

Balanced Banana

A Distributed Task Scheduling System

Qualitätssicherung

Niklas Lorenz, Thomas Häuselmann, Rakan Zeid Al Masri,
Christopher Lukas Homberger und Jonas Seiler

29. März 2020

Inhaltsverzeichnis

| | | |
|----------|---|----------|
| 1 | Einleitung | 1 |
| 2 | Testabdeckung | 1 |
| 2.1 | Klassenüberdeckung | 1 |
| 2.2 | Integrationstests | 3 |
| 3 | Umgesetzte Funktionale Anforderungen | 4 |
| 4 | Testen des Systems | 4 |
| 4.1 | Aufsetzen des Systems | 4 |
| 4.2 | Konklusion | 5 |

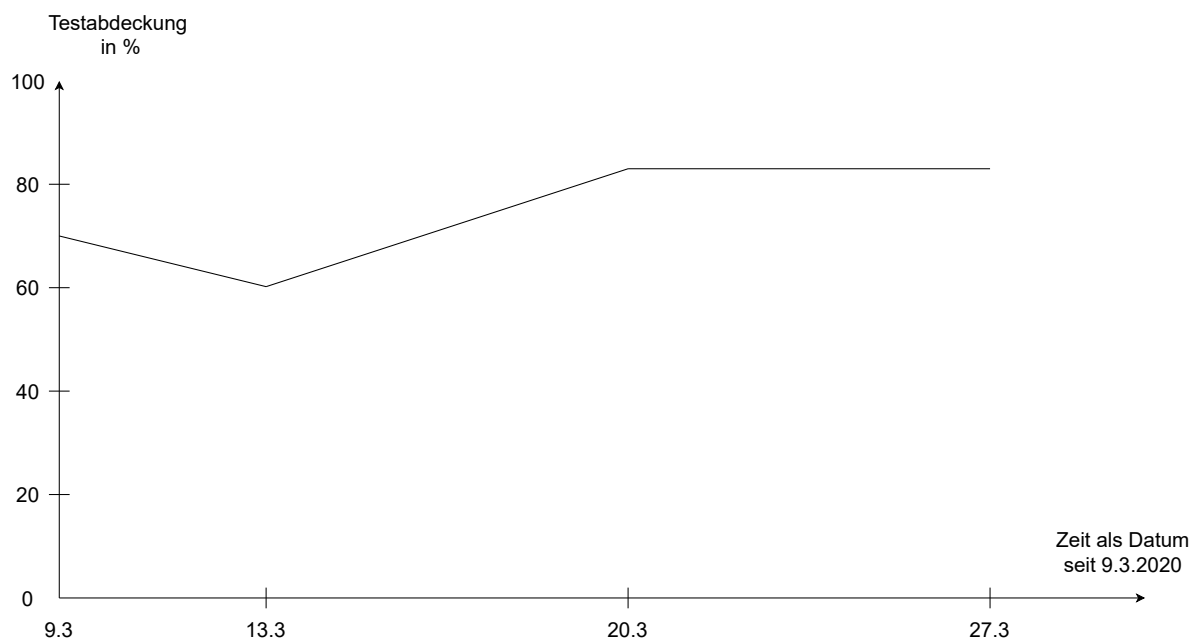
1 Einleitung

Dieses Dokument dient als Übersicht der Qualitätssicherung. Erläutert werden sowohl unsere Methoden zur Qualitätssicherung als auch das Ergebnis dieser. Ebenfalls beinhaltet dieses Dokument eine Anleitung zum Aufsetzen und Benutzen des endgültigen Programms.

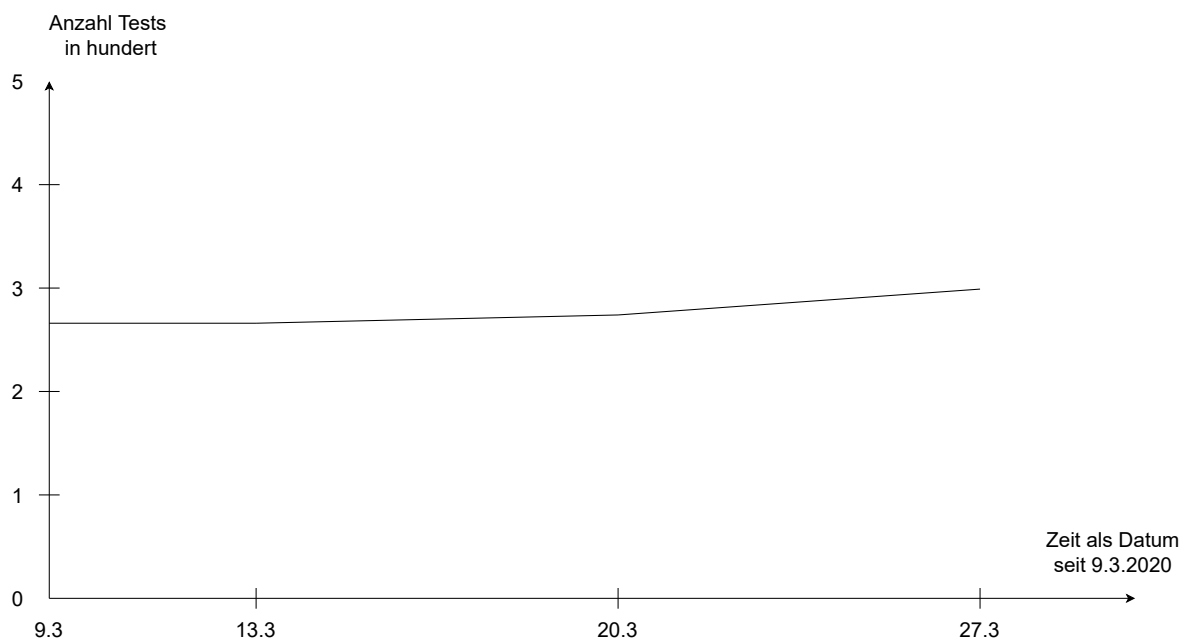
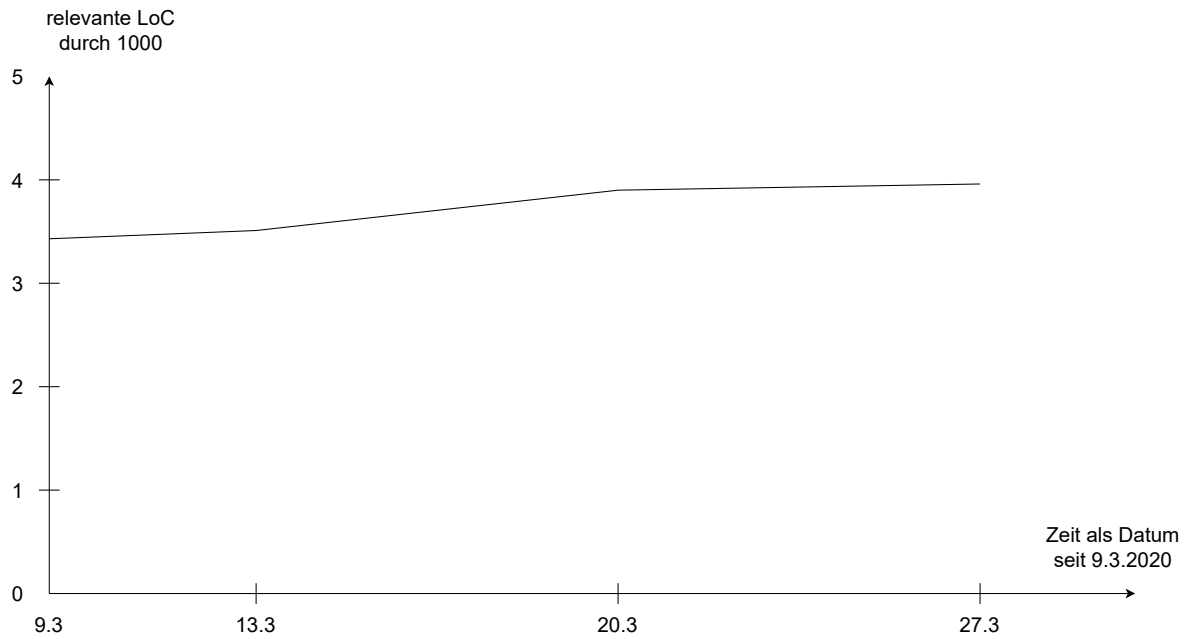
2 Testabdeckung

2.1 Klassenüberdeckung

Die Testüberdeckung der einzelnen Klassen ist in der ersten Woche der Qualitätssicherung gefallen, dies liegt daran, dass einige Klassen zwar intern funktionieren, sie jedoch im Zusammenspiel mit anderen nicht. In der ersten Woche wurden also hauptsächlich fehlerhafte Schnittstellen oder andere Fehler beseitigt und entsprechende Unit Tests erst in den folgenden Wochen eingeführt.



Entsprechend sehen wir einen Anstieg in Lines of Code (LOC) sowie der Anzahl Tests.



Da der Großteil des Codes bereits implementiert ist, sowie Integrationstests eher länger dafür aber weniger als Unittests sind, steigt auch die Anzahl der Tests nicht stark.

2.2 Integrationstests

Für das Automatisierte Testen des ganzen Systems haben wir ein Skript in Python benutzt, dieses beinhaltete diverse Szenarien und prüfte mit Positiv- als auch mit Negativtests. Neben diesem haben wir auch einiges "von Hand" getestet. Die, im Pflichtenheft definierten, Testfälle werden wie folgt abgedeckt:

| T | Kurzbeschreibung | Wie getestet? |
|-----|---|---------------|
| T1 | Verbinden des Clients mit Server | Automatisch |
| T2 | Festlegen von Prioritäten | Manuell |
| T3 | Festlegen des Betriebssystems | Manuell |
| T4 | Abfragen eines Aufgabenstatus | Automatisch |
| T5 | Benachrichtigung bei Abschluss einer Aufgaben | Manuell |
| T6 | Erstellen von Sicherungen | Manuell |
| T7 | Anforderung von Ausgabe | Manuell |
| T8 | Abbrechen von zu langen Aufgaben | - |
| T9 | Manuelles Stoppen von Aufgaben | Manuell |
| T10 | Manuelle Sicherung von Aufgaben | Manuell |

In den Integrationstests haben wir hauptsächlich Verhalten getestet die so im Pflichtenheft noch nicht voraussagbar waren, daher haben wir tatsächlich mehr Tests als nur durch die vordefinierten ersichtlich.

3 Umgesetzte Funktionale Anforderungen

| FA | Kurzbeschreibung | Implementiert? |
|------|---|----------------|
| FA1 | Client verbindet sich mit Server | Ja |
| FA2 | Benutzer authentifiziert sich | Ja |
| FA3 | Benutzer kann Aufgaben einreihen | Ja |
| FA4 | Benutzer kann Parameter übergeben | Ja |
| FA41 | Client speichert Parameter in Config | Ja |
| FA42 | Benutzer kann Config übergeben | Ja |
| FA43 | Benutzer kann Priorität festlegen | Ja |
| FA44 | Benutzer kann min und max Cores festlegen | Ja |
| FA45 | Benutzer kann min und max RAM festlegen | Ja |
| FA46 | Benutzer kann Betriebssystem festlegen | Ja |
| FA47 | Benutzer kann angeben ob Client blockiert | Ja |
| FA48 | Benutzer übergibt Pfad zu Aufgabe | Ja |
| FA49 | Es können Standardwerte benutzt werden | Ja |
| FA5 | Benutzer kann Status von Aufgabe einsehen | Ja |
| FA6 | Benutzer bekommt Email bei Abschluss | Ja |
| FA7 | Server erstellt Backups | (Nein) |
| FA8 | Benutzer kann Ausgabe anfordern | Ja |
| FA9 | Benutzer kann Statistiken abfragen | Ja |
| OFA1 | Benutzer kann Restzeit einsehen | Nein |
| OFA2 | Server stoppt lange Aufgaben | Nein |
| OFA3 | Benutzer kann manuell Aufgaben stoppen | Nein |
| OFA4 | Benutzer kann manuell Aufgaben sichern | (Nein) |
| OFA5 | Benutzer kann Pausierbarkeit angeben | (Nein) |

Für Anforderungen mit einem (Nein) existiert zwar in manchen Teilen des Programms die Funktionalität dafür, die Funktion selber ist jedoch nicht umgesetzt oder nutzbar.

4 Testen des Systems

4.1 Aufsetzen des Systems

Welche Umgebung hatten wir, was haben wir im Endeffekt getestet (zb das aufgesetzte Netz von Florian, eine Woche Testbetrieb)

- Herunterladen des Programms unter
<https://github.com/balancedbanana/balancedbanana/releases>
- Einrichten der MySql Datenbank mit

```
cat balancedbanana.sql | sudo mysql
```

Die SQL Datei ist im Archiv unter
`share/balancedbanana/balancedbanana.sql`

zu finden.

Nun ist die Datenbank unter dem Schema balancedbanana installiert. In der MySQL-Konsole einen Datenbank Nutzer für balancedbanana anlegen mit folgenden MySQL-Befehlen:

```
CREATE USER 'balancedbanana'@'localhost' IDENTIFIED BY
'balancedbanana';

GRANT ALL PRIVILEGES ON balancedbanana.* TO 'balancedbanana'@'localhost';

FLUSH PRIVILEGES;

exit
```

- Configfiles pro Benutzer anlegen unter \$HOME/.bbc bzw .bbs oder .bbd.
Dazu einfach die Configfiles von share/balancedbanana/.bbc bzw .bbs oder .bbd kopieren.
- Das Passwort balancedbanana nach IDENTIFIED BY bei Bedarf anpassen und in \$HOME/.bbs/appconfig.ini in Zeile databasepassword anpassen.

4.2 Konklusion

Wie lief es, Leistung, entdeckte Bugs, etc.