## BabyPort

Abschlusspräsentation (12.06.2024) Felix Schiele, Nils Heinzelmann, Marius Wörfel, Sarah Ficht

# Vision

Die Software BabyPort dient als Container Management System, zur Administration und Überwachung von Docker Containern. BabyPort bildet die verschiedenen administrativen Funktionen auf Docker Containers mittels einer UI ab.

#### Projektziele (Server-Client)

Ziele	Status (Zielerreichung)
Planung Architektur Server-Client	<b>✓</b>
Erstellung einer benutzerfreundlichen UI	<b>✓</b>
Personalisierung der UI (Propertymanager)	<b>✓</b>
Event-Handling zwischen MQTT und Controller	<b>✓</b>
Event-Handling zwischen Controller und UI	<b>∠</b>
MQTT Kommunikation zwischen Agent und Server-Client	

### Projektziele (Agent)

Ziele	Status (Zielerreichung)
Planung Architektur Agent	<b>∠</b>
Operationen auf Docker Container (CRUD)	
Wrapping von Docker Commands	
Zeitgesteuerter Log Transfer	
Eigenes Exception Handling	
Konfiguration as Code	
MQTT Kommunikation zwischen Agent und Server-Client	

#### Projektziele (Qualitätssicherung)

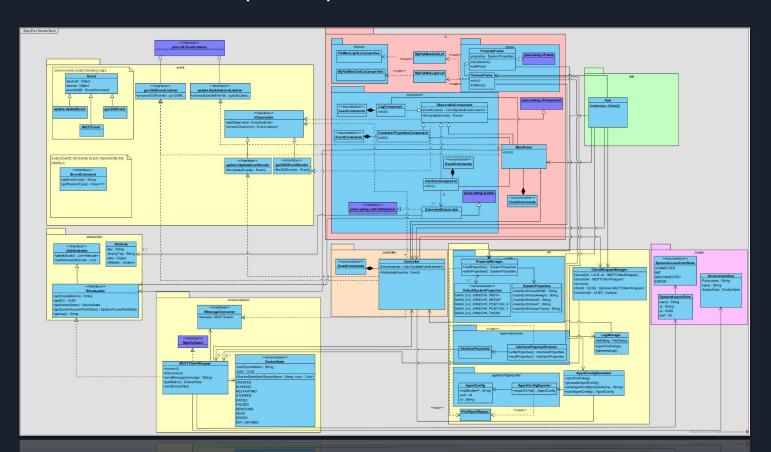
Ziele	Status (Zielerreichung)
Nutzung von CI/CD (Jenkins)	
Nutzung von statischer Code-Analyse (SonarQube)	
Nutzung von Security Scannern (Snyk)	
Deployment in dev (ehemals Sandbox) Umgebung	
Deployment in prod Umgebung	
Unit Tests	
Integration Tests	<b>✓</b>
(manuelle) End to End Tests	

### Architekturentscheidungen

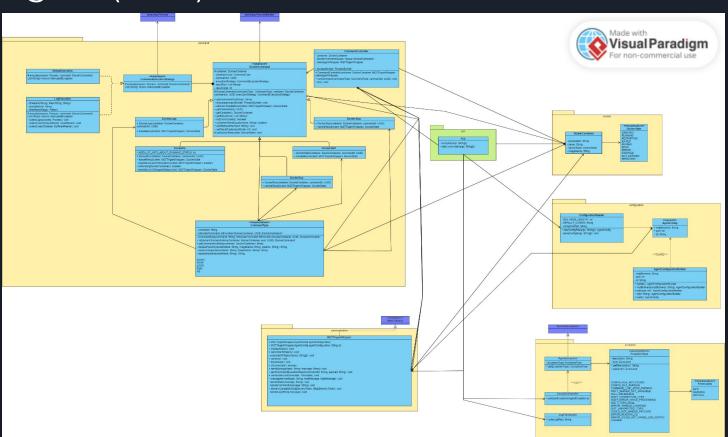
#### Architekturentscheidungen

Entscheidungen	Begründungen
Model-View-Controller	Strikte Trennung zwischen UI, Business Logic und Model
Sidecar	Eigenständige Komponente (Orientierung Kubernetes Sidecar Container)
Observer Pattern	Kommunikation der einzelnen Komponenten im Server
Strategy Pattern	Unterteilung der Ausführungsarten der Docker Commands
Command Pattern	Unterteilung der einzelnen Docker Commands

#### Server-Client (UML)



#### Agent (UML)



### Angewandte Design Patterns

#### Resumé

Type	Ziele	Status (Zielerreichung)
Pattern	Observer, Strategy, Factory, Command	<b>∠</b>
Principal	Interface segregation	V
Principal	Single responsibility	ОК
Principal	Liskov substitution principle	<b>✓</b>
Principal	KISS & DRY	OK
Architectural Decision	MVC	V
Architectural Decision	Sidecar	V

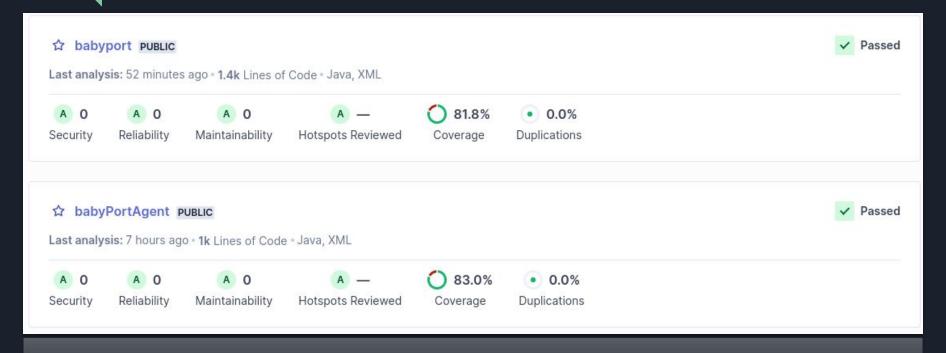
## Technology solutions

#### Genutzte Technologien

Technologie Kategorie	Name / Technologie
Programmier- / Scriptsprachen	Java, Java Swing, Bash, Docker, Groovy
Qualitätssicherungs Tools	Renovate, SonarQube, Snyk, Junit 5, Mockito
DevOps - Tools	GitLab, Jenkins
Bibliotheken	Jackson, Eclipse Paho (MQTT Client), Args4J, Flatlaf, Junit 5 (Jupiter), Mockito

## Qualitätssicherung

#### SonarQube (statische Code-Analyse)



secunity

Reliability

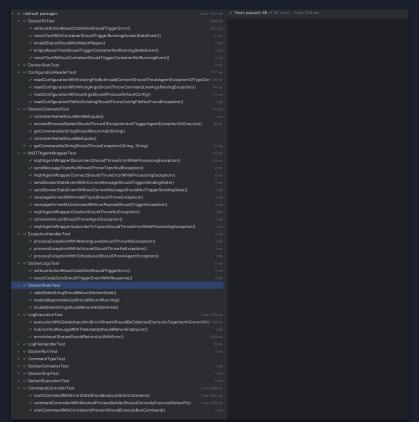
Maintainabilit

Hotspots Reviewed

coverage

Duplications

#### Tests (Unit und Integration)



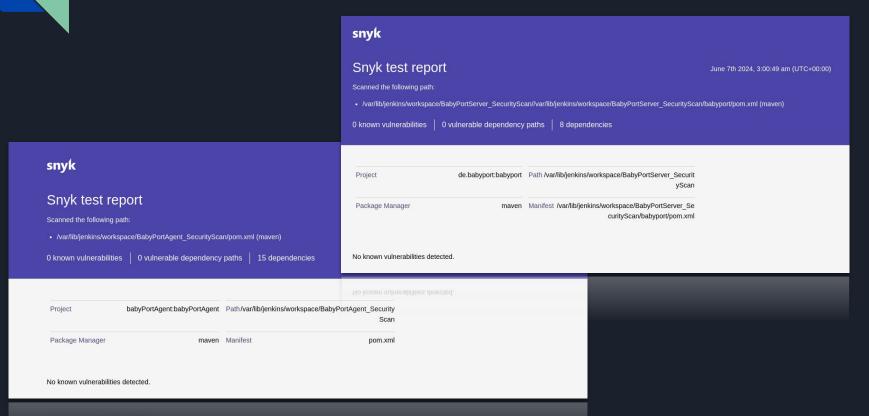
ec 611ms ---- Intellij IDEA coverage runner ----✓ mqttExceptionErrorEvent11sec 596ms tracing and tracking per test coverage ... ✓ throwableErrorEventShouldRetur8ms include patterns: ✓ parsehvaldFileShouldWriteDef; 24 ms Jun 88, 2024 1:18:51 AM de.babyport.communication.MQTTClientWrapper lamber. y guit-daringemischesdisch im 100 Treidhri-2021-9821-9821-9821 (1921) ( Spidynimicholomococcocción
 Spidynimicholomococcocción
 Spidynimicholomococcocción
 Spidynimicholomococcocción
 Spidynimicholomococcocción
 Spidynimicholomococcocción
 Spidynimicholomococcocción
 Spidynimicholomococcocción
 Spidynimicholomococcocción
 Spidynimicholomococción
 Spidynim ✓ connectWithoutBrokerShouldTrissms Jun 88, 2024 1:18:52 AM de.babyport.ui.MainGui processUpdateEvent Combination Control (Computer Control Con Medicolates/Minimization and the property of the property ✓ mqttMessageReceivedWithError10ms INFO: Incoming event in MainGui registered Event [sourcemde.babyport.ui.component.I ✓ cancelEventOfContainerPropert 40ms INFO: Incoming event in MainGui registered Mock for GUIEvent, hashCode: 2095986405 ✓ parseNoFileShouldReturnEmpty 18ms Process finished with exit code 0

Server

#### Pipeline Status



#### Snyk (Security Scanner)



## CI/CD Setup

GIT CHECKOUT main

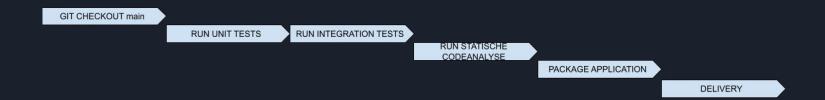
GIT CHECKOUT main

RUN UNIT TESTS

RUN INTEGRATION TESTS





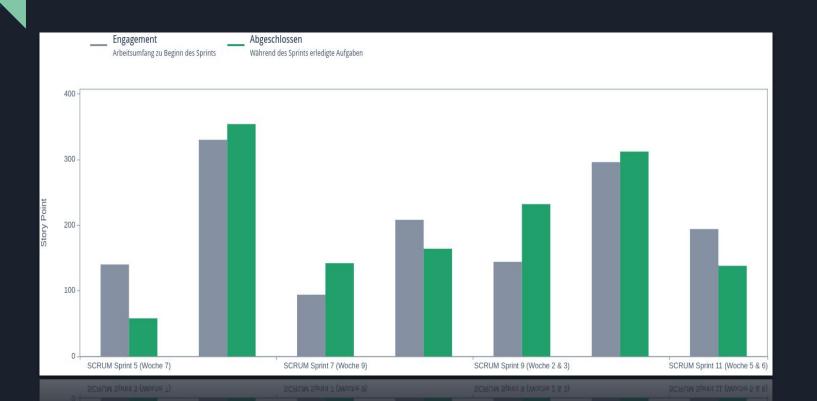


### Statistiken

#### Sprint-Burndown-Chart (exemplarisch)



#### Velocity-Bericht



#### Kumulatives Flussdiagramm



#### Lessons Learned

## Qualitätskontrolle

Die kontinuierliche Überwachung der Qualität der Arbeitsergebnisse verhindert Mängel und darauf folgende Nacharbeiten.

### Dokumentation

Eine kontinuierliche Dokumentation erleichtert die Nachverfolgung des Projektfortschritts sowie das Management bei Änderungen.

## Teamarbeit

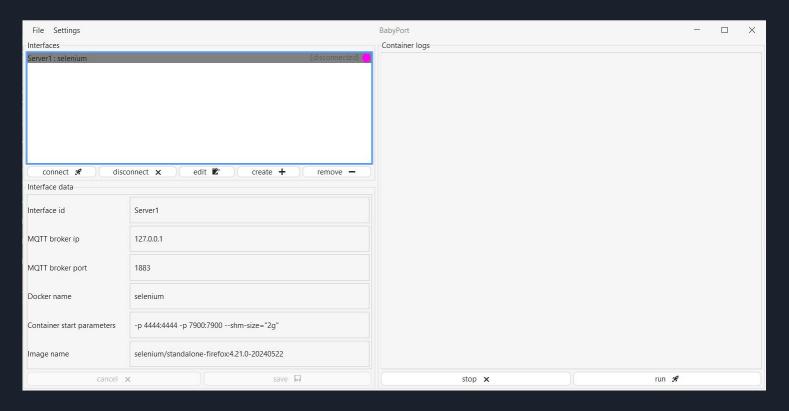
Ein gut funktionierendes Team, das effektiv zusammenarbeitet, trägt maßgeblich zum Erfolg des Projekts bei.

## Live Demo

BACKUP (nicht geplanter Teil der Präsentation)

## Live-Demo

## Step 1: Anlegen eines Containers in dem ServerClient.



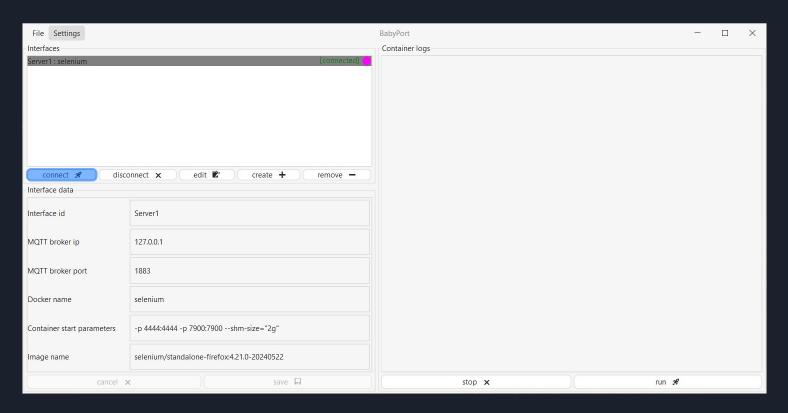
## Step 2: Ausfüllen der Agentconfig passend zum Interface im Server-Clients.

```
mqttBrokerIP: 127.0.0.1
port: 1883
id: Server1
```

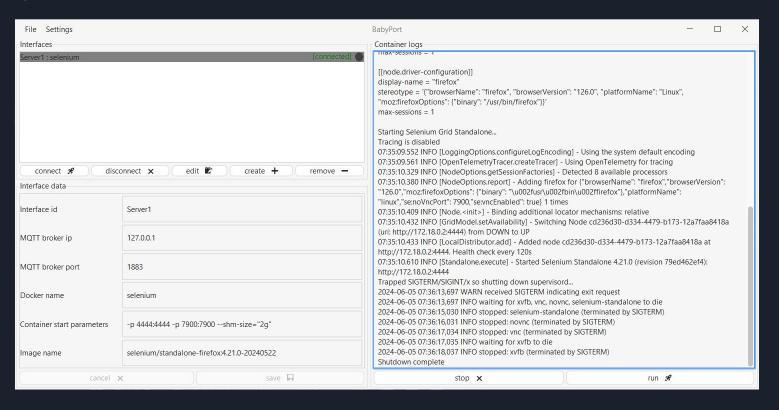
#### Step 3: Starten des Agents

```
"C:\Program Files\Eclipse Adoptium\jdk-17.0.8.101-hotspot\bin\java.exe" ...
Juni 05, 2024 9:41:00 AM app.App runAgent
INFORMATION: Agent started and connected
```

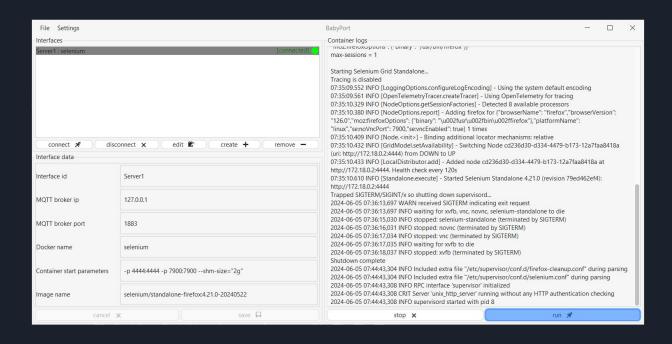
## Step 4: Connecten des Server-Clients mit dem Broker



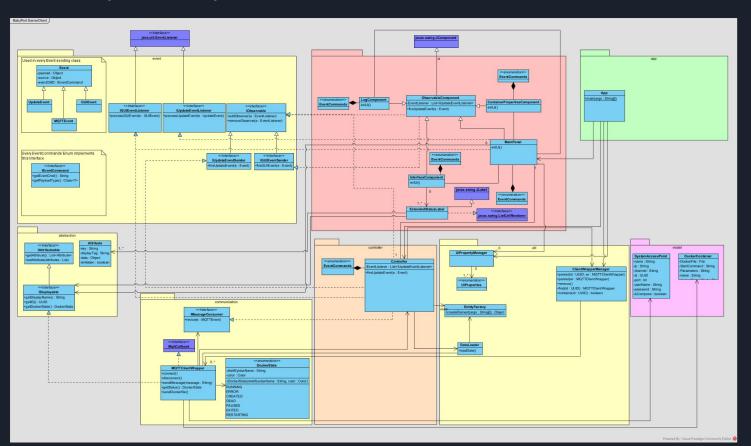
## Step 5: Server-Client lädt State und Logs nach ein paar Sekunden



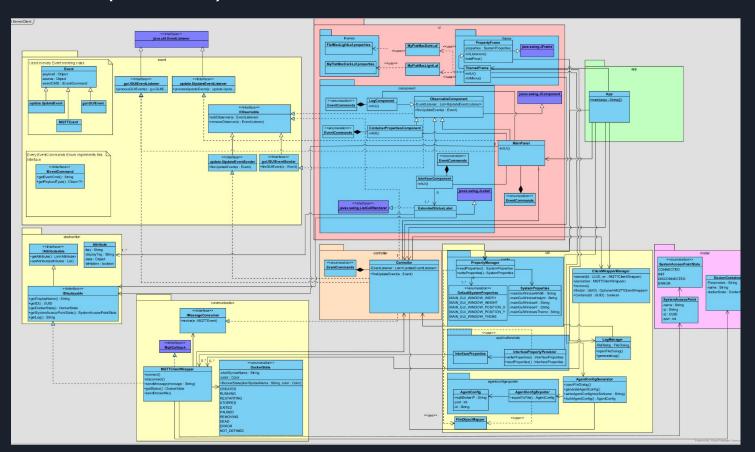
Step 6: ServerClient gibt dem Agenten den Auftrag den Container zu starten. Der Server-Client aktualisiert kurz darauf den State des Containers.



#### UML (Server) - 3. Semester



#### UML (Server) - 4. Semester



#### Burnup-Bericht (exemplarisch)

