

Continuous Testing

**regelmäßig!
automatisiert!
prüfen!**





Christian Kühn

Software - Entwickler

Gute-Laune-Verbreiter

Interessen:
Security, DevOps, Automation

chris@clowncomputing.de
@CYxChris



dmTECH GmbH

100% dm-Tochter

komplette Konzern-IT

Hardware
Software
Betrieb

Wozu Tests?

aktuellen Zustand aufzeigen

Vergleich mit Wunschzustand ermöglichen

idealerweise frühestmöglich in der Entwicklung



Achtung!?

*"Program testing can be used to show the presence of bugs,
but never to show their absence!"*

- Edsger W. Dijkstra



Continuous Integration

"In software engineering, continuous integration (CI) is the practice of merging all developers' working copies to a shared mainline several times a day"

(Wikipedia)





Continuous Integration



Continuous Testing

"Reducing wasted development time via continuous testing"

**14th. IEEE International Symposium on Software Reliability Engineering
ISSRE 2003**



Continuous Testing

Testen als Teil von Continuous Integration

Probleme früher erkennen

schneller reagieren

manuelle Tests ersetzen





Continuous Integration



Kennzahlen regelmäßig prüfen

so viele Tests wie nötig

so wenige Tests wie möglich 😇



Was kann man eigentlich testen?



Cucumber

behaviour driven development

**Entwicklungsprozess mit enger
Zusammenarbeit und
Integration aller Stakeholder**

**Fachabteilung definiert Testfälle
mit Dev-Team und Product Owner**

**"lesbare" Testbeschreibung
für Nicht-Techniker**

Spezifikation der Software wird gemeinsam ausgearbeitet



Auftraggeber (Fachabteilung, Kunde):
Beschreibung der Business-Prozesse, fachlicher Ansprechpartner

Entwicklungsteam:
Erarbeitung einer Lösungsstrategie und deren Umsetzung

Q&A-Team:
Unterstützung der Lösungsentwicklung, Perspektive Qualität: "What-if?"



Testwerkzeuge

Abbildung der Spezifikation in Form von Given - When - Then notation

Szenariogrundriss: Ein Kunde versucht sich mit seinem Passwort zu authentifizieren.

Angenommen Ein Kunde startet den Checkout-Prozess

Wenn der Kunde mit der AccountId "12345" sich mit Passwort "hunter2" authentifiziert

Dann antwortet der Auth-Service mit dem Status angemeldet



<https://www.cucumber.io>



Szenariogrundriss: Ein Kunde versucht sich mit seinem Passwort zu authentifizieren.

Angenommen Ein Kunde startet den Checkout-Prozess

Wenn der Kunde mit der AccountId "12345" sich mit Passwort "hunter2" authentifiziert

Dann antwortet der Auth-Service mit dem Status angemeldet

```
@Given("Ein Kunde startet den Checkout-Prozess")
public void prepareCheckout() {
    doPreparation();
}

@When("der Kunde mit der AccountId \"{login}\" sich mit Passwort \"{pass}\" authentifiziert")
public void authenticate(String login, String pass) {
    RequestSpecification request = given();
    addAuthorizedAuthentication(request);

    Map<Object, Object> authenticateRequest = new HashMap<>();
    authenticateRequest.put("login", login);
    authenticateRequest.put("pass", pass);
    request.contentType(ContentType.JSON).body(authenticateRequest);

    responseHolder.setHttpResponse(
        request
            .post(s: "/api/authenticate/")
            .andReturn());

    authentication.setIsAuthenticated(responseHolder.getHttpResponse().jsonPath().get("isAuthenticated"));
}

@Then("antwortet der Auth-Service mit dem Status angemeldet")
public void assertAuthentication() {
    assertThat(authentication.isAuthenticated()).isTrue();
}
```

BDD in Continuous Integration / Testing?

Ersetzung manueller Tests (Businesslogik)

Ansatz auch Test-Driven (TDD) möglich

Fehlerzustand komplexer Abläufe
für Entwickler leichter erkennbar



Gatling

continuous performance testing



in jedem Build

verschiedene Szenarien
je nach Environment oder Build

Historie vergleichen

Performance - Testing



"Load Test as Code"

Scala - basiert (Netty / Akka)

Open Source (Apache 2.0)

metric-Export*

*** (enterprise-Version oder plugin)**

<https://gatling.io>



Scenario

Beschreibung des Testfalls

```
val successfulAuthentication: ScenarioBuilder = scenario("Successful Authenticate")
    .exec(http("auth")
        .post("/api/authenticate")
        .body(StringBody("""{ "account": "12345", "pass": "hunter2" }""")).asJson
        .check(status.is(200)))

val failedAuthentication: ScenarioBuilder = scenario("Failed Authenticate")
    .exec(http("auth")
        .post("/api/authenticate")
        .body(StringBody("""{ "account": "12345", "pass": "passw0rd" }""")).asJson
        .check(status.is(401)))
```

Scenario (2)

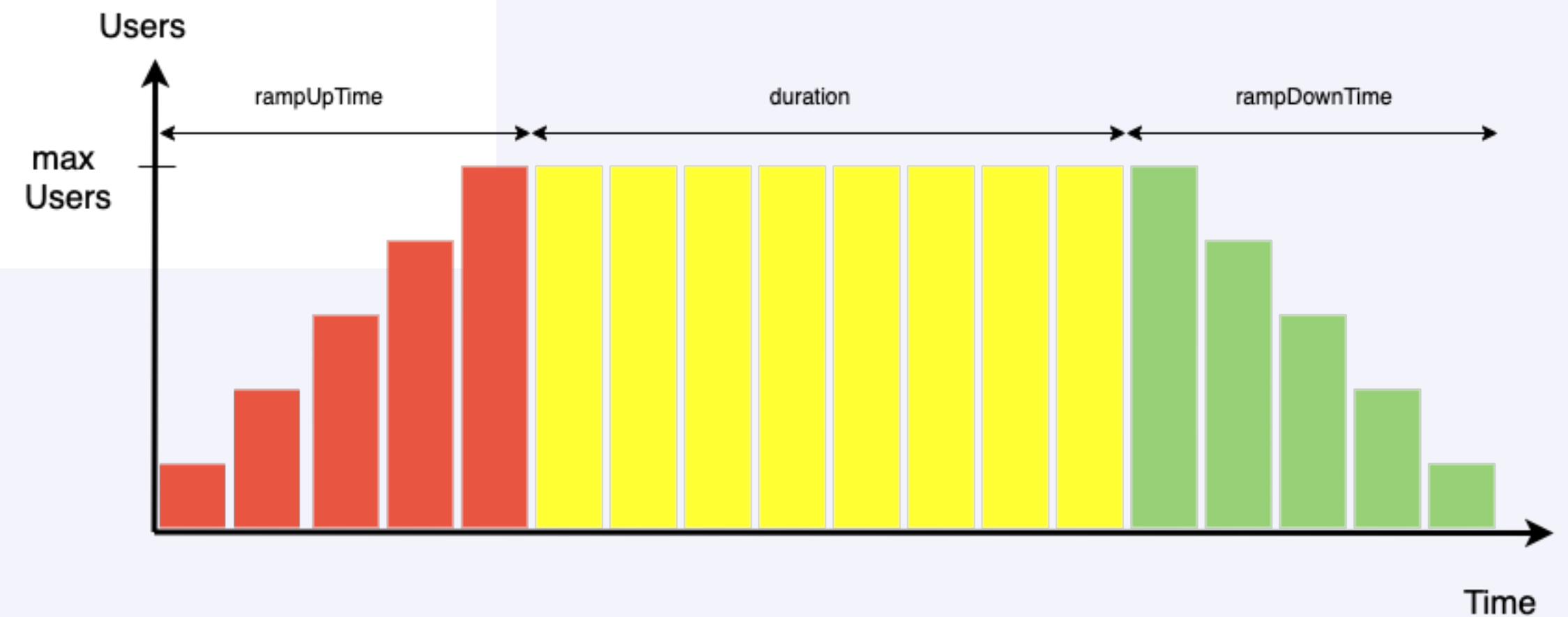
Kombination mehrerer Szenarien

```
val authenticateCombined: ScenarioBuilder = scenario("Authenticate combo")  
    .randomSwitch(  
        85.0 -> exec(successfulAuthentication),  
        15.0 -> exec(failedAuthentication))
```



Simulation (Beispiel)

```
setUp(  
  AuthenticateScenarios.authenticateCombined.inject(  
    rampUsersPerSec(0) to maxUsersPerSecond during (rampupTime seconds),  
    constantUsersPerSec(maxUsersPerSecond) during(durationTime seconds) randomized,  
    rampUsersPerSec(maxUsersPerSecond) to 0 during (rampdownTime seconds))  
).protocols(httpConfig)  
  .assertions(  
    Seq.apply(global.failedRequests.percent.lte(0.05))  
    ++ AuthenticateScenarios.timingAssertions  
  )
```



```
val httpConfig: HttpProtocolBuilder = http  
  .baseUrl("http://localhost:8080")  
  .userAgentHeader("Gatling")  
  .acceptEncodingHeader("gzip")  
httpConfig.basicAuth(user, password)
```

```
val timingAssertions: Iterable[Assertion] = Seq.apply(  
  details("auth").responseTime.percentile4.lte(1000), // 99% below 1000ms  
)
```



```
Simulation authservice.AuthServicePerfSimulation completed in 79 seconds
Parsing log file(s)...
Parsing log file(s) done
Generating reports...
```

```
----- Global Information -----
```

| | | | | |
|---------------------------------|--------|------------|------|---|
| > request count | 2085 | (OK=2085 | K0=0 |) |
| > min response time | 302 | (OK=302 | K0=- |) |
| > max response time | 3181 | (OK=3181 | K0=- |) |
| > mean response time | 1935 | (OK=1935 | K0=- |) |
| > std deviation | 841 | (OK=841 | K0=- |) |
| > response time 50th percentile | 2116 | (OK=2116 | K0=- |) |
| > response time 75th percentile | 2640 | (OK=2640 | K0=- |) |
| > response time 95th percentile | 2930 | (OK=2930 | K0=- |) |
| > response time 99th percentile | 3022 | (OK=3022 | K0=- |) |
| > mean requests/sec | 26.062 | (OK=26.062 | K0=- |) |

```
----- Response Time Distribution -----
```

| | | |
|------------------------|------|--------|
| > t < 800 ms | 316 | (15%) |
| > 800 ms < t < 1200 ms | 191 | (9%) |
| > t > 1200 ms | 1578 | (76%) |
| > failed | 0 | (0%) |

```
Reports generated in 0s.
```

```
Please open the following file: /Volumes/zDev/cy/talk/performance/gatling-demo/build,
```

```
Global: percentage of failed events is less than or equal to 0.05 : true
```

```
Global: 99th percentile of response time is less than or equal to 1200.0 : false
```

```
> Task :gatlingRun FAILED
```

```
FAILURE: Build failed with an exception.
```

Testcontainers

next level Integration testing

**realitätsnahes Testen auf
Basis von Docker-Containern**

Orchestrierung direkt aus JUnit

**komplexe Testlandschaft möglich
(microservice-Integration)**

Testcontainers



Tests gegen dritte Services

**Datenbankoperationen gegen echte
MySQL, MariaDB, PostgreSQL anstatt h2**

**UI-Tests mithilfe containerisierter Browser
Selenium-kompatibel**

<https://www.testcontainers.org/>



Wiremock

Http Simulator



**vorbereitete Responses
abhängig vom Request**

record / playback

JUnit oder Standalone

Http-Server bzw. -Services mocken

**Responses abhängig vom Request
z.B. bestimmtes Mapping auf auth-header**

Konfiguration in Java oder JSON



Consumer testen für Drittanbieter-Services

Fehler und Exceptions in Request einbauen

Latenzen simulieren (z.B. für performance-Tests)



verschiedene Zustände pro Scenario möglich

Steuerung über "Admin"-Schnittstelle

**Mock für eigene Services anbieten
als Sandbox für deren Consumer**



```
{
  "request": {
    "url": "/api/authenticate",
    "method": "POST",
    "bodyPatterns": [
      {
        "matchesJsonPath": "$[?(@.pass == 'hunter2')]"
      }
    ]
  },
  "response": {
    "status": 200,
    "body": "{ \"isAuthenticated\": \"true\" }",
    "fixedDelayMilliseconds": 350,
    "headers": {
      "Content-Type": "application/json;charset=UTF-8",
      "Cache-Control": "no-cache, no-store, max-age=0, must-revalidate",
      "Pragma": "no-cache",
      "Expires": "0",
      "Date": "{{now timezone='GMT' format='EEE, d MMM yyyy HH:mm:ss z'}}"
    }
  },
  "metadata": {
    "mapping-name": "authenticate/successfully_authenticated.json",
    "description": "Nutzer mit richtigem Passwort logt sich ein."
  }
}
```

dependency-check
software supply chain absichern

DevSecOps*

**bekannte Schwachstellen
in Abhängigkeiten finden**

**Anwendung absichern
ohne Security-Kenntnisse**

OWASP - Projekt

**Analyse von externen Abhängigkeiten (Bibliotheken)
auf bekannte Schwachstellen**





One or more dependencies were identified with known vulnerabilities in hello:

See the [dependency-check report](#) for more details.

```
[INFO] -----  
[INFO] BUILD SUCCESS  
[INFO] -----  
[INFO] Total time: 10.747 s  
[INFO] Finished at: 2020-11-22T01:56:12+01:00  
[INFO] -----
```


Published Vulnerabilities

[CVE-2018-1000873](#)

Fasterxml Jackson version Before 2.9.8 contains a CWE-20: Improper Input Validation vulnerability in Jackson-Modules-Java8 that can result in Causes a denial-of-service (DoS). This attack appear to be exploitable via The victim deserializes malicious data. The vulnerability appears to have been fixed in 2.9.8.

CWE-20 Improper Input Validation

CVSSv2:

- Base Score: MEDIUM (4.3)
- Vector: /AV:N/AC:M/Au:N/C:N/I:N/A:P

CVSSv3:

- Base Score: MEDIUM (6.5)
- Vector: /AV:N/AC:L/PR:N/UI:R/S:U/C:N/I:N/A:H

References:

- CONFIRM - https://bugzilla.redhat.com/show_bug.cgi?id=1665601
- CONFIRM - <https://security.netapp.com/advisory/ntap-20200904-0004/>
- MISC - <https://github.com/FasterXML/jackson-modules-java8/issues/90>
- MISC - <https://github.com/FasterXML/jackson-modules-java8/pull/87>
- MISC - <https://www.oracle.com/security-alerts/cpuoct2020.html>
- MISC - <https://www.oracle.com/technetwork/security-advisory/cpujul2019-5072835.html>
- MISC - <https://www.oracle.com/technetwork/security-advisory/cpuoct2019-5072832.html>
- MLIST - [\[drill-dev\] 20191017 Dependencies used by Drill contain known vulnerabilities](#)
- MLIST - [\[drill-dev\] 20191021 \[jira\].\[Created\].\(DRILL-7416\) Updates required to dependencies to resolve potential security vulnerabilities](#)
- MLIST - [\[drill-issues\] 20191021 \[jira\].\[Created\].\(DRILL-7416\) Updates required to dependencies to resolve potential security vulnerabilities](#)
- MLIST - [\[nifi-commits\] 20191113 svn commit: r1869773 - /nifi/site/trunk/security.html](#)
- MLIST - [\[nifi-commits\] 20200123 svn commit: r1873083 - /nifi/site/trunk/security.html](#)
- MLIST - [\[pulsar-commits\] 20190416 \[GitHub\]. \[pulsar\] one70six opened a new issue #4057: Security Vulnerabilities - Black Duck Scan - Pulsar v.2.3.1](#)
- N/A - [N/A](#)
- OSSINDEX - [\[CVE-2018-1000873\] Improper Input Validation](#)

Vulnerable Software & Versions: ([show all](#))

- [cpe:2.3:a:fasterxml:jackson-databind:*:*:*:*:* versions up to \(excluding\) 2.9.8](#)
- ...

[CVE-2018-14718](#)

FasterXML jackson-databind 2.x before 2.9.7 might allow remote attackers to execute arbitrary code by leveraging failure to block the slf4j-ext class from polymorphic deserialization.

CWE-502 Deserialization of Untrusted Data

CVSSv2:

- Base Score: HIGH (7.5)
- Vector: /AV:N/AC:L/Au:N/C:P/I:P/A:P

CVSSv3:

- Base Score: CRITICAL (9.8)
- Vector: /AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

References:

- BID - [106601](#)
- BUGTRAQ - [20190527 \[SECURITY\].\[DSA 4452-1\] jackson-databind security update](#)
- CONFIRM - <https://github.com/FasterXML/jackson-databind/commit/87d29af25e82a249ea15858e2d4ecbf64091db44>
- CONFIRM - <https://github.com/FasterXML/jackson-databind/issues/2097>
- CONFIRM - <https://github.com/FasterXML/jackson/wiki/Jackson-Release-2.9.7>
- CONFIRM - <https://security.netapp.com/advisory/ntap-20190530-0003/>
- CONFIRM - <https://www.oracle.com/technetwork/security-advisory/cpujan2019-5072801.html>
- DEBIAN - [DSA-4452](#)
- MISC - <https://www.oracle.com/security-alerts/cpujan2020.html>

**Erkennen von
technischen Fehlern
und Bugs**

SonarQube
statische code-Analyse

automatisches Code Review

Quality Gate



Analyse-Ergebnis eines Pull-Requests

| QUALITY GATE STATUS ? | FAILED CONDITIONS | MEASURES |
|---|--|---|
| <div>Conditions on New Code ?</div> <div>Failed</div> | <div><div>C</div><div>Reliability Rating on New Code is worse than A</div></div> <div><div>B</div><div>Maintainability Rating on New Code is worse than A</div></div> <div><div>71.03%</div><div>Coverage on New Code is less than 80.0%</div></div> | <div><div>1</div><div>New Bugs</div><div>Reliability <div>C</div></div></div> <div><div>0</div><div>New Vulnerabilities</div><div>Security <div>A</div></div></div> <div><div>0</div><div>New Security Hotspots</div><div>Security Review <div>A</div></div></div> <div><div>32</div><div>New Code Smells</div><div>Maintainability <div>B</div></div></div> <div><div><div></div></div><div><div>71.03%</div><div>Coverage on <div>73</div> New Lines to cover</div></div><div><div>17.9%</div><div>Estimated after merge</div></div></div> <div><div><div></div></div><div><div>2.94%</div><div>Duplications on <div>272</div> New Lines</div></div><div><div>14.5%</div><div>Estimated after merge</div></div></div> |

Ergebnis als Kommentar am Merge-Request



SonarQube Code Analysis

Quality Gate failed

Failed





- ✖ C Reliability Rating on New Code (is worse than A)
- ✖ B Maintainability Rating on New Code (is worse than A)
- ✖ 71.0% Coverage on New Code (is less than 80%)

[See analysis details on SonarQube](#)

Additional information

The following metrics might not affect the Quality Gate status but improving them will improve your project code quality and security.

33 Issues

-  C 1 Bug
-  A 0 Vulnerabilities
-  A 0 Security Hotspots
-  B 32 Code Smells

Coverage and Duplications

- 71.0% Coverage (17.9% Estimated after merge)
- 2.9% Duplication (14.5% Estimated after merge)



Erklärung und Beispiele

> Scope

> Resolution

> Status

> Security Category

> Creation Date NEW CODE Clear

> Language

> Rule

> Tag

> Directory

> File

> Assignee

Code Smell

Major

Open

Not assigned

5min effort

Comment

performance

Format specifiers should be used instead of string concatenation.

Why is this an issue?

Code Smell

Major

Open

Not assigned

10min effort

Comment

cert, confusing

Directly append the argument of String.valueOf(). Why is this an issue?

Code Smell

Minor

Open

Not assigned

5min effort

Comment

clumsy

Catch Exception instead of Throwable. Why is this an issue?

Code Smell

Major

Open

Not assigned

20min effort

Comment

bad-practice, cert, cppcoreguidelines, ...

Format specifiers should be used instead of string concatenation.

Why is this an issue?

Code Smell

Major

Open

Not assigned

10min effort

Comment

cert, confusing

Throwable and Error should not be caught

java:S1181

Code Smell

Major

bad-practice, cert, cppcoreguidelines, ...

Available Since Jan 13, 2021

SonarQube (Java)

Constant/issue: 20min

Throwable

 is the superclass of all errors and exceptions in Java.

Error

 is the superclass of all errors, which are not meant to be caught by applications.

Catching either

Throwable

 or

Error

 will also catch

OutOfMemoryError

 and

InternalError

 , from which an application should not attempt to recover.

Noncompliant Code Example

```
try { /* ... */ } catch (Throwable t) { /* ... */ }
try { /* ... */ } catch (Error e) { /* ... */ }
```

MEHR!

Infrastruktur

Monitoring

"FinOps"

Security

UX/UI

Compliance

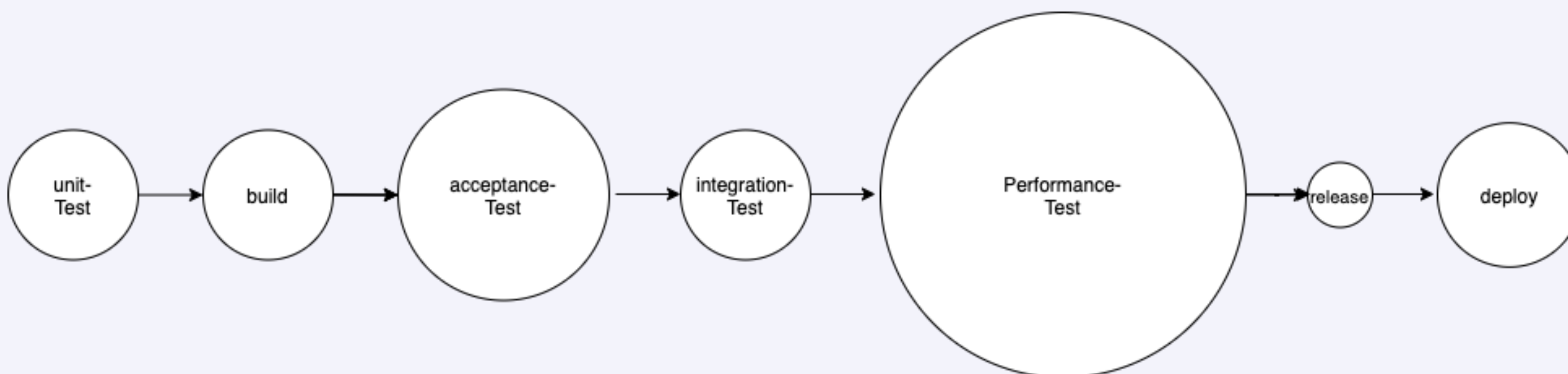
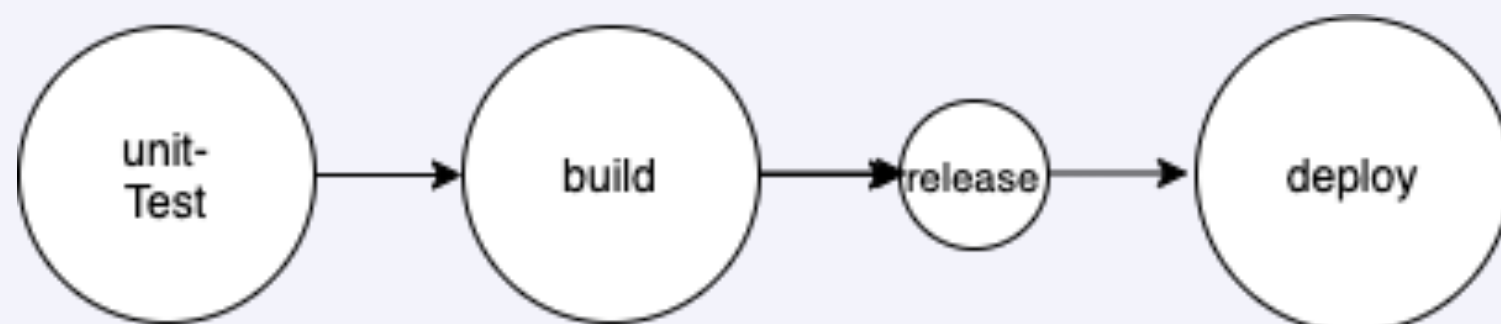
Architektur



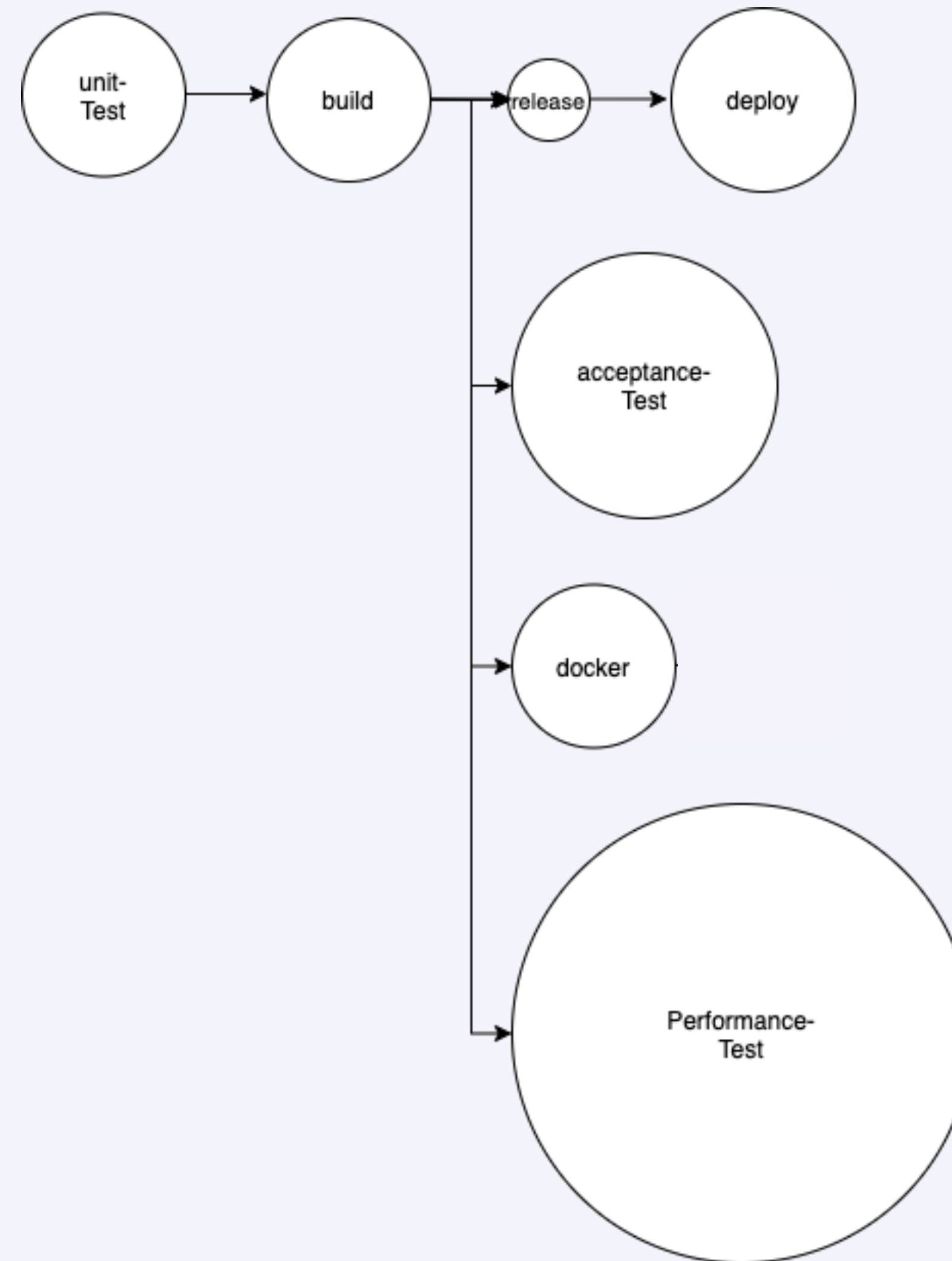
was muss man noch beachten?



build-Dauer!



parallelisieren oder asynchron testen



Vorteile Continuous Testing

Tests werden nicht vergessen

Tests werden immer gleich ausgeführt

Test laufen im Hintergrund



Nachteile Continuous Testing

- \ (ツ) / -



Nachteile Continuous Testing

Zeitaufwand zur Erstellung der Tests

Kosten für Testinfrastruktur, Lizenzen, Skills



Danke für euer Interesse! :)

Fragen?

A decorative graphic in the bottom left corner consisting of several overlapping squares and triangles in shades of light blue and white, with a small white circle in the center.

Freue mich über Feedback chris@clowncomputing.de  [@CYxChris](https://twitter.com/CYxChris)