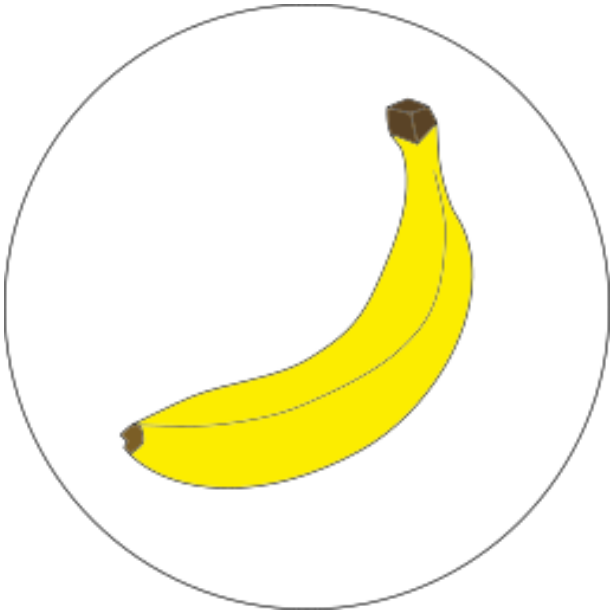


Balanced Banana

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

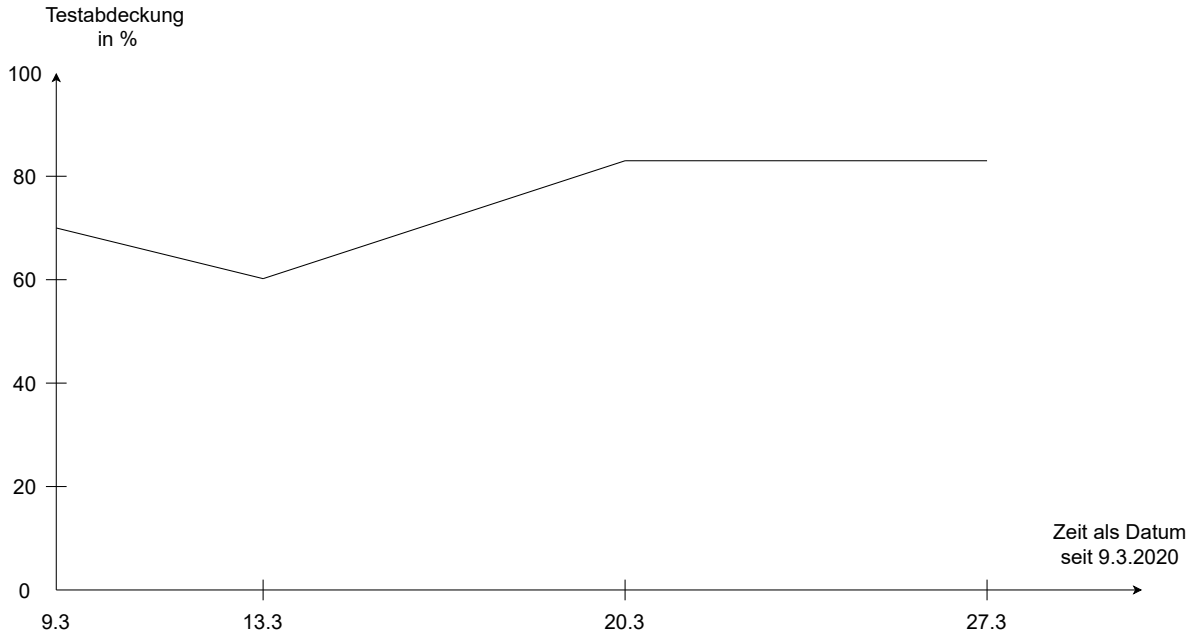
1 Einleitung

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.

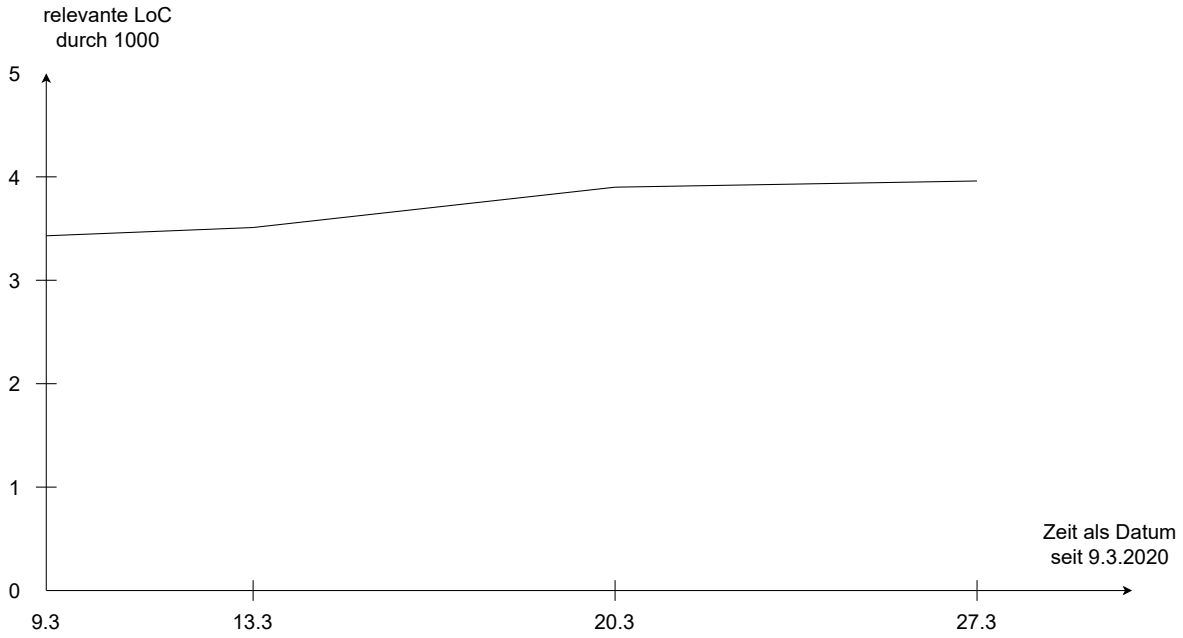
2 Testabdeckung

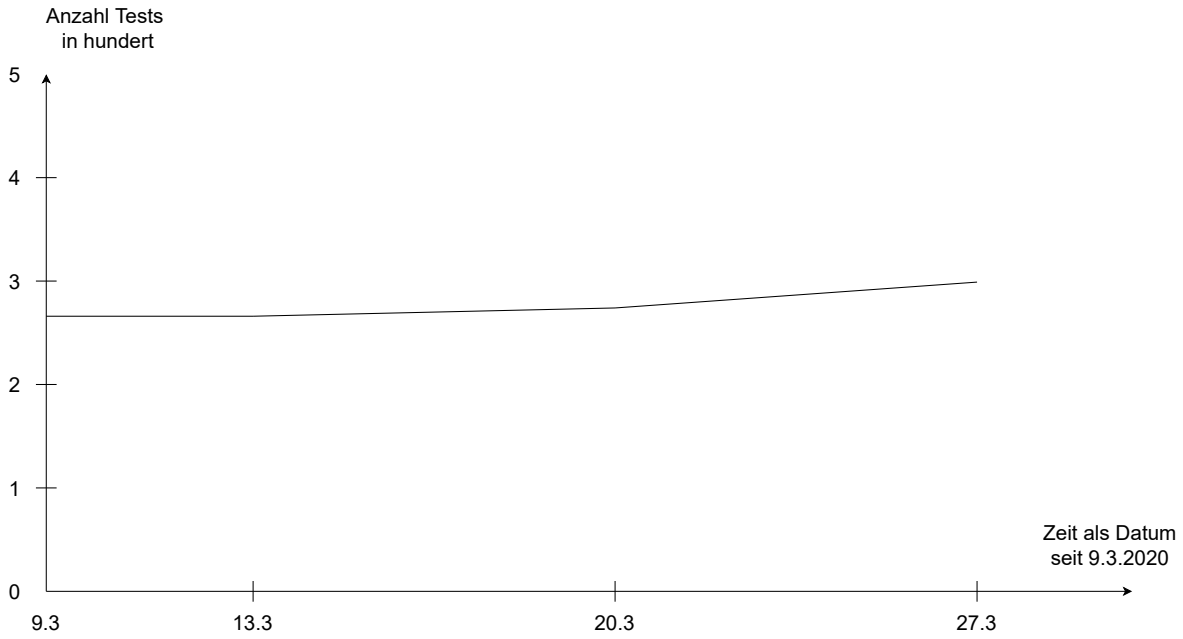
2.1 Klassenüberdeckung

Die Testüberdeckung der einzelnen Klassen ist in der ersten Woche der Qualitätssicherung gefallen, dies liegt daran, dass einige Klassen zwar intern funktioneren, 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?	Überdeckt von?
FA1	Client verbindet sich mit Server	Ja	T1
FA2	Benutzer authentifiziert sich	Ja	Integrationstest
FA3	Benutzer kann Aufgaben einreihen	Ja	T2 & T3
FA4	Benutzer kann Parameter übergeben	Ja	T2 & T3
FA41	Client speichert Parameter in Config	Ja	Manuelltest
FA42	Benutzer kann Config übergeben	Ja	Manuelltest
FA43	Benutzer kann Priorität festlegen	Ja	T2
FA44	Benutzer kann min und max Cores festlegen	Ja	Manuelltest
FA45	Benutzer kann min und max RAM festlegen	Ja	Manuelltest
FA46	Benutzer kann Betriebssystem festlegen	Ja	T3
FA47	Benutzer kann angeben ob Client blockiert	Ja	Manuelltest
FA48	Benutzer übergibt Pfad zu Aufgabe	Ja	Integrationstest
FA49	Es können Standardwerte benutzt werden	Ja	Integrationstest
FA5	Benutzer kann Status von Aufgabe einsehen	Ja	T4
FA6	Benutzer bekommt Email bei Abschluss	Ja	T5
FA7	Server erstellt Backups	(Nein)	T6
FA8	Benutzer kann Ausgabe anfordern	Ja	T7
FA9	Benutzer kann Statistiken abfragen	Ja	Manuell
OFA1	Benutzer kann Restzeit einsehen	Nein	-
OFA2	Server stoppt lange Aufgaben	Nein	T8
OFA3	Benutzer kann manuell Aufgaben stoppen	Nein	T9
OFA4	Benutzer kann manuell Aufgaben sichern	(Nein)	T10
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 Benutzeranleitung

4.1 Aufsetzen des Schedulers

Zuerst müssen wir das System aufsetzen, das Programm wird in einem Paket geliefert, die entsprechende Funktionen finden sich dann in den einzelnen ausführbaren Dateien.

- 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.

Nun ist der Scheduler aufgesetzt.

4.2 Konklusion

Wie lief es, Leistung, entdeckte Bugs, etc.