Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Automatisierte Bewertung von Java-Programmieraufgaben im Rahmen einer Moodle E-Learning Plattform

Niels Gandraß <Niels.Gandrass@haw-hamburg.de>
Axel Schmolitzky <Axel.Schmolitzky@haw-hamburg.de>

08. Oktober 2019

4. Workshop "Automatische Bewertung von Programmieraufgaben" 2019, Essen

MINTFIT HAMBURG
Eine Initiative der Hamburger Hochschulen

Hamburg | Behörde für Wissenschaft, Forschung und Gleichstellung

Übersicht

1. Motivation, Problemstellung und Zielsetzung
2. Entwickelter Moodle-Fragetyp
3. Einsatz an der HAW Hamburg

Einleitung

Motivation

Einsatz von Online-Aufgaben in der einführenden Programmierlehre.

Schulung der *Lese-* und *Schreibkompetenz* für Java-Quellcode.

viaMINT-Plattform

Interaktive Vorkursangebote

Semesterbegleitende E-Assessment Inhalte

Problemstellung

Probleme mit bisherigen Moodle-Plugins (u. a. `javaunittest` [Be16] und `sojunit` [Öz08]) zur Bewertung von Java-Quellcode:

- Fehlende Unterstützung benötigter Funktionalitäten
- Feedback nur in Form der rohen JUnit-Ausgaben
- Signaturprüfung unzureichend bzw. fehleranfällig
- Geringe Effizienz / lange Auswertungsdauer
- Momentan nicht aktiv gepflegt

Zielsetzung

Zur Umsetzung von Java-*Schreibkompetenz*-Aufgaben haben wir u. a. folgende Anforderungen an einen Moodle-Fragetypen identifiziert:

Anforderungen an Moodle-Fragetyp

- Mehrere virtuelle Dateien / Eingabepuffer
- Statische Prüfung der Code-Signatur inkl. Generics
- Erweiterte Feedbackmöglichkeiten
- Sperren und Verbergen (komplett / nur Javadoc) von Codeblöcken
- Performante und parallele Auswertung von Tests
- Vollständige Isolation der ausgeführten Aufgaben

Entwickelter Moodle-Fragetyp

`qtype_junittest`

Beispielaufgaben

Frage **1**

Unvollständig

Erreichbare Punkte: 1,00

Schreibe eine Methode `public int maximum(int a, int b, int c)`, die den größten der drei übergebenen `int`-Werte zurückliefert.

Beispiel: `a = 1; b = 42; c = 21;`
Rückgabewert: `42`

ServiceMix

```
1  class ServiceMix {
2
3      /**
4       * Ermittelt das Maximum (den größten) der 3 Parameter
5       * und gibt diesen zurück.
6       *
7       * @param a Erste zu vergleichende Zahl
8       * @param b Zweite zu vergleichende Zahl
9       * @param c Dritte zu vergleichende Zahl
10      * @return Größte der drei Zahlen
11      */
12     public int maximum(int a, int b, int c) {
13         // Hier Code einfügen
14     }
15
16 }
```

Prüfen

Abbildung 1: Einfache Beispielfrage im Moodle-LMS

Beispielaufgaben

Korrekte Antwort! Alle Tests erfolgreich bestanden.

Richtig
Bewertung für diese Einreichung: 1,00/1,00.

Teilweise korrekte Antwort. 1 von insgesamt 3 Tests erfolgreich bestanden.

Fehlgeschlagene Tests:

testMaximumSimpel Fehler bei Aufruf mit den Parametern: `maximum(1, 3, 2)`, Erwartungswert: `3`, Ergebniswert: `1`

testMaximumNegativ Fehler bei Aufruf mit den Parametern: `maximum(-2, -1, -3)`, Erwartungswert: `-1`, Ergebniswert: `-2`

Teilweise richtig
Bewertung für diese Einreichung: 0,33/1,00.

Der Quellcode entspricht nicht der erwarteten Signatur.

Die Methode `maximum(int, int, int)` konnte nicht in der Klasse oder dem Interface `ServiceMix` gefunden werden

Falsch
Bewertung für diese Einreichung: 0,00/1,00.

Abbildung 2: Auszug der Feedbacks zur einfachen Beispielfrage

Beispielaufgaben

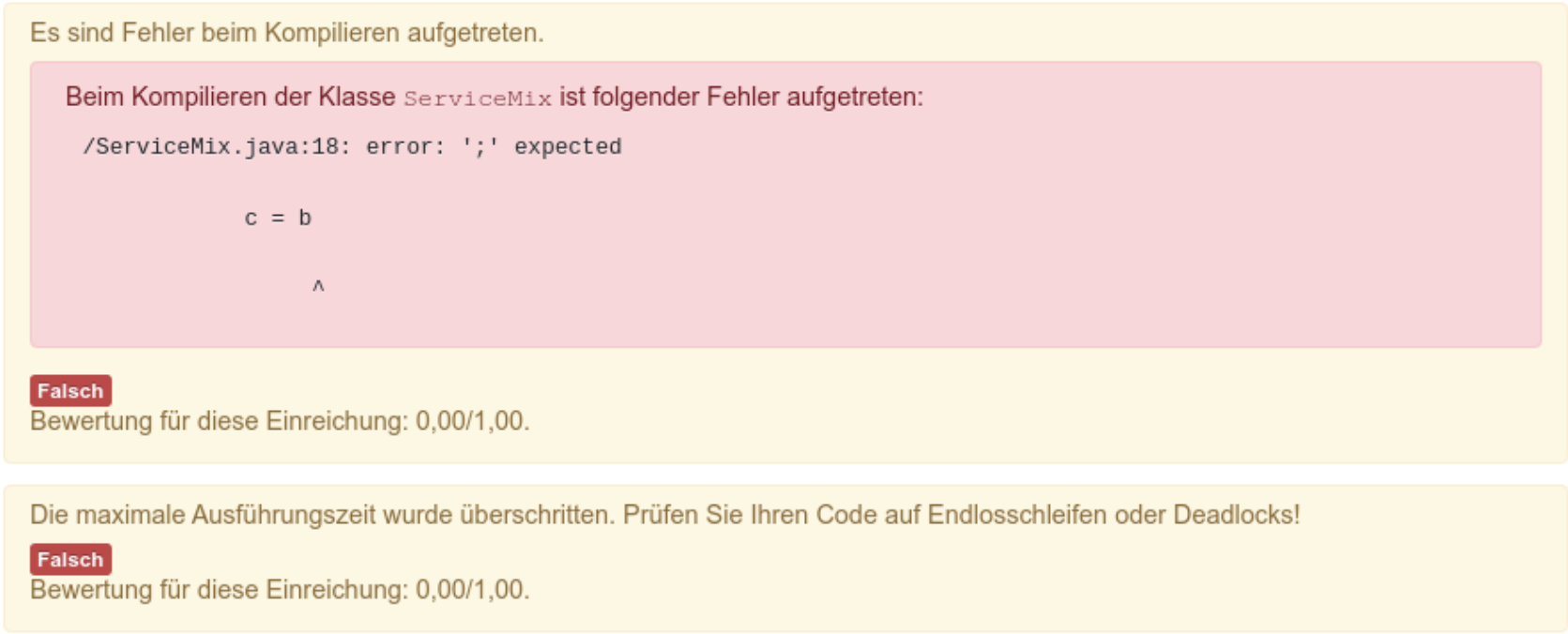

Abbildung 2: Auszug der Feedbacks zur einfachen Beispielfrage

Beispielaufgaben

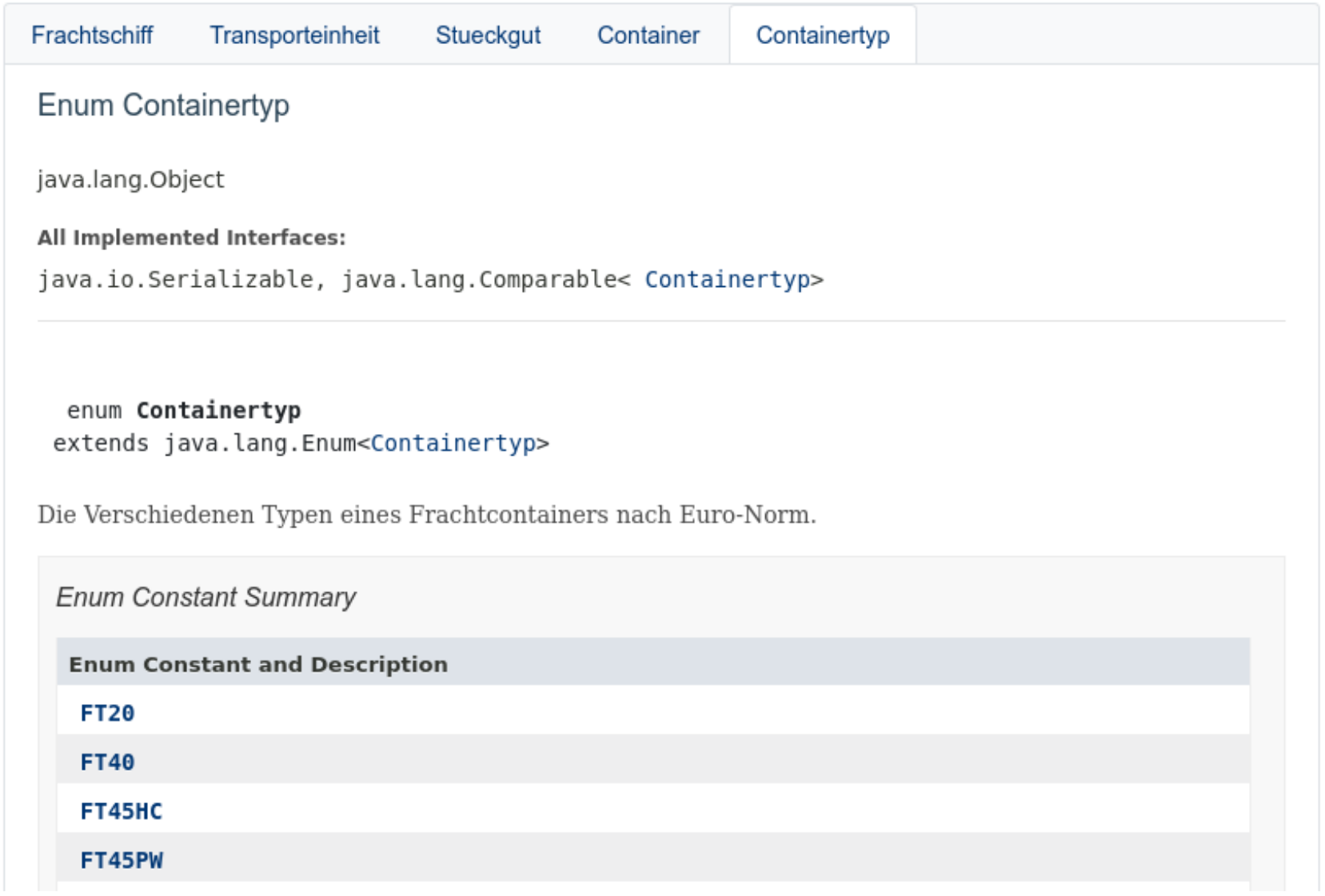

Abbildung 3: Javadoc-Ansicht eines bereitgestellten Enums

Aufbau des Fragetyps

Aufteilung in Moodle-Plugin und eigenständigen Webservice zur Durchführung der JUnit-Tests inkl. statischer Signaturprüfung.

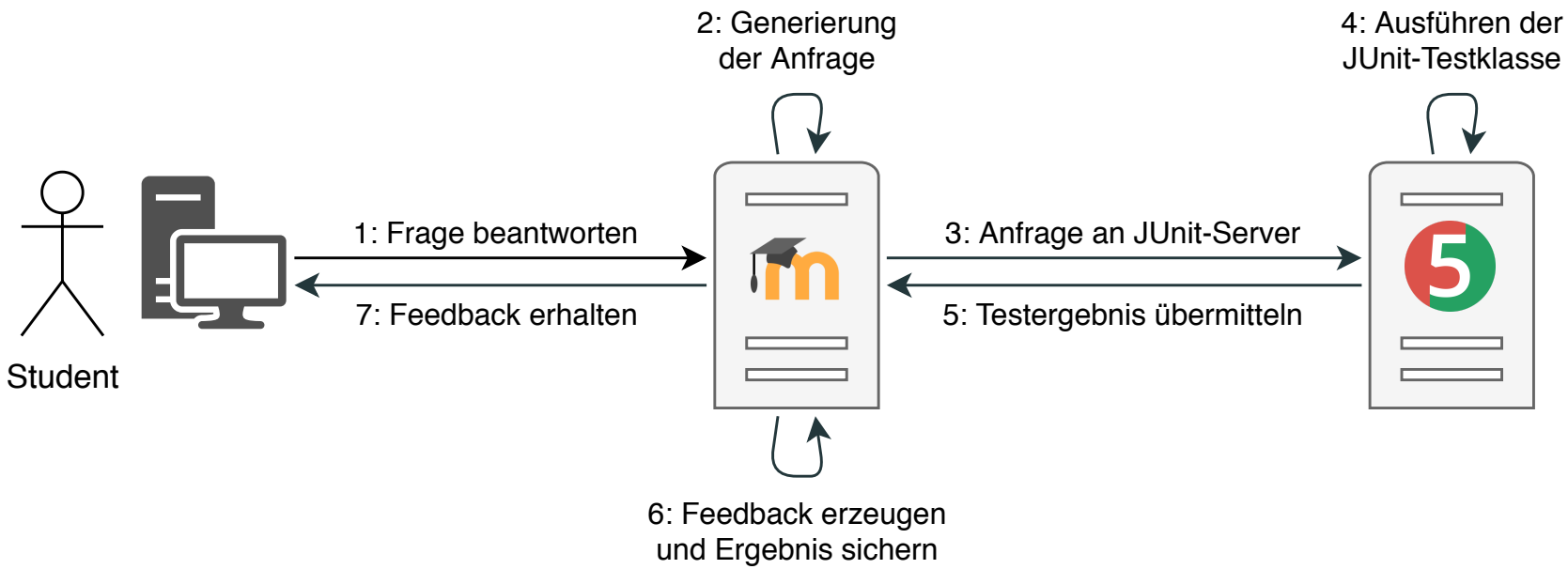

Abbildung 4: Interaktion der Komponenten des Fragetyps

Bewertungsablauf

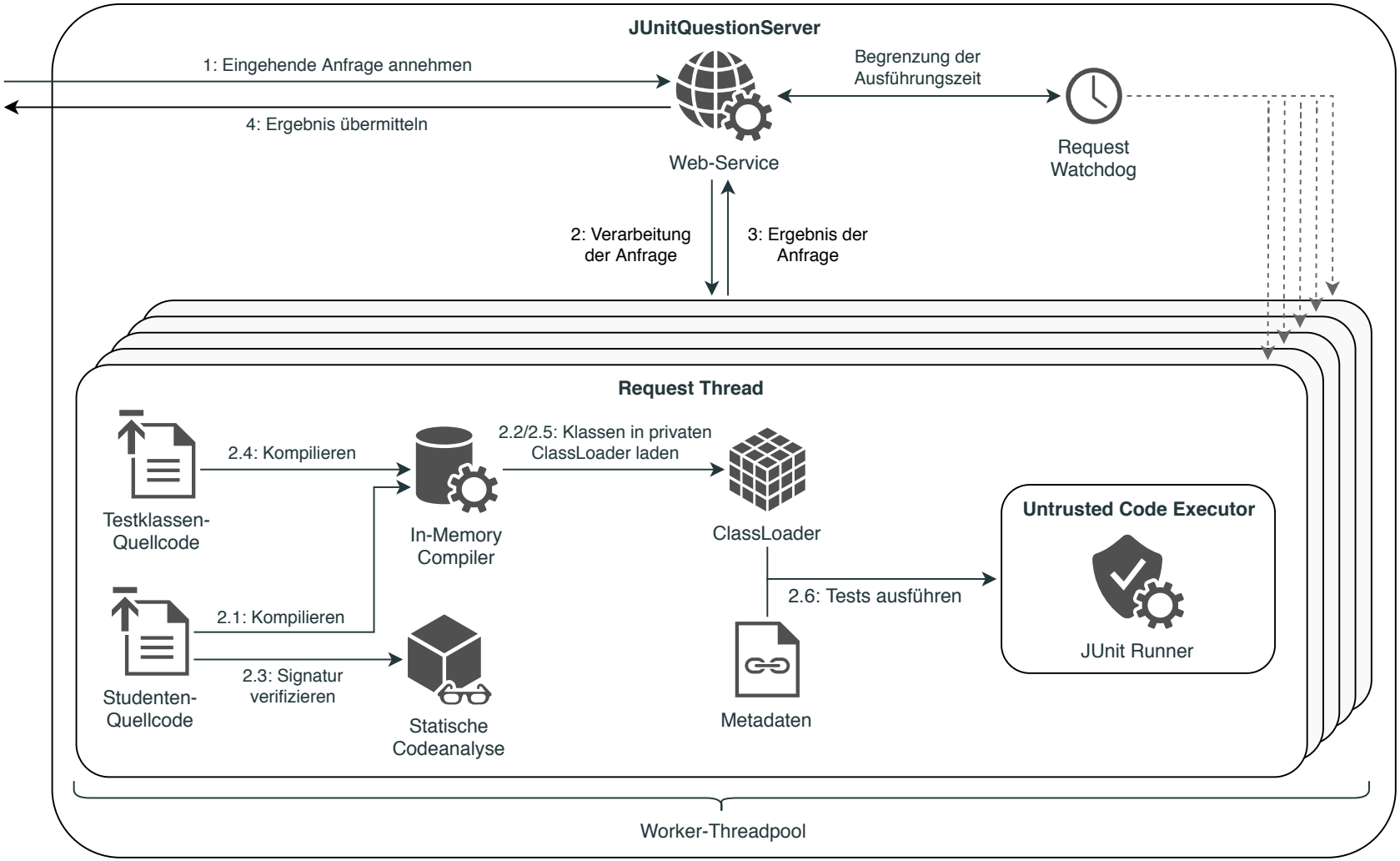

Abbildung 5: Architekturdetails des entwickelten Auswertungsservers

Bewertungsablauf (Normalfall)

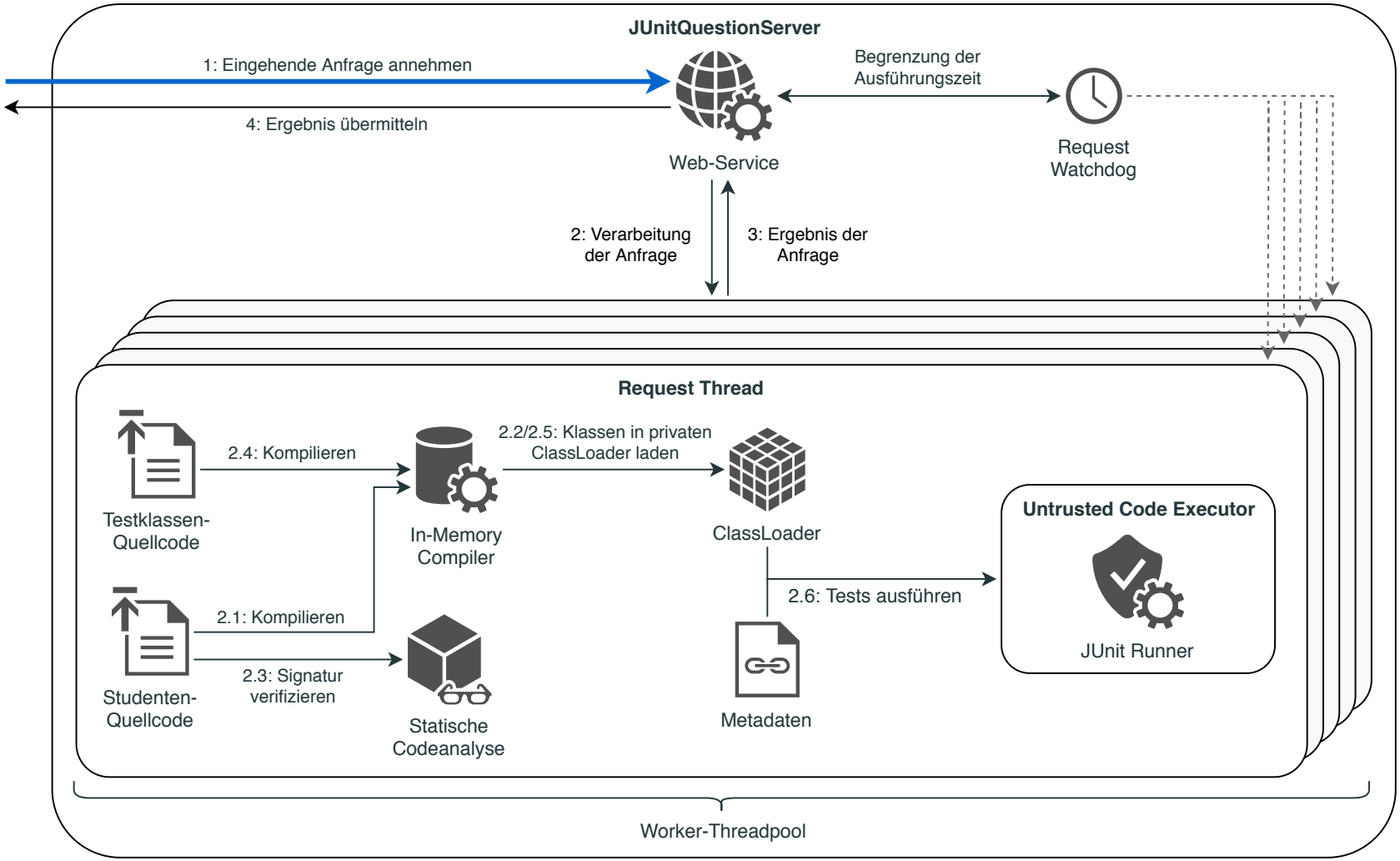

Abbildung 5: Architekturdetails des entwickelten Auswertungsservers

Bewertungsablauf (Normalfall)

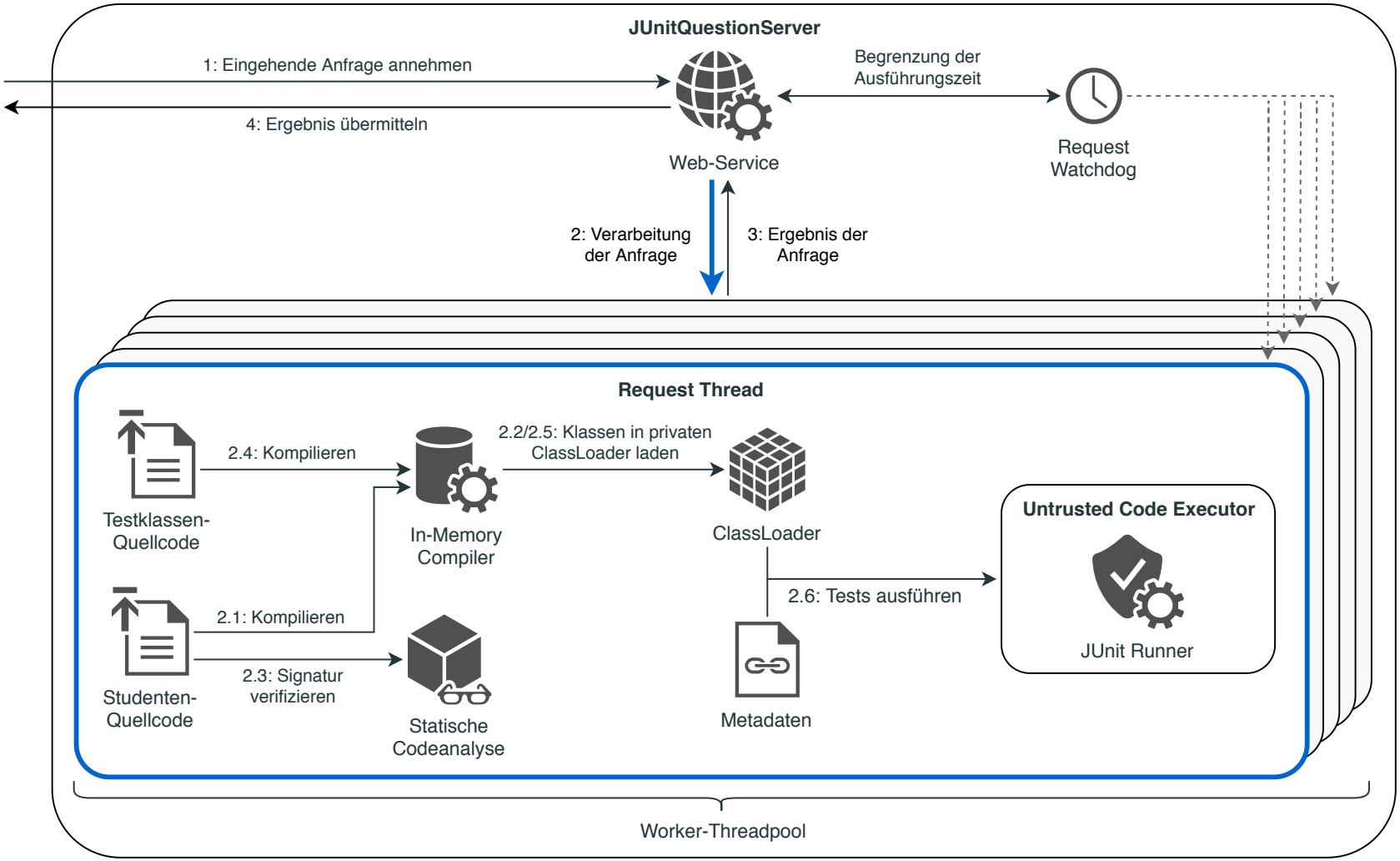

Abbildung 5: Architekturdetails des entwickelten Auswertungsservers

Bewertungsablauf (Normalfall)

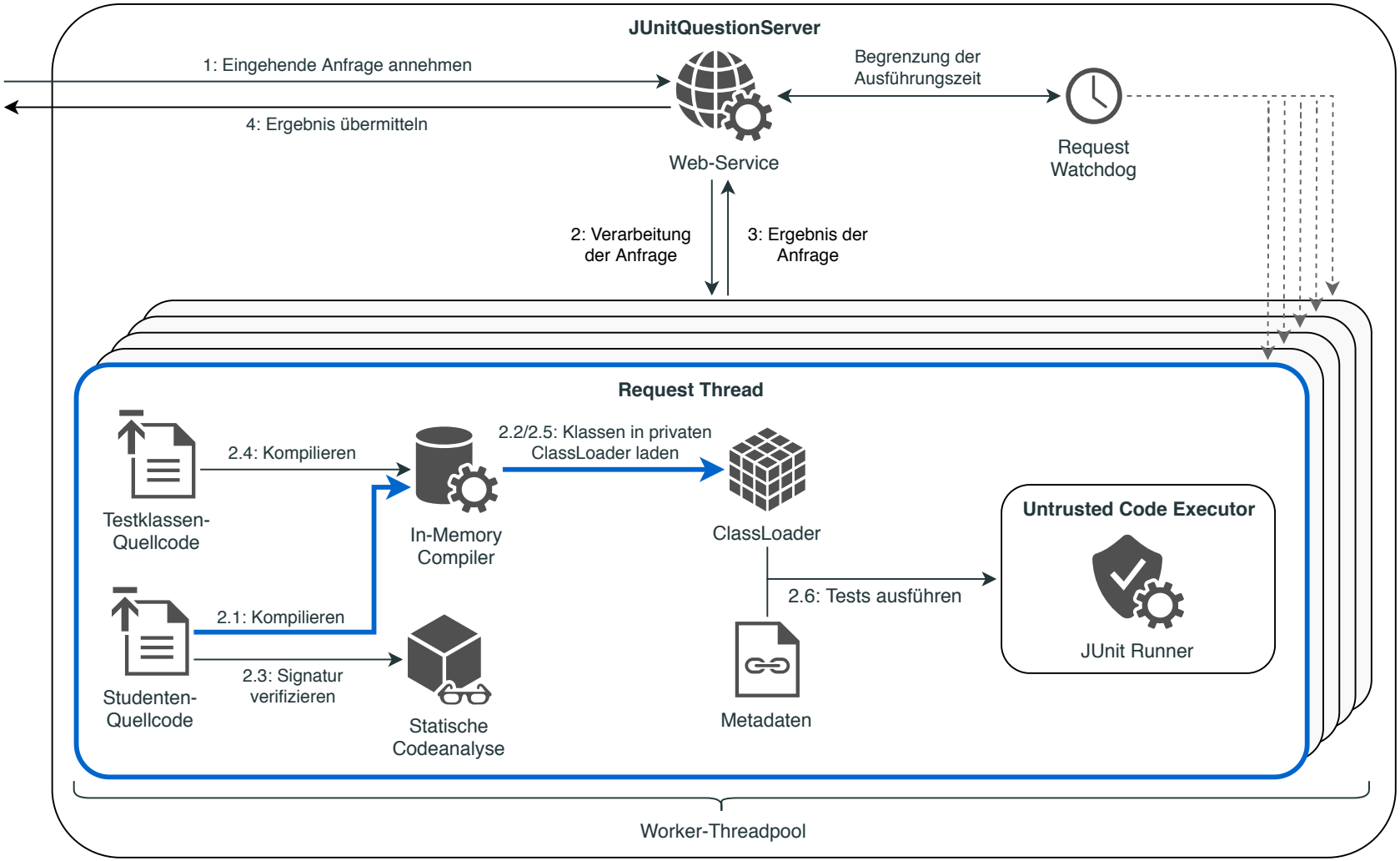

Abbildung 5: Architekturdetails des entwickelten Auswertungsservers

Bewertungsablauf (Normalfall)

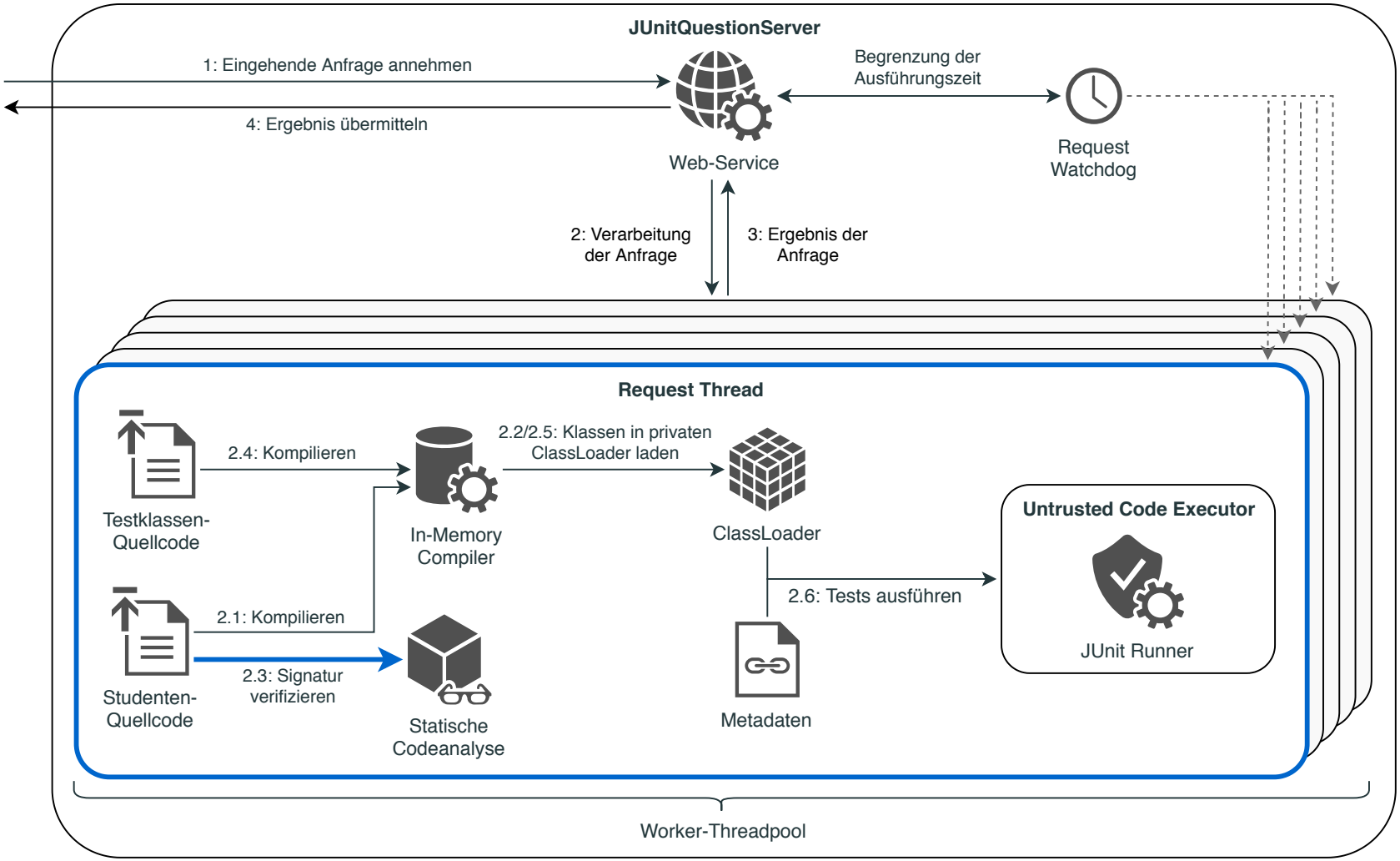

Abbildung 5: Architekturdetails des entwickelten Auswertungsservers

Bewertungsablauf (Normalfall)

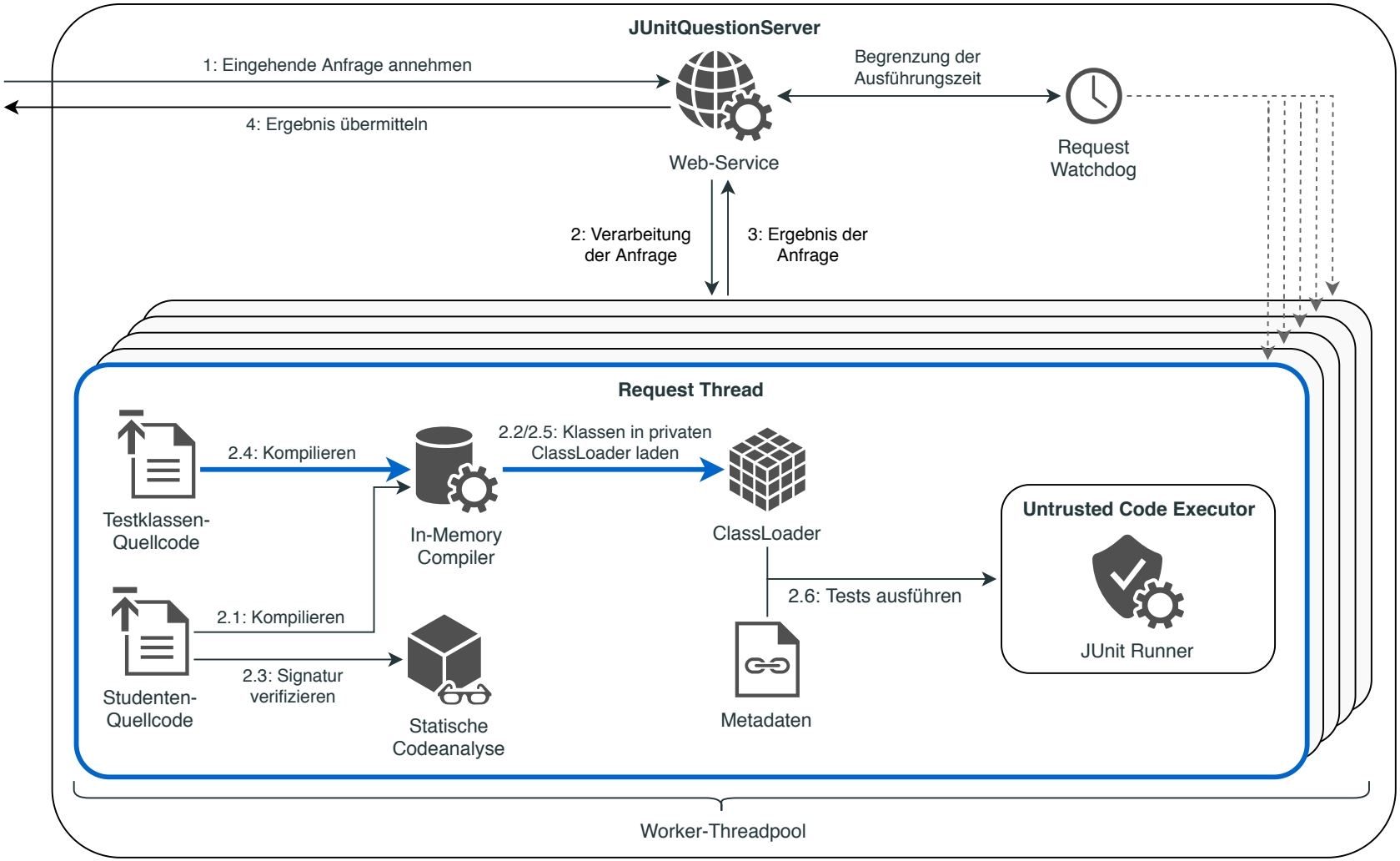

Abbildung 5: Architekturdetails des entwickelten Auswertungsservers

Bewertungsablauf (Normalfall)

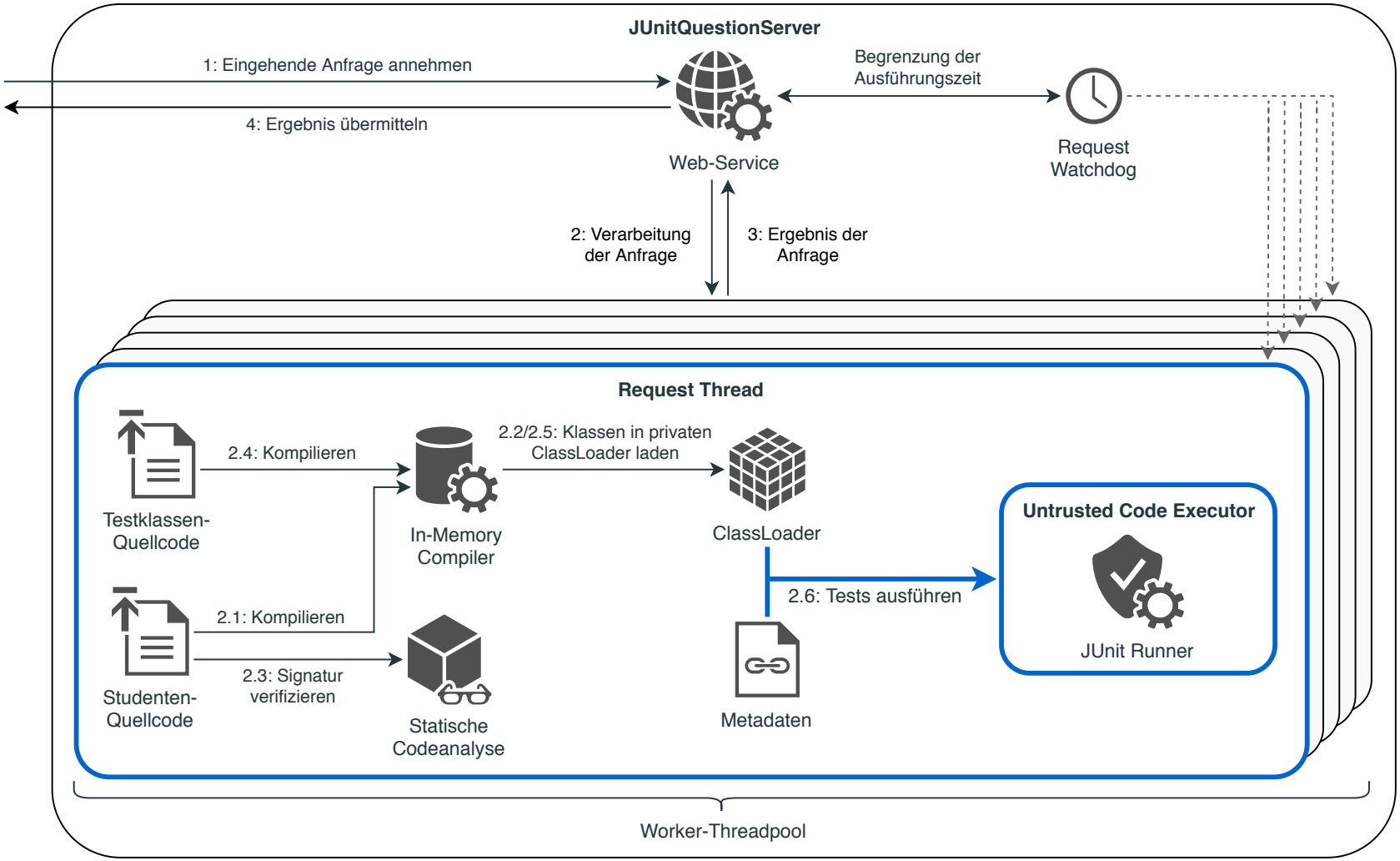

Abbildung 5: Architekturdetails des entwickelten Auswertungsservers

Bewertungsablauf (Normalfall)

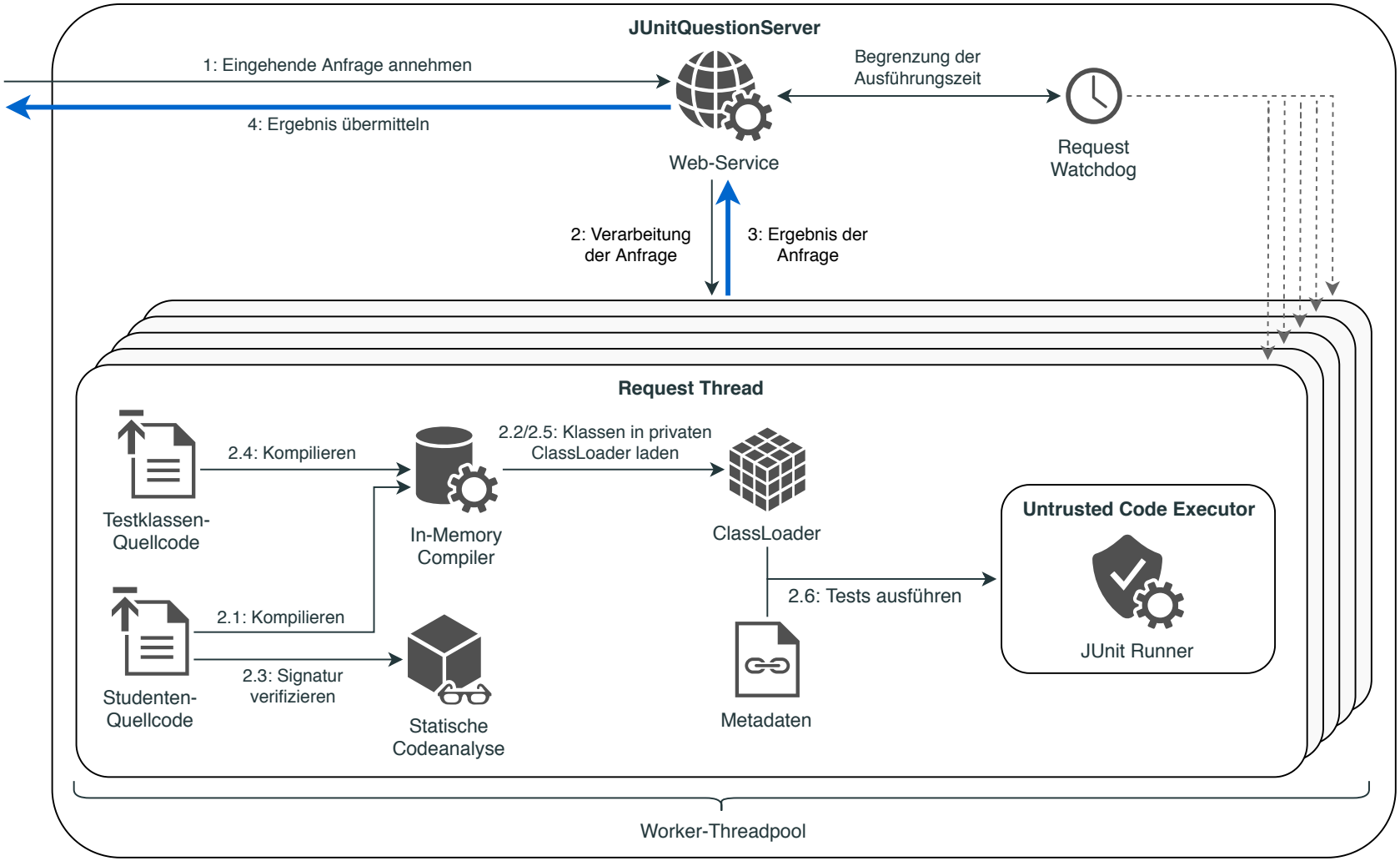

Abbildung 5: Architekturdetails des entwickelten Auswertungsservers

Bewertungsablauf

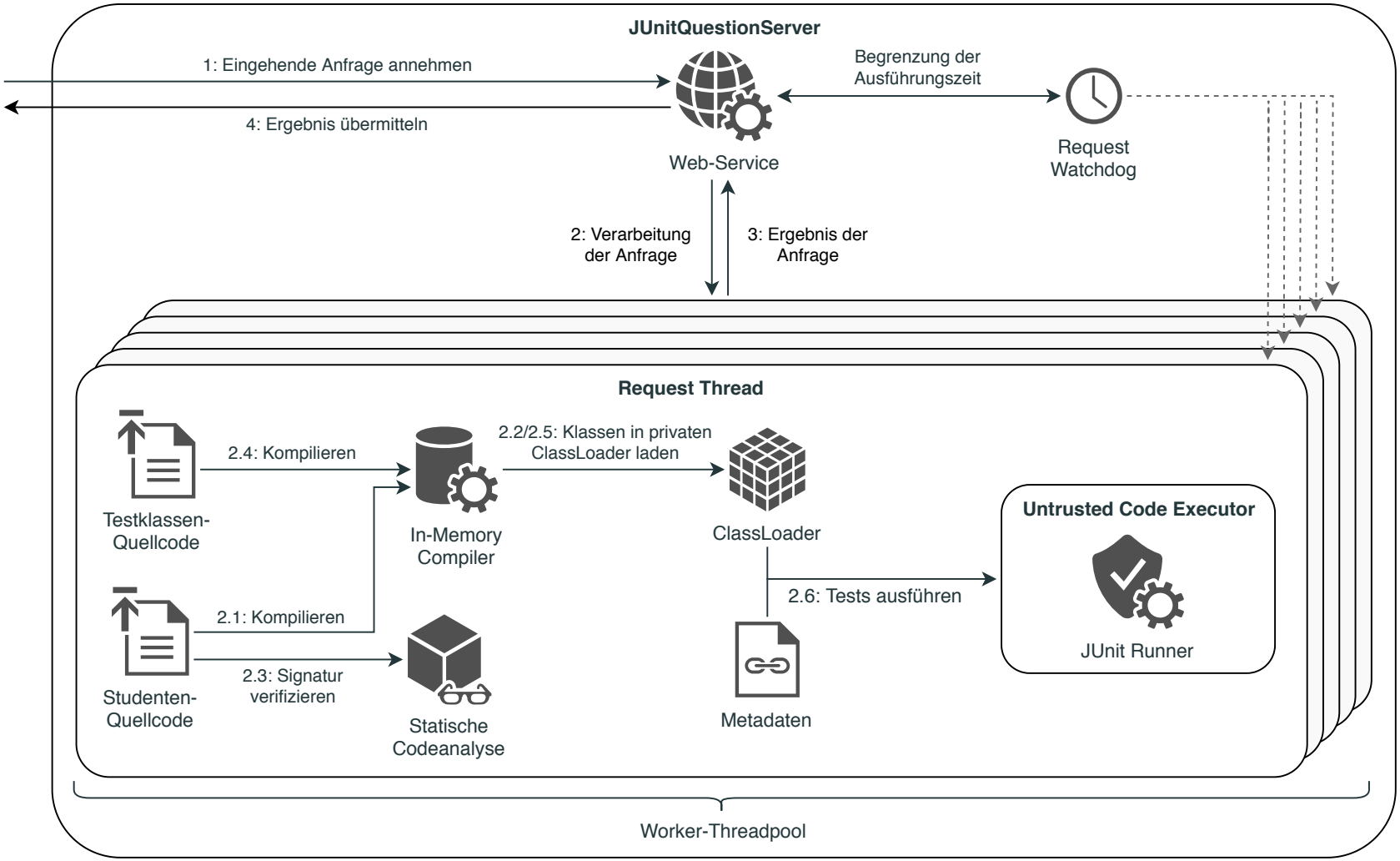

Abbildung 5: Architekturdetails des entwickelten Auswertungsservers

Bewertungsablauf (Fehlerfall / Abbruch)

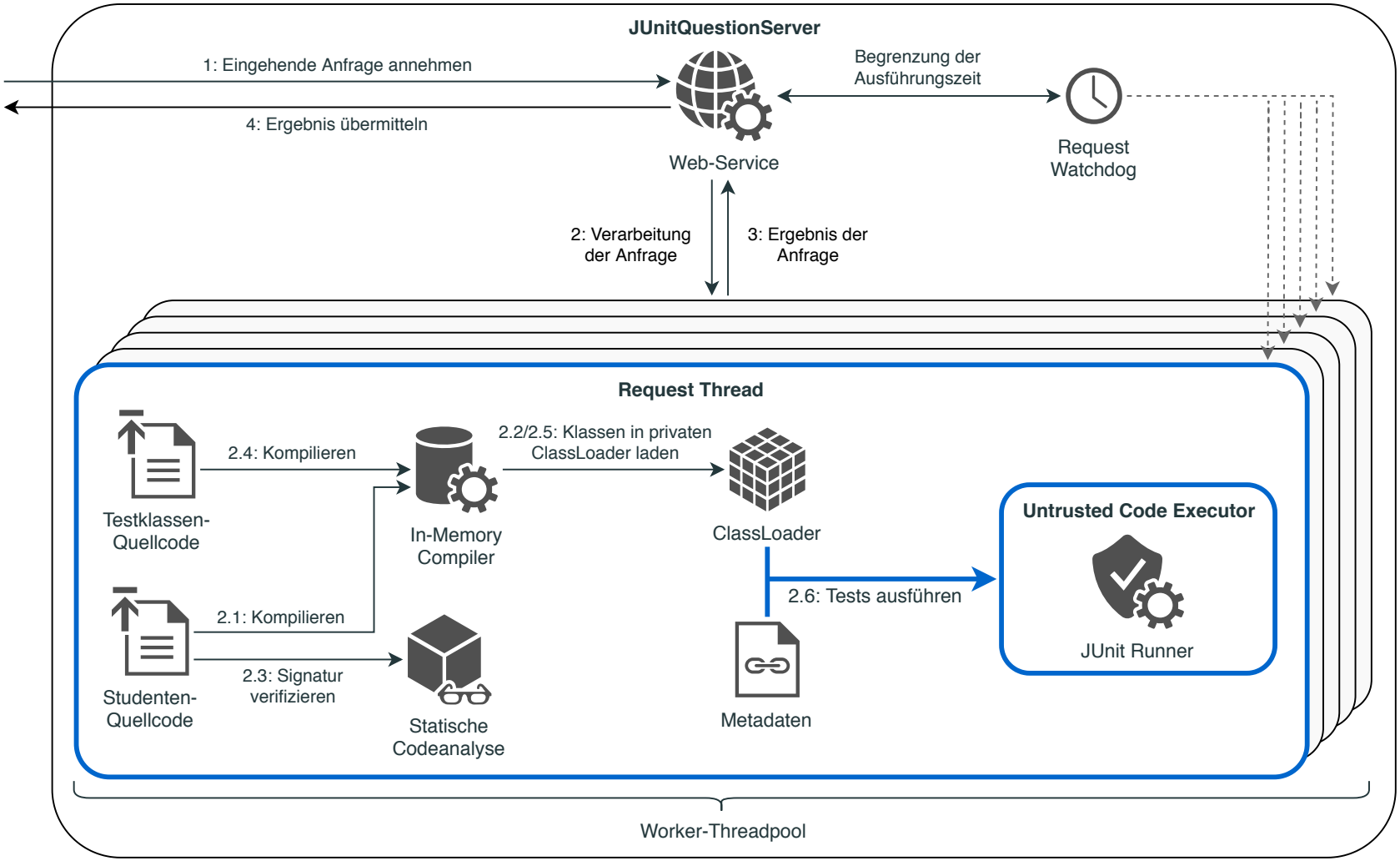

Abbildung 5: Architekturdetails des entwickelten Auswertungsservers

Bewertungsablauf (Fehlerfall / Abbruch)

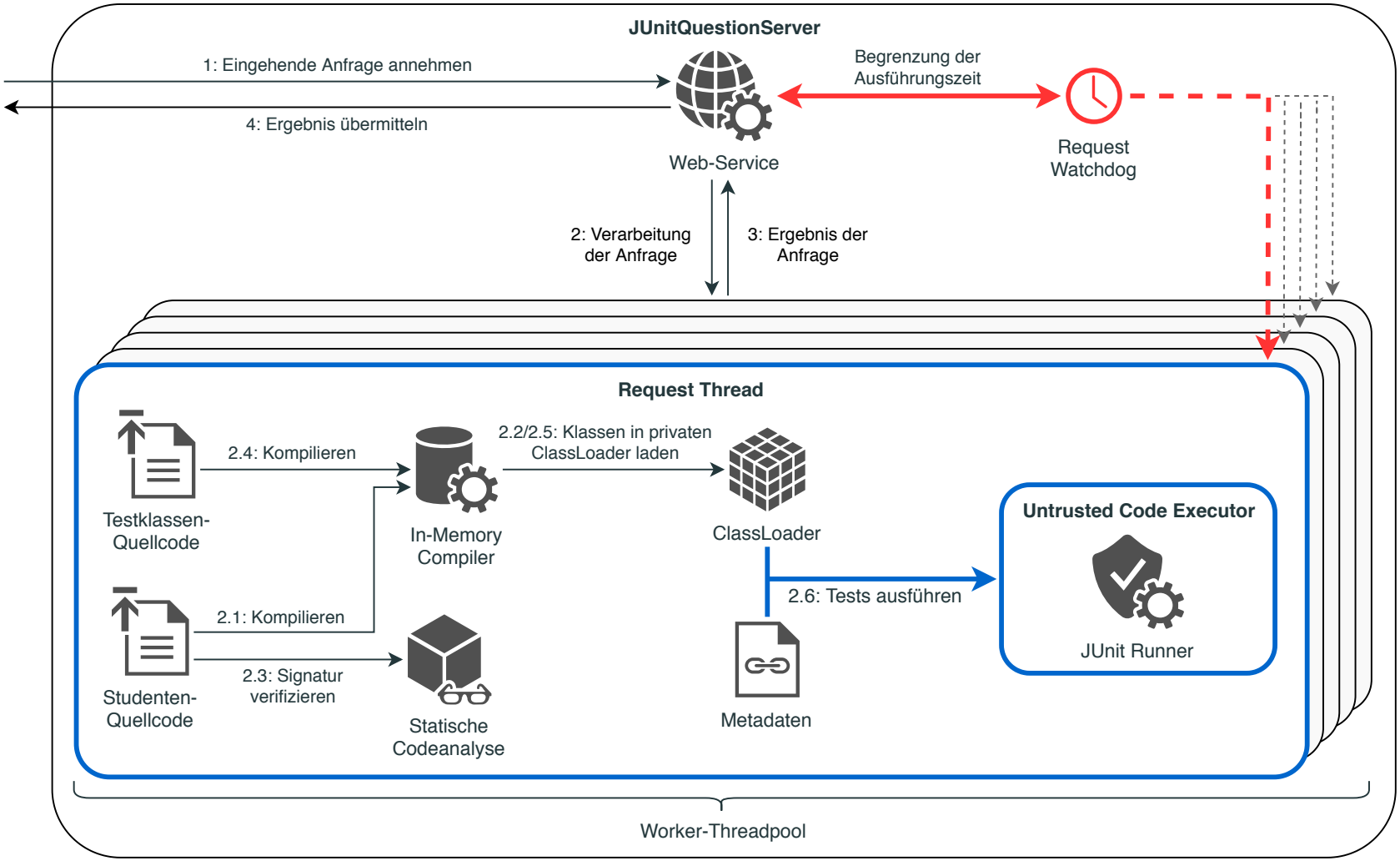

Abbildung 5: Architekturdetails des entwickelten Auswertungsservers

Bewertungsablauf (Fehlerfall / Abbruch)

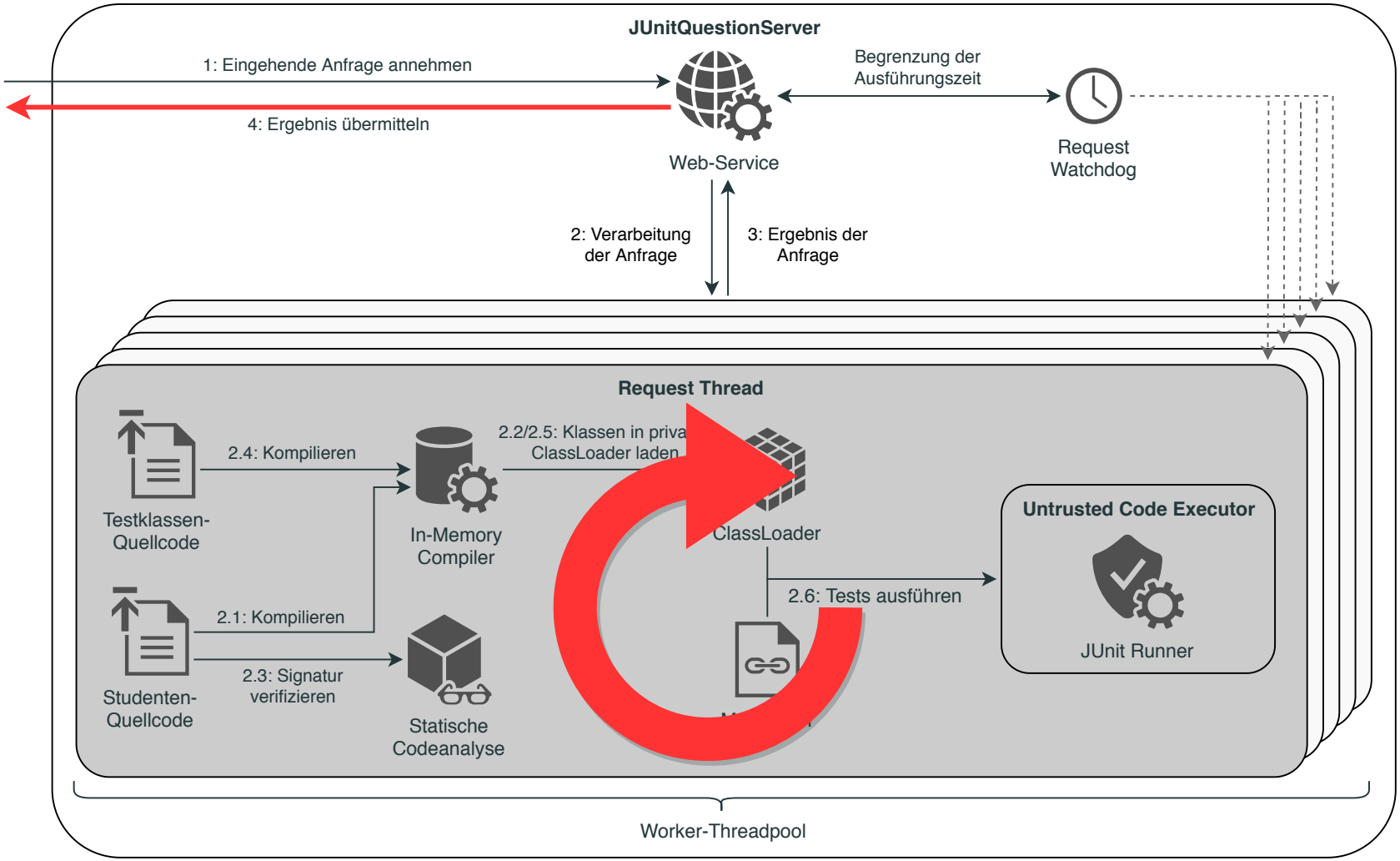

Abbildung 5: Architekturdetails des entwickelten Auswertungsservers

Bewertungsablauf

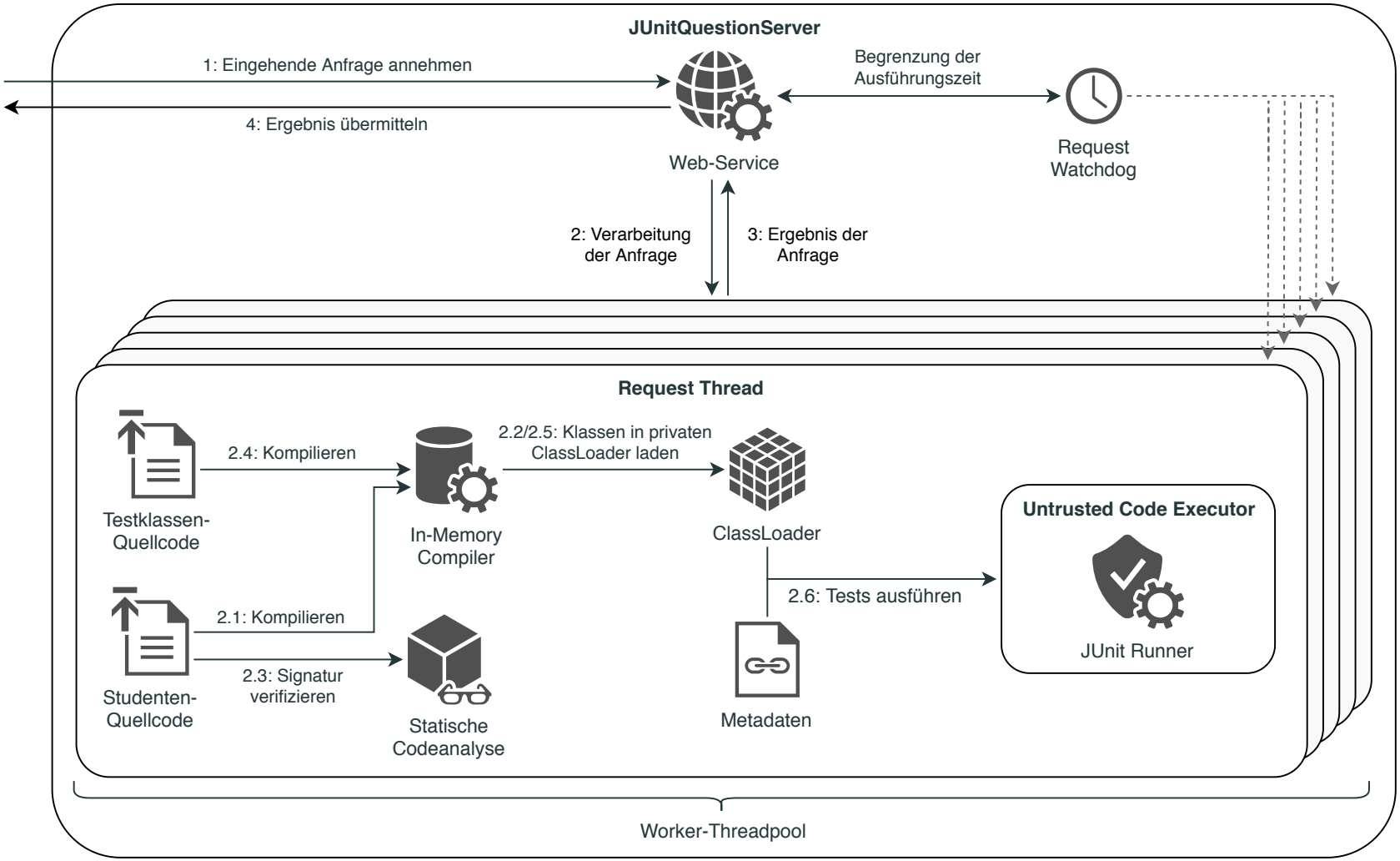

Abbildung 5: Architekturdetails des entwickelten Auswertungsservers

Einsatz an der HAW Hamburg

Einsatz an der HAW Hamburg

Aktueller Einsatz an der HAW Hamburg

- Semesterbegleitende Übungsaufgaben zur Vermittlung von Basiskonzepten der (objektorientierten) Programmierung
- Sowohl *Lese-* als auch *Schreibkompetenz*-Aufgaben
- Hohe Nutzungsquote unter den Studierenden trotz freiwilliger Bearbeitung außerhalb der Präsenzzeiten

Zukünftige Einsatzmöglichkeiten

- Nutzung im Rahmen von Prüfungsvorleistungen oder E-Klausuren
- Integration in interaktive Vorkurse auf der viaMINT-Plattform

Fazit

Fazit

- Entwickelter Moodle-Fragetyp erfolgreich seit mehreren Semestern an der HAW Hamburg im Einsatz
- Umsetzung von Java-Online-Aufgaben mit fortgeschrittenen Programmierkonzepten (Vererbung, Generizität, . . .) u. a. durch virtuelle Dateien und statische Typprüfungen
- Detailliertes und interaktives Feedback noch vor Abgabe der Aufgabe
- Deutliche Verringerung der Auswertungszeit bei gleichzeitiger Isolation der einzelnen JUnit-Runner durch Auslagerung in Webservice

Fragen?

Diskussion!

Literatur i

[Be16] Bertalan, G.; Rumler, M., Moodle Fragetyp: javaunittest (qtype_javaunittest), Version 2.03, Technischen Universität Berlin, März 2016, URL: https://moodle.org/plugins/view.php?plugin=qtype_javaunittest, Stand: 31. 05. 2019.

[Öz08] Özcan, S., Moodle Fragetyp: sojunit (qtype_sojunit), Sep. 2008, URL: https://moodle.org/mod/forum/discuss.php?d=102690, Stand: 31. 05. 2019.

[Lü17] Lückemeyer, G., Moodle Plugin: JUnit Exercise Corrector (assignsubmission_mojec), Version 1.0, Hochschule für Technik Stuttgart, Jan. 2017, URL: https://moodle.org/plugins/assignsubmission_mojec, Stand: 31. 05. 2019.

Literatur ii

[Ba13] Becker, S.; et al., Prototypische Integration automatisierter Programmbewertung in das LMS Moodle, in: 1. Workshop Autom. Bewertung von Programmieraufgaben. 2013.

[St14] Stöcker, A.; Becker, S.; Garmann, R.; Heine, F.; Kleiner, C.; Werner, P.; Grzanna, S.; Bott, O. J., Die Evaluation generischer Einbettung automatisierter Programmbewertung am Beispiel von Moodle und aSQLg, in: DeLFI. 2014.

[KSZ02] Krinke, J.; Störzer, M.; Zeller, A., Web-basierte Programmierpraktika mit Praktomat, Softwaretechnik-Trends 22/3, 2002.

[Ei03] Eichelberger, H.; Fischer, G.; Grupp, F.; von Gudenberg, J. W., Programmierausbildung Online, in: DeLFI. 2003.

Literatur iii

[OKP17] Oster, N.; Kamp, M.; Philippsen, M., AuDoscore: Automatic Grading of Java or Scala Homework, in: 3. Workshop Automatische Bewertung von Programmieraufgaben. 2017.

[Sc17] Schmolitzky, A., Zahlen, Beobachtungen und Fragen zur Programmierlehre, in: Tagungsband 15. Workshop "Software Engineering im Unterricht der Hochschulen". 2017.

[La18] Landenfeld, K.; Göbbels, M.; Hintze, A.; Priebe, J., A Customized Learning Environment and Individual Learning in Mathematical Preparation Courses, in: Distance Learning, E-Learning and Blended Learning in Mathematics Education: International Trends in Research and Development. Springer International Publishing, Cham, S. 93–111, 2018.

Literatur iv

[We87] Wegner, P., Dimensions of Object-based Language Design, in: Conference Proceedings. OOPSLA '87, ACM, Orlando, Florida, USA, S. 168–182, 1987.

[GS17] Goedicke, M.; Striewe, M., 10 Jahre automatische Bewertung von Programmieraufgaben mit JACK, in: INFORMATIK 2017. Gesellschaft für Informatik, Bonn, S. 279–283, 2017.

[Ga16] Garmann, R., Graja - Autobewerter für Java-Programme, Techn. Ber., 2016, S. 20.