Praktikum „Ingenieursmäßige Software-Entwicklung“ Dockerifizierung of Palladio

Thomas Weber

Institute for Program Structures and Data Organization (IPD)
Advisor: M.Sc. Dominik Werle and M.Sc. Stephan Seifermann

1 Motivation

The reproducibility of scientific experiments is one of their core requirements. However, this is often difficult or even impossible to implement because the environment of the experiment changes too quickly. Dependencies on operating systems, programming languages and platforms can lead to relatively large changes in the results of an experiment. In order to record these influences and to make the experiments easy and uncomplicatedly reproducible, the creation of a docker image is a good idea, which records all relevant influences. This makes the experiments reproducible, expandable, portable and archivable.

2 Palladio-Docker

2.1 Description

The subprojects in this project contain mostly dockerfiles for the execution of a headless eclipse application in general and more specific the Palladio Experiment Automation.

2.1.1 Eclipse

Project contains a dockerfile installing the *openjdk-11-jre-headless* and *wget* on the **ubuntu:20.04** image. Unpacks *eclipse-java-2019-12* from the fau[1] into **/usr**. The eclipse application is located in **/usr/eclipse**.

[1] <https://www.fau.eu/>

2.1.2 EclipseExampleApplication

Project extends my eclipse docker image[2] with a custom headless application that is installed into eclipse via the dropins folder[3].

2.1.3 EclipseModellingFramework

Project contains a dockerfile installing the *openjdk-11-jre-headless* and *wget* on the **ubuntu:20.04** image. Unpacks *eclipse-modeling-2019-12* from the fau[4] into **/usr**. The eclipse application is located in **/usr-eclipse**. If you do not need the modelling tools, consider using my **eclipse**[5] image.

2.1.4 EclipsePalladio

Project extends my eclipse modeling tools image[6]. In General can install all packages named in features.txt from the given update sites. This dockerfile installs all components used by the Palladio Component Model[7]. It can be modified to enable headless builds similar to this[8].

2.1.5 PalladioExperimentAutomation

Project extends my eclipse palladio installation image[9] with files to run ExperimentAutomation[10] experiments. Also installs **xfvb** and **libgtk-3-0** because they are used by palladio during the experiment runs. Adds the scripts shown in the next sections to the image. To run an experiment refer to the projects PalladioExperimentImage[11] and PalladioRunExperiment[12].

2.1.6 PalladioExperimentImage

Project extends my eclipse palladio installation image[13] with sources to run an ExperimentAutomation[14] experiment[15].

[2] <https://hub.docker.com/repository/docker/thomasweber/eclipse>
[3] <https://help.eclipse.org/2019-12/index.jsp?topic=\char‘%{}2Forg.eclipse.platform.doc.isv\char‘%{}2Freference\char‘%{}2Fmisc\char‘%{}2Fp2_dropins_format.html>
[4] <https://www.fau.eu/>
[5] <https://github.com/TomWerm/Palladio-Docker/tree/master/Eclipse>
[6] <https://hub.docker.com/repository/docker/thomasweber/eclipsemodelingtools>
[7] <https://www.palladio-simulator.com/home/>
[8] <https://gnu-mcu-eclipse.github.io/advanced/headless-builds/>
[9] <https://hub.docker.com/repository/docker/thomasweber/eclipsepalladio>
[10] <https://sdqweb.ipd.kit.edu/wiki/Palladio_Experiment_Automation>
[11] <https://github.com/TomWerm/Palladio-Docker/tree/master/PalladioExperimentImage>
[12] <https://github.com/TomWerm/Palladio-Docker/tree/master/PalladioRunExperiment>
[13] <https://hub.docker.com/repository/docker/thomasweber/eclipsepalladio>
[14] <https://sdqweb.ipd.kit.edu/wiki/Palladio_Experiment_Automation>
[15] <https://github.com/PalladioSimulator/Palladio-Addons-ExperimentAutomation>

2.1.7 PalladioRunExperiment

Project extends my eclipse palladio experiment automation image[16]. It simply adds a small cmd script to run experiments with a few parameters and mounts.

2.1.8 Documentation

Motivation for the project and the Readmes in a latex project.

2.2 Further reading

- p2 director[17]
- Experiment automation (EA)[18]
- EA - Github[19]
- PCM[20]
- Experiment Application[21]
- PCM Nightly[22]
- EA espresso - Github[23]
- Docker-MavenXvfb[24]
- UnsatisfiedLinkError[25]
- mirror eclipse plugin sites[26]
- Eclipse headless[27]
- Valid URL[28]
- Meta-Model[29]
- Eclipse Core[30]

[16] <https://github.com/TomWerm/Palladio-Docker/tree/master/PalladioExperimentAutomation>
[17] <https://help.eclipse.org/kepler/index.jsp?topic=/org.eclipse.platform.doc.isv/guide/p2_director.html>
[18] <https://sdqweb.ipd.kit.edu/wiki/Palladio_Experiment_Automation>
[19] <https://github.com/PalladioSimulator/Palladio-Addons-ExperimentAutomation>
[20] <https://github.com/PalladioSimulator/Palladio-Bench-Product/blob/master/products/org.palladiosimulator.product/org.palladiosimulator.palladiobench.product>
[21] <https://github.com/PalladioSimulator/Palladio-Addons-ExperimentAutomation/blob/master/bundles/org.palladiosimulator.experimentautomation.application/src/org/palladiosimulator/experimentautomation/application/ExperimentApplication.java>
[22] <https://updatesite.palladio-simulator.com/palladio-build-updatesite/nightly/>
[23] <https://github.com/PalladioSimulator/Palladio-Addons-ExperimentAutomation/tree/master/bundles/org.palladiosimulator.experimentautomation.examples.espresso>
[24] <https://github.com/kit-sdq/Docker-MavenXvfb>
[25] <https://bugs.eclipse.org/bugs/show_bug.cgi?id=549244>
[26] <https://stackoverflow.com/questions/1371176/downloading-eclipse-plug-in-update-sites-for-offline-installation>
[27] <https://gnu-mcu-eclipse.github.io/advanced/headless-builds/>
[28] <https://stackoverflow.com/questions/1547899/which-characters-make-a-url-invalid>
[29] <https://github.com/PalladioSimulator/Palladio-Addons-ExperimentAutomation/blob/master/bundles/org.palladiosimulator.experimentautomation/model/experimentautomation.ecore>
[30] <https://www.eclipse.org/eclipse/platform-core/>

- Alpine Eclipse[31]

3 Eclipse on Ubuntu 20.04

3.1 Description

Project contains a dockerfile installing the *openjdk-11-jre-headless* and *wget* on the **ubuntu:20.04** image. Unpacks *eclipse-java-2019-12* from the fau[32] into **/usr**. The eclipse application is located in **/usr/eclipse**.

3.2 Usage

Simple docker image with a working headless eclipse installation. For an example how to use the application have a look at **EclipseExampleApplication**[33] as a folder in this[34] repository.

3.3 Docker hub

The image can be found at docker hub[35]. Auto-Build is currently disabled.

3.4 Dockerfile

```
FROM ubuntu:20.04
RUN apt-get clean && \
    apt-get update --fix-missing && \
    apt-get install -y --fix-missing openjdk-11-jre-headless wget
RUN /usr/bin/wget 'http://ftp.fau.de/eclipse/technology/epp/downloads/release
   /2019-12/R/eclipse-java-2019-12-R-linux-gtk-x86_64.tar.gz' && \
    tar -xzf eclipse-java-2019-12-R-linux-gtk-x86_64.tar.gz -C /usr && \
    rm eclipse-java-2019-12-R-linux-gtk-x86_64.tar.gz
```

3.5 Known Issues

This image cannot execute a non-headless eclipse instance as it will produce a

```
java.lang.UnsatisfiedLinkError: 'void org.eclipse.swt.internal.gtk.OS.
   _cachejvmptr()'
```

If you have any dependencies on an user interface consider installing *xvfb* and *libgtk-3-0* as used in the PalladioExperimentAutomation[36] project.

[31] <https://stackoverflow.com/questions/43209656/cannot-run-jfrog-executable-from-inside-alpine-linux-container>
[32] <https://www.fau.eu/>
[33] <https://github.com/TomWerm/Palladio-Docker/tree/master/EclipseExampleApplication>
[34] <https://github.com/TomWerm/Palladio-Docker>
[35] <https://hub.docker.com/repository/docker/thomasweber/eclipse>
[36] <https://github.com/TomWerm/Palladio-Docker/tree/master/PalladioExperimentAutomation>

4 Example application for eclipse

4.1 Description

Project extends my eclipse docker image[37] with a custom headless application that is installed into eclipse via the dropins folder[38].

4.2 Usage

This example illustrates how to run a headless eclipse applications in a docker container. Import your own application by changing the copied file in the dockerfile *_edu.kit.sdq.exampleapplication* 1.0.0.202002111755.jar[39] and the name of the application in the **RunApplication.sh**[40] script. Build the dockerfile to test if your application is working (as the script will start your application). The configuration problem (org.eclipse.m2e.logback.configuration: The org.eclipse.m2e.logback. configuration bundle was activated before the state location was initialized. Will retry after the state location is initialized.) has no known effects. The plug-in contained in the image will simply print "plugin started" to the standard output. The plug-in (application) creation process is described in the next paragraph.

4.3 Write your own application

Install eclipse[41] and the eclipse plug-in development environment[42]. Then create a new plug-in project with **File**->**New**->**Other**->**Plug-in-Development**->**Plug-in-Project**. Now give it a name and click next. Deactivate **Generate an activator** and deactivate **This plug-in will make contributions to the UI**. Be sure to chose a Java version prior to java 11, because that is the version this image is extending. If you choose Java 12 for example, the application will not show up in the list of loaded applications without any hint why it does not. Hit finish and head to the Extensions section of the Manifest.mf in the eclipse editor (deselect Show only extension points from the required plug-ins). Add **org.eclipse.core.runtime.applications** to the extensions of your plug-in and enable the singleton property. Open the **plugin.xml** with the text editor and add a name and change the id for the **application** extension point implemented by the plug-in. Add this code

```
<run class="yourpackage.yourclass">
</run>
```

with your package- and class-name to your application. It should look like this:

[37] <https://hub.docker.com/repository/docker/thomasweber/eclipse>

[38] <(https://help.eclipse.org/2019-12/index.jsp?topic=\char‘%{}2Forg.eclipse.platform.doc.isv\char‘%{}2Freference\char‘%{}2Fmisc\char‘%{}2Fp2_dropins_format.html)>

[39] <https://github.com/TomWerm/Palladio-Docker/blob/master/EclipseExampleApplication/edu.kit.sdq.exampleapplication_1.0.0.202002111755.jar>

[40] <https://github.com/TomWerm/Palladio-Docker/blob/master/EclipseExampleApplication/RunApplication.sh>

[41] <https://www.eclipse.org/eclipseide/>

[42] <https://www.eclipse.org/pde/>

```
<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.4"?>
<plugin>
   <extension
         id="application"
         point="org.eclipse.core.runtime.applications">
      <application
            cardinality="singleton-global"
            thread="main"
            visible="true">
         <run
               class="edu.kit.sdq.exampleapplication.Main">
         </run>
      </application>
   </extension>

</plugin>
```

The class you added (in my case *edu.kit.sdq.exampleapplication.Main*) has to implement the **IApplication**-Interface. An example implementation can look like this:

```
package edu.kit.sdq.exampleapplication;

import org.eclipse.equinox.app.IApplication;
import org.eclipse.equinox.app.IApplicationContext;

public class Main implements IApplication {

  @Override
  public Object start(final IApplicationContext context) throws Exception {
    System.out.println("plugin started");
    return IApplication.EXIT_OK;
  }

  @Override
  public void stop() {
    // Add operations when your plugin is stopped
  }
}
```

To export your plug-in hit **File**->**Export**->**Plug-in Development**->**Deployable plug-ins and fragments**. Select your project and add an output path and hit **finish**. More information about a headless application can be found here[43] and general information about eclipse applications here[44].

[43] <https://codeandme.blogspot.com/2012/02/creating-headless-application.html>
[44] <https://wiki.eclipse.org/FAQ_What_is_an_Eclipse_application\char'%{}3F>

4.4 Docker hub

The image cannot be found at docker hub[45] because it is only for demonstrational purposes. If you want to use your own application have a look at the two previous chapters.

4.5 Dockerfile

```
FROM thomasweber/eclipse
COPY RunApplication.sh /usr/RunApplication.sh
RUN chmod a+rx usr/RunApplication.sh
COPY edu.kit.sdq.exampleapplication_1.0.0.202002111755.jar /usr/eclipse/
    dropins/plugins/edu.kit.sdq.exampleapplication_1.0.0.202002111755.jar
RUN /usr/RunApplication.sh
```

4.6 Scripts

```
#!/bin/bash
/usr/eclipse/eclipse \
     -clean \
     -application edu.kit.sdq.exampleapplication.application \
     -consoleLog
```

5 Eclipse Modeling on Ubuntu 20.04

5.1 Description

Project contains a dockerfile installing the *openjdk-11-jre-headless* and *wget* on the **ubuntu:20.04** image. Unpacks *eclipse-modeling-2019-12* from the fau[46] into **/usr**. The eclipse application is located in **/usr-eclipse**. If you do not need the modelling tools, consider using my **eclipse**[47] image.

5.2 Usage

Simple docker image with a working headless eclipse installation. For an example how to use the application have a look at **EclipseExampleApplication**[48] or **PalladioExperimentAutomation**[49] as a folder in this[50] repository.

5.3 Docker hub

The image can be found at docker hub[51]. Auto-Build is currently disabled.

[45] <https://hub.docker.com/>
[46] <https://www.fau.eu/>
[47] <https://github.com/TomWerm/Palladio-Docker/tree/master/Eclipse>
[48] <https://github.com/TomWerm/Palladio-Docker/tree/master/EclipseExampleApplication>
[49] <https://github.com/TomWerm/Palladio-Docker/tree/master/PalladioExperimentAutomation>
[50] <https://github.com/TomWerm/Palladio-Docker>
[51] <https://hub.docker.com/repository/docker/thomasweber/eclipsemodelingtools>

5.4 Dockerfile

```
FROM ubuntu:20.04
RUN apt-get clean && \
    apt-get update --fix-missing && \
    apt-get install -y --fix-missing openjdk-11-jre-headless wget
RUN /usr/bin/wget 'https://ftp.fau.de/eclipse/technology/epp/downloads/
    release/2019-12/R/eclipse-modeling-2019-12-R-linux-gtk-x86_64.tar.gz' &&
    \
    tar -xzf eclipse-modeling-2019-12-R-linux-gtk-x86_64.tar.gz -C /usr && \
    rm eclipse-modeling-2019-12-R-linux-gtk-x86_64.tar.gz
```

6 Eclipse with installed palladiosimulator

6.1 Description

Project extends my eclipse modeling tools image[52]. In General can install all packages named in features.txt from the given update sites. This dockerfile installs all components used by the Palladio Component Model[53]. It can be modified to enable headless builds similar to this[54].

6.2 Usage

Simple docker image with a working headless eclipse and palladio installation. For an example how to use the application have a look at **PalladioExperimentAutomation**[55] as a folder in this[56] repository.

6.3 Docker hub

The image can be found at docker hub[57]. Auto-Build is currently disabled.

6.4 Dockerfile

- probably you have to add basic update sites[58] that cannot be located during the installation process to the list in features.txt

```
FROM thomasweber/eclipsemodelingtools
# Install everything from features.txt
COPY InstallFeature.sh /usr/InstallFeature.sh
```

[52] <https://hub.docker.com/repository/docker/thomasweber/eclipsemodelingtools>
[53] <https://www.palladio-simulator.com/home/>
[54] <https://gnu-mcu-eclipse.github.io/advanced/headless-builds/>
[55] <https://github.com/TomWerm/Palladio-Docker/tree/master/PalladioExperimentAutomation>
[56] <https://github.com/TomWerm/Palladio-Docker>
[57] <https://hub.docker.com/repository/docker/thomasweber/eclipsepalladio>
[58] <https://stackoverflow.com/questions/1371176/downloading-eclipse-plug-in-update-sites-for-offline-installation>

```
COPY InstallLocalUpdates.sh /usr/InstallLocalUpdates.sh
COPY features.txt usr/features.txt
RUN chmod a+rx usr/InstallLocalUpdates.sh
RUN chmod a+rx usr/InstallFeature.sh
RUN usr/InstallLocalUpdates.sh
```

6.5 Scripts

6.5.1 InstallLocalUpdates.sh

- piping every 2 arguments in a line of the input file in the InstallFeature script

```
sed '/^#/ d' /usr/features.txt | sed 's/\([^%]*\)%\([^%]*\)/ "\1" "\2"/
    ' | xargs -r -n 2 /usr/InstallFeature.sh
```

6.5.2 InstallFeature.sh

- installs the given features from the given update sites with the p2 director[59]
- the basic update site for the 2019-12 release is already added

```
echo installing "$1"
echo from server "$2"
/usr/eclipse/eclipse \
-application org.eclipse.equinox.p2.director \
-repository https://download.eclipse.org/releases/2019-12/,"$2" \
-installIU "$1"
```

6.5.3 features.txt

- The format of the file is due to this site describing a valid url[60]:

```
# comment lines
org.feature1,org.feature2%updatesite1,updatesite2
```

- contains all features needed for a palladio[61] experiment automation run[62]

```
#package1,package2,etc%updatesite1,updatesite2,etc
#currently all packages needed to run the palladio experiment
    automation
org.palladiosimulator.edp2.feature.feature.group,org.palladiosimulator.
    pcm.feature.feature.group,org.palladiosimulator.simucom.feature.
    feature.group,org.palladiosimulator.solver.feature.feature.group,org
```

[59] <https://help.eclipse.org/kepler/index.jsp?topic=/org.eclipse.platform.doc.isv/guide/p2_director.html>

[60] <https://stackoverflow.com/questions/1547899/which-characters-make-a-url-invalid>

[61] <https://github.com/PalladioSimulator/Palladio-Bench-Product/blob/master/products/org.palladiosimulator.product/org.palladiosimulator.palladiobench.product>

[62] <https://updatesite.palladio-simulator.com/palladio-build-updatesite/nightly/>

```
    .palladiosimulator.recorderframework.feature.feature.group,org.
    palladiosimulator.analyzer.feature.feature.group,org.
    palladiosimulator.monitorrepository.feature.feature.group,org.
    palladiosimulator.simulizar.feature.feature.group,org.
    palladiosimulator.simulation.abstractsimengine.desmoj.feature.
    feature.group%https://updatesite.palladio-simulator.com/palladio-
    build-updatesite/releases/4.2.0/
# comment this line and uncomment the following one to add the export
    function in the nightly release
# add the feature back to the list above once a new release with export
     is available
org.palladiosimulator.experimentautomation.application.feature.feature.
    group,org.palladiosimulator.experimentautomation.feature.feature.
    group,org.palladiosimulator.experimentautomation.application.
    tooladapter.simulizar.feature.feature.group%https://updatesite.
    palladio-simulator.com/palladio-addons-experimentautomation/branches
    /csvExportRelease/,https://updatesite.palladio-simulator.com/
    palladio-build-updatesite/releases/4.2.0/
# org.palladiosimulator.experimentautomation.application.feature.
    feature.group%https://updatesite.palladio-simulator.com/palladio-
    build-updatesite/releases/nightly/
```

7 Eclipse with installed palladiosimulator and files for ExperimentAutomation runs

7.1 Description

Project extends my eclipse palladio installation image[63] with files to run ExperimentAutomation[64] experiments[65]. Also installs **xfvb** and **libgtk-3-0** because they are used by palladio during the experiment runs. Adds the scripts shown in the next sections to the image. To run an experiment refer to the projects PalladioExperimentImage[66] and PalladioRunExperiment[67].

7.2 Usage

This docker image adds scripts and installs programs. It cannot be executed.

[63] <https://hub.docker.com/repository/docker/thomasweber/eclipsepalladio>
[64] <https://sdqweb.ipd.kit.edu/wiki/Palladio_Experiment_Automation>
[65] <https://github.com/PalladioSimulator/Palladio-Addons-ExperimentAutomation>
[66] <https://github.com/TomWerm/Palladio-Docker/tree/master/PalladioExperimentImage>
[67] <https://github.com/TomWerm/Palladio-Docker/tree/master/PalladioRunExperiment>

7.3 Docker hub

The image can be found at docker hub[68]. Auto-Build is currently disabled.

7.4 Dockerfile

```
FROM thomasweber/eclipsepalladio
RUN apt-get clean && \
    apt-get update --fix-missing && \
    apt-get install -y --fix-missing xvfb libgtk-3-0
COPY RunExperimentAutomation.sh /usr/RunExperimentAutomation.sh
RUN chmod a+rx usr/RunExperimentAutomation.sh
COPY ModifyExperimentsFile.sh /usr/ModifyExperimentsFile.sh
RUN chmod a+rx usr/ModifyExperimentsFile.sh
```

7.5 ModifyExperimentsFile.sh

- modifies the line with the datasource in the given file to export csv and add a valid location in the docker image

```
#!/bin/bash
# modifies the file given with the first parameter and saves it to the file
    given with the second parameter
sed -E 's/<datasource [^>]*\/>/<datasource xsi:type=\"ExperimentAutomation.
    Experiments.AbstractSimulation:FileDatasource\" location=\"\/result\"
    exportOption=\"CSV\"\/>/' "$1" > "$2"
```

7.6 RunExperimentAutomation.sh

- starts the xvfb server and runs the experiment with the given parameters

```
#!/bin/bash
# first argument is the path to the original experiments file, second is the
    path where to store the generated file
Xvfb :99 -screen 0 1920x1080x16 &
export DISPLAY=:99
/usr/ModifyExperimentsFile.sh "$1" "$2"
/usr/eclipse/eclipse \
     -clean \
     -application org.palladiosimulator.experimentautomation.application \
     -consoleLog "$2" \
     -data "/usr/workspace"
```

[68] <https://hub.docker.com/repository/docker/thomasweber/palladioexperimentautomation>

8 PalladioExperimentAutomation

8.1 Description

Project extends my eclipse palladio installation image[69] with sources to run an ExperimentAutomation[70] experiment[71].

8.2 Usage

Adapt the parameters in **RunDockerImage.cmd** to your experiment and the experiment data in the folder *ExperimentData*. Run the experiment with the cmd script. In this way, the results of the experiments are reproducible on any machine. For another approach to use the dockerfile have a look at the PalladioRunExperiment[72]-Image that mount the input and output folders directly in the script.

8.3 Docker hub

The image can be found at docker hub[73]. Auto-Build is currently disabled.

8.4 Dockerfile

```
FROM thomasweber/palladioexperimentautomation:latest
COPY ExperimentData/ /usr/ExperimentData
```

8.5 RunDockerImage.cmd

- windows script to run the experiments in the annotated file and store the results at the current location
- change the paths to match your local folder structure
- change the used image to the one you build with your experiment data (can be a local image)
- executes the given experiment and saves the results in the folder \Output

```
SET SRC_PATH=%cd%
SET IMAGE_NAME=thomasweber/palladioexperimentimage:latest
SET CONTAINER_PATH=/usr
SET EXPERIMENT_FILE_NAME=Capacity.experiments
SET EXPERIMENT_GEN_FILE_NAME=Generated.experiments
SET EXPERIMENTS_FILE_DIR=/ExperimentData/model/Experiments/
SET EXPERIMENTS_FILE_DIR_WINDOWS=\ExperimentData\model\Experiments\
docker pull %IMAGE_NAME%
```

[69] <https://hub.docker.com/repository/docker/thomasweber/eclipsepalladio>
[70] <https://sdqweb.ipd.kit.edu/wiki/Palladio_Experiment_Automation>
[71] <https://github.com/PalladioSimulator/Palladio-Addons-ExperimentAutomation>
[72] <https://github.com/TomWerm/Palladio-Docker/tree/master/PalladioRunExperiment>
[73] <https://hub.docker.com/repository/docker/thomasweber/palladioexperimentimage>

```
docker run -it -d %IMAGE_NAME%
FOR /F "tokens=*" %%g IN ('docker ps -q') do (SET CONTAINER_ID=%%g)
FOR /F "tokens=*" %%g IN ('docker ps --format "{{.Names}}"') do (SET
    CONTAINER_NAME=%%g)
docker exec -it %CONTAINER_NAME% /usr/RunExperimentAutomation.sh %
    CONTAINER_PATH%%EXPERIMENTS_FILE_DIR%%EXPERIMENT_FILE_NAME% %
    CONTAINER_PATH%%EXPERIMENTS_FILE_DIR%%EXPERIMENT_GEN_FILE_NAME%
docker cp %CONTAINER_NAME%:/result "%SRC_PATH%\Output"
docker stop %CONTAINER_ID%
PAUSE
```

8.6 Known issues with eclipse / palladio

- all of these issues have no known effect on the experiment run itself or the results

```
SWT SessionManagerDBus: Failed to connect to org.gnome.SessionManager: Failed
     to execute child process ?dbus-launch? (No such file or directory)
SWT SessionManagerDBus: Failed to connect to org.xfce.SessionManager: Failed
    to execute child process ?dbus-launch? (No such file or directory)

[WARN] Problem while deactivating CDOViewProviderRegistryImpl
java.lang.InterruptedException

java.lang.InterruptedException
```

9 Palladio example runs

9.1 Description

Project extends my eclipse palladio experiment automation image[74]. It simply adds a small cmd script to run experiments with a few parameters and mounts.

9.2 Usage

Adapt the constants in the cmd script and if neccessary the mounts in the docker run command.

9.3 Docker hub

The image cannot be found at docker hub because it only contains a cmd script to run an experiment.

[74] <https://github.com/TomWerm/Palladio-Docker/tree/master/PalladioExperimentAutomation>

9.4 Scripts

9.4.1 RunDockerImage.cmd

- defines constants for the different paths needed
- pulls the image from docker hub
- runs the RunExperimentAutomation.sh[75] script
- with input and output folders mounted (if you have some trouble perhaps this[76] contains a solution)

```
SET SRC_PATH=%cd%
SET IMAGE_NAME=thomasweber/palladioexperimentautomation:latest
SET CONTAINER_PATH=/usr
SET EXPERIMENT_FILE_NAME=Capacity.experiments
SET EXPERIMENT_GEN_FILE_NAME=Generated.experiments
SET EXPERIMENTS_FILE_DIR=/ExperimentData/model/Experiments/
docker pull %IMAGE_NAME%
docker run -it -v ""%SRC_PATH%\Output"":/result -v""%SRC_PATH%\ExperimentData
    :/usr/ExperimentData"" %IMAGE_NAME% /usr/RunExperimentAutomation.sh %
    CONTAINER_PATH%%EXPERIMENTS_FILE_DIR%%EXPERIMENT_FILE_NAME% %
    CONTAINER_PATH%%EXPERIMENTS_FILE_DIR%%EXPERIMENT_GEN_FILE_NAME%
PAUSE
```

9.5 Known issues with eclipse / palladio

- all of these issues have no known effect on the experiment run itself or the results

```
SWT SessionManagerDBus: Failed to connect to org.gnome.SessionManager: Failed
     to execute child process ?dbus-launch? (No such file or directory)
SWT SessionManagerDBus: Failed to connect to org.xfce.SessionManager: Failed
    to execute child process ?dbus-launch? (No such file or directory)

[WARN] Problem while deactivating CDOViewProviderRegistryImpl
java.lang.InterruptedException

java.lang.InterruptedException
```

[75] <https://github.com/TomWerm/Palladio-Docker/tree/master/PalladioExperimentAutomation>
[76] <https://rominirani.com/docker-on-windows-mounting-host-directories-d96f3f056a2c>