Karlsruher Institut für Technologie

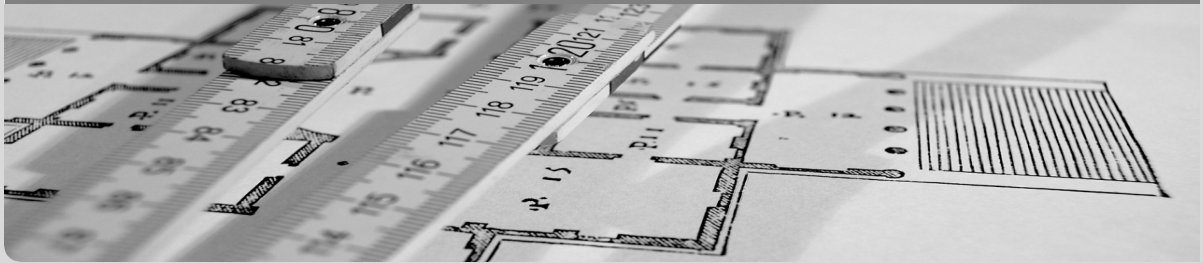
Software Design and Quality

Praktikum „Ingenieursmäßige Software-Entwicklung“
Dockerifizierung von Palladio

Betreuer: Dominik Werle und Stephan Seifermann

Thomas Weber | 25.03.2020

INSTITUT FÜR PROGRAMMSTRUKTUREN UND DATENORGANISATION



Motivation

Reproduzierbarkeit

- zentrale Anforderung an wissenschaftliche Experimente
- häufig aber nur schwer oder überhaupt nicht umsetzbar
  - Abhängigkeiten von äußeren Umständen lassen sich nicht exakt nachstellen
  - Aktualisierungen der Plattform verändern Ergebnisse

Große Zahl von Einflussfaktoren in Palladio

- OS
- Java
- Eclipse
- Palladio
- etc.

P.I.V.A

Problem

- Experimente zur Reproduktion der Ergebnisse zur Verfügung stellen umständlich
- Experimente nachbauen ebenfalls nicht einfach umsetzbar durch Einflussfaktoren

Idee

- Erstellen eines Docker-Images mit Konfigurations- und Experimentdaten

Vorteile

- Experimente können einfach zur Verfügung gestellt und durchgeführt werden
- Aufwand für Reviewer geringer

Aktionen

- Erstellen eines Dockerfiles für Palladio und die Experiment Automation

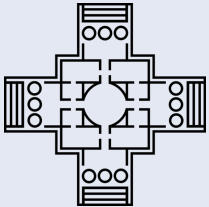
Palladio Experiment Automation

Palladio

- QoS-Vorhersagen für Software-Architekturen
- Modellbasierter Vergleich verschiedener Architekturalternativen

Experiment Automation

- ermöglicht die automatische Ausführung von Simulationen
- unterstützt verschiedene Analysen
- einzelne Experimente müssen nicht mehr separat konfiguriert oder gestartet werden
- besitzt eigenes Metamodell zur Konfiguration
- läuft headless



Docker

Was ist Docker? [1]

- Werkzeug zur Erstellung von portablen Docker-Images zur Ausführung in Docker-Containern
- Verpacken aller nötigen Ressourcen in Docker-Image
- Docker-Image kann durch einen Docker-Container überall ausgeführt werden
- unabgängig vom installierten Host-System (sofern Docker verwendbar ist)
- leichtgewichtiger als eine Virtual Machine (VM)

Docker

Begriffe [2]

- Dockerfile
  - ’Bauanleitung’ zur Erstellung eines Images
  - bestehendes Docker-Image wird als Grundlage verwendet
  - Hinzufügen weiterer Ressourcen (Skripte, Bibliotheken, etc.)
- Docker-Image
  - vergleichbar mit ’Snapshot’ einer VM
  - nicht veränderbar
  - Ausführbarkeit bleibt erhalten
  - verschiedene Schichten
- Docker-Container
  - Docker Virtual Machine
  - veränderbar

Github und Docker Hub

Github

- öffentliche Speicherung des Projekts
- abrufbar durch Docker Hub

Docker Hub

- Bibliothek für Container Images
- stellt Ressourcen zum Bauen von Images bereit
- gebaute Images auf `https://hub.docker.com/` abrufbar
- nutzbar als Basis-Image für andere Dockerfiles

Eclipse

Eclipse

- Eclipse Modelling Tools als Grundlage für Palladio

Begriffe

- Eclipse Application
  - Plug-In, welches Einstiegspunkt nach Framework-Initialisierung bietet
  - vergleichbar mit einer Java main-Methode
- Eclipse Bundle
  - Menge an Dateien (Ressourcen und Code)
  - stellt Erweiterung für das Eclipse Framwork bereit
- Eclipse Updatesite
  - stellt Bundles zur Installation zur Verfügung

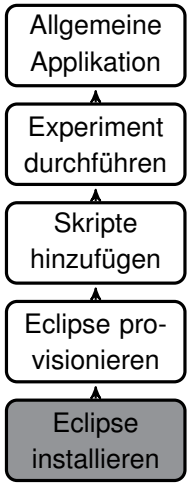
Eclipse Modelling Framework

Eclipse

- Als Basis-Image wird Ubuntu 20.04 genutzt
- Alpine nicht einfach nutzbar, da Eclipse gegen libc kompiliert ist

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 'release.tar.gz' && \
    tar -xzf release.tar.gz -C /usr && \
    rm release.tar.gz
```



Installation von Palladio

Provisionierung von Eclipse

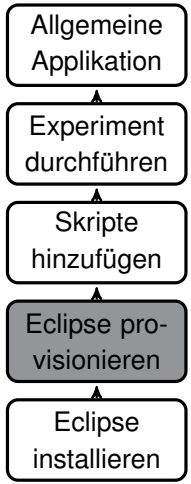
- Nutzung der p2 Provisioning Platform [9]
- Installation über Konfigurationsdatei parametrisierbar

Dockerfile

```
FROM palladiosimulator/eclipse-modeling-tools
COPY EclipsePalladio/InstallFeature.sh /usr/InstallFeature.sh
COPY EclipsePalladio/InstallLocalUpdates.sh /usr/InstallLocalUpdates.sh
COPY EclipsePalladio/features.txt usr/features.txt
RUN usr/InstallLocalUpdates.sh
```

Format der Konfigurationsdatei

package1,package2,etc%updatesite1,updatesite2,etc



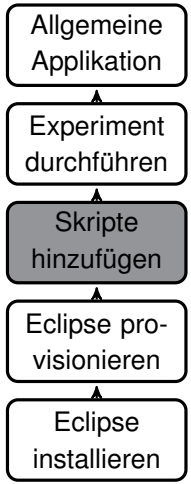
Hinzufügen weiterer Ressourcen

Bibliotheken und weitere Skripte

- xvfb und libgtk-3-0 zur Ausführung der Palladio Experiment Automation
- Skripte zum Modifizieren der .experiments Dateien und zur Ausführung der Experimente

RunExperimentAutomation.sh

```
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"
```



Ausführen der Experimente

Beschaffung der Experiment-Daten im Docker-Image

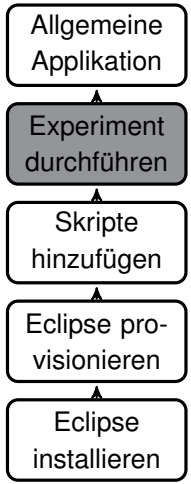
- mit in das Image packen
- über Docker-Run mounten

Dockerfile

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

RunDockerImage.cmd

```
docker run -it -v localPathResult:imagePathResult -v localPathData:imagePathData
    imageName /usr/RunExperimentAutomation.sh pathToExperiment pathToFile
```



Änderung des Metamodells

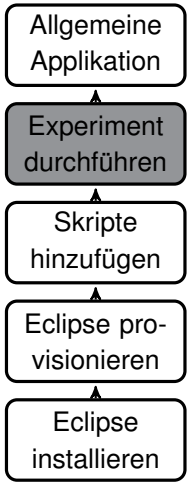
Beschaffung der Experiment-Daten aus dem Docker-Image

- Zur einfacheren Nutzung Export ins CSV-Format
- Export Option CSV zum Metamodell der Palladio Experiment Automation hinzugefügt

ModifyExperimentsFile.sh

- Pfade zu Experimentdaten für das Docker-Image anpassen
- Exportoption in Konfigurationsdatei anpassen

```
sed -E s/<datasource [^>]*\/>/<datasource xsi:type=
    ExperimentAutomation.Experiments.AbstractSimulation:
    FileDatasource location=\/result exportOption=CSV\/>/ $1 > $2
```



Allgemeine Applikation

Allgemeine Applikation

- Ausführung einer headless-Applikation (bspw. Build-Prozess) auch unabgängig von Palladio interessant
- Erstellen einer Beispiel-Applikation für eclipse
- kann in einem Docker-Container auf beliebiger Hardware ausgeführt werden

Dockerfile

```
FROM palladiosimulator/eclipse
COPY RunApplication.sh /usr/RunApplication.sh
RUN chmod a+rx usr/RunApplication.sh
COPY exampleapplication.jar /usr/eclipse/dropins/plugins/
    exampleapplication.jar
RUN /usr/RunApplication.sh
```

Zusammenfassung

Zusammenfassung

- Headless-Ausführung von Eclipse-basierten Applikationen in einem Docker Container
- einfache Reproduzierbarkeit von Palladio-Experiment-Automation-Experimenten

Zukünftige Arbeiten

- Abhängigkeiten von Palladio auf UI-Pakete finden und entfernen
- Example Application erweitern um die Nutzung von ins Image gegebenen Parametern
- Nutzung einer festen Version von Palladio

References I

[1] *Docker*. https://opensource.com/resources/what-docker. Aufgerufen: 18.03.2020.

[2] *Docker Image*. https://searchitoperations.techtarget.com/definition/Docker-image. Aufgerufen: 18.03.2020.

[3] *Docker Logo*. https://de.wikipedia.org/wiki/Docker_(Software)#/media/Datei:Docker_(container_engine)_logo.svg. Aufgerufen: 19.03.2020.

[4] *Docker Logo*. https://www.docker.com/company/newsroom/media-resources. Aufgerufen: 19.03.2020.

References II

[5] *Eclipse Application*. https://wiki.eclipse.org/FAQ_What_is_an_Eclipse_application%3F. Aufgerufen: 19.03.2020.

[6] *Eclipse Bundle*. https://help.eclipse.org/2019-12/index.jsp?topic=%2Forg.eclipse.platform.doc.isv%2Fguide%2Fruntime_model_bundles.htm. Aufgerufen: 19.03.2020.

[7] *Eclipse Logo*. https://www.eclipse.org/artwork/. Aufgerufen: 19.03.2020.

[8] *Github Logo*. https://github.com/logos. Aufgerufen: 19.03.2020.

[9] *P2 Provisioning Platform*. https://help.eclipse.org/2019-12/index.jsp?topic=%2Forg.eclipse.platform.doc.isv%2Fguide%2Fp2_overview.htm. Aufgerufen: 18.03.2020.

References III

[10] *Palladio Logo*.
https://sdqweb.ipd.kit.edu/wiki/Datei:Pcm-logo-stilisiert.png.
Aufgerufen: 19.03.2020.

Skripte

RunApplication.sh

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

Skripte

InstallFeature.sh

```
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"
```

Skripte

InstallLocalUpdates.sh

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

Skripte

feature.txt - 1

```
#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.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/
```

Skripte

feature.txt - 2

```
# 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/
```

Skripte

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
```

Skripte

ModifyExperimentsFile.sh

```
#!/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"
```

Skripte

RunExperimentAutomation.sh

```
#!/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"
```

Skripte

PalladioExperimentImage - RunDockerImage.sh

```
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%
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% %
```

Skripte

PalladioRunExperiment - RunDockerImage.sh

```
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
```