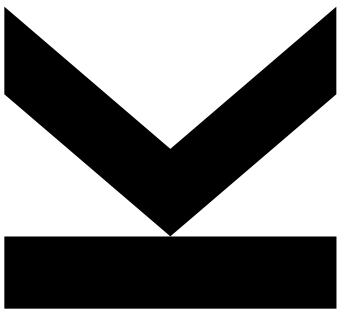
JOHANNES KEPLER
UNIVERSITÄT LINZ

ANFORDERUNGSANALYSE



258321 DKE Projekt

Gruppe 2

Teammitglieder:
k01607605, Aistleithner Andrea
k01256561, Dusanic Maja
k01356577, Teuchtmann Alexander
k01356229, Tomic Milos

JOHANNES KEPLER
UNIVERSITÄT LINZ
Altenberger Straße 69
4040 Linz, Österreich
www.jku.at
DVR 0093696

Inhaltsverzeichnis

1. Introduction 2
    1.1. Purpose 2
    1.2. Definitions 2
    1.3. References 2
2. System overview 3
3. Overall description 3
    3.1. Design constraints 3
    3.2. Product functions 4
    3.3. User characteristics 4
    3.4. Limitations, assumptions and dependencies 4
4. Specific requirements 5
    4.1. Functional requirements 5
        4.1.1. Functional Requirement 1 5
        4.1.2. Functional Requirement 2 5
        4.1.3. Functional Requirement 3 5
        4.1.4. Functional Requirement 4 5
        4.1.5. Functional Requirement 5 5
        4.1.6. Functional Requirement 6 5
        4.1.7. Functional Requirement 7 6
        4.1.8. Functional Requirement 8 6
        4.1.9. Functional Requirement 9 6
        4.1.10. Functional Requirement 10 6
        4.1.11. Funcitonal Requirement 11 6
        4.1.12. Functional Requirement 12 6
    4.2. Logical database requirements 6
    4.3. Software system attributes 6
        4.3.1. Reliability 7
        4.3.2. Availability 7
        4.3.3. Security 7
        4.3.4. Maintainability 7
        4.3.5. Portability 7

1. Introduction

Dieses Dokument soll einen Einblick in die Anforderungen an das Programm geben, welches in der Lehrveranstaltung (LVA) „DKE Praktikum“, an der Johannes Kepler Universität, behandelt wird. Es stellt die allgemeinen Funktionalitäten, Zielgruppen, sowie speziellere Anforderungen dar.

1.1. Purpose

Ziel dieses Praktikums ist es, eine Performance Evaluierung in Form eines Evaluierungsframeworks zu realisieren. Dieses soll auf das „Contextualized Business Rule Management“ und auf die „Rule Module Inheritance with Modification Restrictions“ angewendet werden. Durch Testläufe sollen dann aussagekräftige Performanceanalysen durchgeführt werden, welche wiederum Daten generieren, die für eine Performancesteigerung weiterverwendet werden können. Diese Daten sollen für weitere Zwecke gespeichert werden.
Zusätzlich sollen Datengeneratoren implementiert werden, welche Datensätze erzeugen, und anschließend vom Evaluierungsframework für die Performanceanalyse verwendet werden. Schlussendlich soll das Programm über die Konsole bedient werden können.

Nicht Ziel ist es, eine graphische Benutzeroberfläche für die Performance Evaluierung zu erstellen.

1.2. Definitions

Abkürzung	**Definition**
CBR	Contextualized Business Rule Management
Vadalog	auf Datalog basierende Wissensgenerierung
IEEE	Institute of Electrical and Electronics Engineers

1.3. References

[1] IEEE Software Engineering Standards Committee, “ISO/IEC/IEEE 29148:2011(E), IEEE Systems and software engineering - Life cycle processes - Requirementsengineering”, January 12, 2011.

2. System overview

Das Software System des Programmes besteht aus folgenden Elementen:

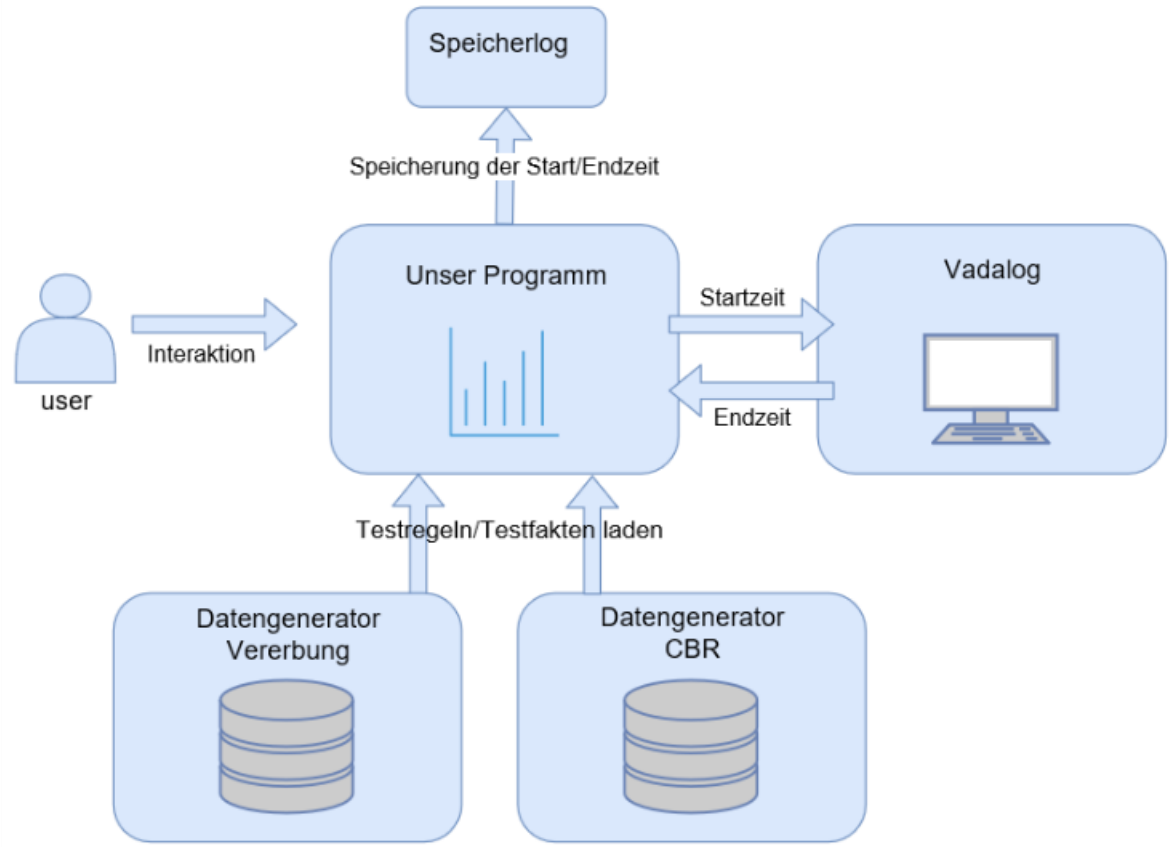


Abbildung 1: Software System Overview

3. Overall description

3.1. Design constraints

- In Bezug auf die Programmiersprache – keine Eingrenzung
- In Bezug auf das Betriebssystems – muss auf Fedora 28 (Linux) lauffähig sein
- In Bezug auf die Benutzung von externen Libraries und/oder Frameworks – keine Eingrenzung

3.2. Product functions

Die allgemeine Funktionalität des Programmes besteht darin, Testdaten für CBR und Inheritance zu generieren um damit verschiedene Testfälle durchführen zu können und somit die Performance der Durchführung messen und evaluieren zu können.

3.3. User characteristics

Insgesamt gibt es drei Arten von Benutzern: „Softwareadministratoren“, „Zuständige für die Geschäftsregeln“ und der spätere Benutzer der Software.

Der Softwareadministrator ist für die Entwicklung und Implementierung der Software und der Schnittstellen verantwortlich. Er ist für den reibungslosen Betrieb des Systems zuständig, sowie für die Sortierung der Geschäftsregeln. Ebenso ist er für die Ausführung der Performanceanalyse, sowie für die Speicherung der Daten, welche auf den Ergebnissen der Analyse basieren, zuständig.

Personen, welche die Geschäftsregeln definieren und laufend, entsprechend der Organisation, aktualisieren, sind ebenso involviert. Diese Regeln werden vom Systemadministrator oder von der jeweilig zuständigen Person in das System eingepflegt.

Der Benutzer der Software wird lediglich mit den für die jeweilige Benutzergruppe vorhergesehenen Informationen konfrontiert. Diese Informationen werden auf Basis der Geschäftsregeln und der Software gefiltert.

3.4. Limitations, assumptions and dependencies

Eine Einschränkung bei der Entwicklung der Performanceanalyse ist, dass jenes grundsätzlich betriebssystemunabhängig arbeiten muss, aber in jedem Fall unter Fedora 28 (Linux) lauffähig sein muss.

4. Specific requirements

4.1. Functional requirements

Im Folgenden werden die Funktionalitäten, welche das Programm haben soll, behandelt.

4.1.1. Functional Requirement 1

Datengenerierung von CBR Datengenerator

- Es sollen Parameter, Parameter Werte, Business Cases, Kontext Modelle und Kontexte für die Testfälle generiert werden.
- need to have

4.1.2. Functional Requirement 2

Datengenerierung von Inheritance Datengenerator

- Es sollen Parameter, Parameter Werte, Business Cases, Kontext Modelle und Kontexte für die Testfälle generiert werden.
- need to have

4.1.3. Functional Requirement 3

Durchführung von CBR Performancetests

- need to have

4.1.4. Functional Requirement 4

Durchführung von Inheritance Performancetests mit Single-Inheritance ohne Modifikation

- need to have

4.1.5. Functional Requirement 5

Durchführung von Inheritance Performancetests mit Multi-Inheritance ohne Modifikation

- need to have

4.1.6. Functional Requirement 6

Durchführung von Inheritance Performancetests mit Single-Inheritance mit Modifikation

- need to have

4.1.7. Functional Requirement 7

Durchführung von Inheritance Performancetests mit Multi-Inheritance mit Modifikation

- need to have

4.1.8. Functional Requirement 8

Durchführung von Inheritance Performancetests mit Single-Inheritance mit Restricitons

- need to have

4.1.9. Functional Requirement 9

Durchführung von Inheritance Performancetests mit Single-Inheritance ohne Restricitons

- need to have

4.1.10. Functional Requirement 10

Durchführung von Inheritance Performancetests mit Multi-Inheritance mit Restricitons

- need to have

4.1.11. Funcitonal Requirement 11

Durchführung von Inheritance Performancetests mit Multi-Inheritance ohne Restricitons

- need to have

4.1.12. Functional Requirement 12

Speicherung der Evaluierungswerte auf einem programmexterenen Ort

- need to have

4.2. Logical database requirements

Es existieren keine spezielle Anforderungen in diesem Bereich.

4.3. Software system attributes

Im Folgenden werden die Attribute Reliability (Zuverlässigkeit), Availability (Verfügbarkeit), Security (Sicherheit), Maintainability (Wartbarkeit) und Portability (Portierung) behandelt.

4.3.1. Reliability

Für die Zuverlässigkeit des Systems werden folgende Faktoren benötigt:

- Daten sind korrekt geschrieben (keine Syntax-Fehler)
- Die Semantik der Daten entspricht dem geplanten Ergebnis
- Die Fakten sind korrekt geschrieben (keine Syntax-Fehler)
- Die Semantik der Fakten entspricht dem geplanten Ergebnis
- Die Schnittstelle zum Vadalog ist fehlerfrei

4.3.2. Availability

Für die Verfügbarkeit des Systems sind folgende Faktoren wichtig:

- Die Benutzung von Checkpoints

4.3.3. Security

Um die Sicherheit zu gewährleisten werden folgende Punkte benötigt:

- Die Datensicherheit gewährleisten
- Die Datenintegrität bei kritischen Variablen gewährleisten
- Eine beschränkte Kommunikation zwischen einigen Teilen des Programms

4.3.4. Maintainability

Die Wartbarkeit des Systems wird durch folgende Punkte berücksichtigt:

- Die Benutzung eines simplen Namensschemas
- Die Dokumentation von wichtigen Entscheidungen bezüglich des Codes

4.3.5. Portability

Für die Erleichterung der Portierung des Systems auf andere Maschinen und/oder Betriebssysteme werden folgende Punkte berücksichtigt:

- Die Benutzung einer portablen Programmiersprache (z.B. Java)
- Die Benutzung eines portablen Compilers

Abbildungsverzeichnis

Abbildung 1: Software System Overview 3