed through either the development or container configuration.
2. To launch the local database and web server you can use the XAMPP control panel and click start on Apache and MySQL. This was covered during database instantiation in the previous sections.
3. To start the Spring Boot microservices you can run them from Spring Tool Suite by opening the application and selecting (Re)start on ManageExperiment, RunExperiment, and ViewExperiment.



# Configuring Software

1. The software by default is configured to point to [C:\Dev\BuildDir], this can be changed to reflect any base directory.
2. Configuration files for containers can be found in the [Settings] folder. Any changes here are mapped to the container and will be refreshed on the next reboot of the container. The properties files are for the microservices and the JavaScript files are for the web applications.
3. Configuration files for development are included in their respective source folders. The web configurations are in the [js/config] folder. The microservices have their property files in [src/main/resources].

## Web Settings

1. Sim Manager – cfg\_simmanager.js
   1. A list of REST service addresses. These will require changing if different ports / servers are used.
2. Sim Visualizer – cfg\_simvisualizer.js
   1. A list of REST service addresses. These will require changing if different ports / servers are used.

## Java Microservices

1. All
   1. Each microservice will have a section for database config where the connection string, username, and password are set. Please note containers are virtualized and have a different IP address from your local machine. You will need to update these values to your local host IP address.
   2. Experiments.basedir – sets the file location for where the experiment files are stored. Remember in containers this is based on the container file system, not the host file system.
2. Run Experiments – application.properties contains more values which are required for use in the XML to Cadmium conversion process.
   1. XSD prefixes tell the application where XSD files are located to validate the XML files.
   2. Experiments.compile / experiments.run point to the shell scripts required to compile and run the experiments.
   3. Cadmium prefixes tell the application where to find specific include files.
   4. Cadmium.resetcommon tells the application where the script is to re-compile the pre-compiled components such as messages.