

A dark blue vertical bar on the left side of the page. A blue arrow points to the right from the bar, containing the date.

28.10.2018

# UniRisk

Architekturdokument

Several thin, curved, light blue lines that sweep upwards from the bottom left corner of the page.

Oliver Bosin

VERSION: 1.0

STATUS: IM BEARBEITUNG

## Historie

Version	Status	Datum	Autor(en)	Erläuterung
1.0	In Bearbeitung	28.10.18	Oliver Bosin	Erster Entwurf

## Inhaltsverzeichnis

Historie .....	1
1    Einleitung .....	3
1.1    Zweck des Dokuments.....	3
1.2    Architekturmuster .....	3
2    Technische Architektur .....	3
2.1    TI-Architektur – Technische Infrastruktur.....	3
2.2    Datenmodell.....	3
3    Anwendungs-Architektur.....	4
3.1    Komponentenmodell.....	4
4    IT-Konzept.....	5
4.1    Detaillierte Schnittstellenbeschreibung.....	5
4.2    Dynamische Beschreibung exemplarischer Anwendungsfälle .....	5
4.3    Detaillierte Beschreibung des internen Ablaufs.....	5

# 1 Einleitung

## 1.1 Zweck des Dokuments

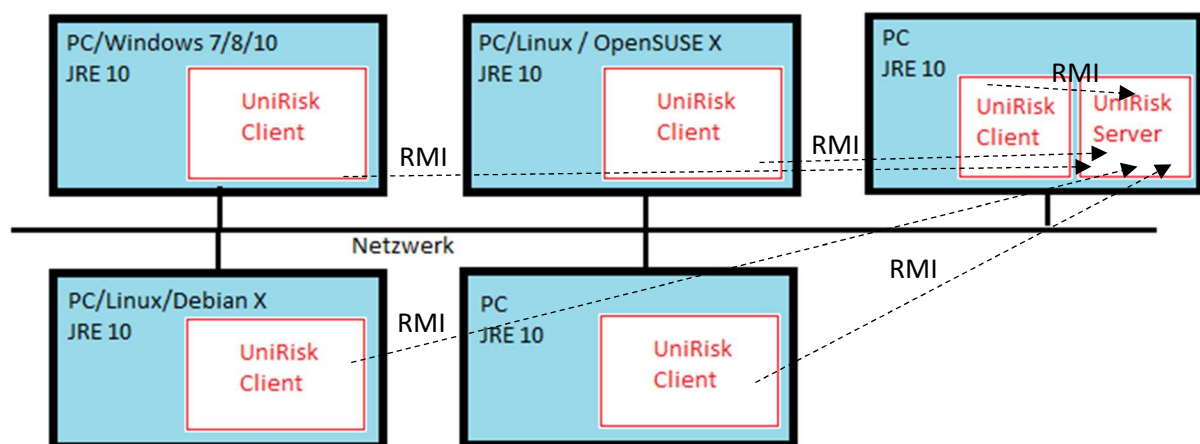
Dieses Dokument beschreibt die Struktur („Komponentisierung“) des Softwaresystems „UniRisk“, aber auch Informationen über die Kommunikation zwischen Komponenten, sowie deren Abbildung auf Software-Ressourcen ist Bestandteil. Der Vorteil der Softwarearchitekturbeschreibung besteht darin, dass diese über den gesamten Lebenszyklus des Software-Systems genutzt werden kann.

## 1.2 Architekturmuster

Da das Projekt „UniRisk“ sowohl ein interaktives System als auch ein verteiltes System darstellt, wird eine Kombination aus dem Model-View-Controller-Muster und dem Client-Server-Muster umgesetzt.

# 2 Technische Architektur

## 2.1 TI-Architektur – Technische Infrastruktur



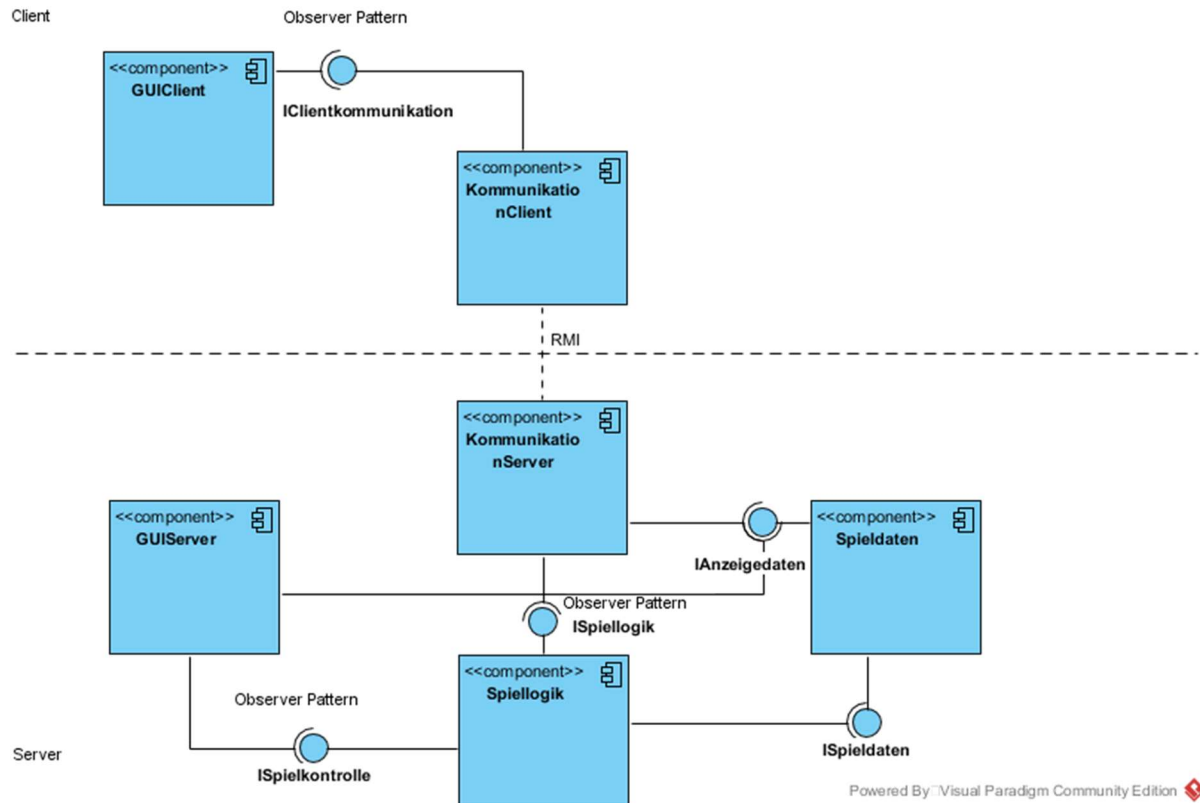
In der Infrastruktur wird es einen PC geben auf dem die „UniRisk“-Serveranwendung läuft. Mit diesem Host können sich dann ein bis fünf Clients verbinden. Der Host selbst kann einer dieser Clients sein, muss er aber nicht zwingend.

## 2.2 Datenmodell

Es sind keine Änderungen zum fachlichen Datenmodell vorhanden.

## 3 Anwendungs-Architektur

### 3.1 Komponentenmodell



Die „**GUIServer**“ Komponente bietet dem Host eine Bedienoberfläche. Dazu verwendet diese Komponente die Schnittstelle „**ISpielkontrolle**“ und „**IAnzeigedaten**“ der Komponenten „**Spiellogik**“ und „**Spieldaten**“. Über die „**GUIServer**“ Komponente werden die Anwendungsfälle „**Spiel anlegen**“, „**Spiel laden**“, „**Spiel starten**“ und „**Spiel beenden**“ gestartet.

Die „**GUIClient**“ Komponente bietet den Clients zu Anfang eine Oberfläche zur Anmeldung, sowie die Lobby und nach Spielbeginn eine Spielfläche. Die Komponente verwendet dazu die Schnittstelle „**IClientkommunikation**“ um Daten an die „**KommunikationClient**“ Komponente zu überreichen bzw. Daten von dieser entgegen zu nehmen. In der „**GUIClient**“ Komponente kann man die Anwendungsfälle „**Am Server anmelden**“, „**Bereit melden**“ in der Lobby-Oberfläche starten und die Anwendungsfälle „**Ersties verteilen**“, „**Spielzug durchführen**“ (+alle Phasen) sowie „**Angriff abwehren**“ in der Spielfläche starten.

Die „**KommunikationClient**“ Komponente beinhaltet alle Klassen und Methoden, die benötigt werden, um Daten des Clients entgegen zu nehmen und an den Server weiterzureichen. Genauso nimmt sie die Daten vom Server entgegen und gibt diese weiter an den Client. Dies wird durch ihre Schnittstelle „**IClientkommunikation**“ zur Komponente „**GUIClient**“ und RMI mit „**KommunikationServer**“ ermöglicht.

Die „**KommunikationServer**“ Komponente beinhaltet alle Klassen und Methoden, die benötigt werden, um Daten von der Komponente Spiellogik sowie der Komponente „**Spieldaten**“ entgegenzunehmen und an den Client weiter zu reichen. Genauso nimmt sie die Daten vom Client entgegen und reicht diese an den Server, zur weiteren Verarbeitung. Dies wird durch ihre Schnittstelle „**ISpiellogik**“ zur Komponente „**Spiellogik**“, bzw. ihre Schnittstelle „**IAnzeigedaten**“ zur Komponente

„**Spieldaten**“ und RMI mit „**KommunikationClient**“ ermöglicht. Insgesamt betrachtet ist die Aufgabe der beiden Komponenten „**KommunikationClient**“ und „**KommunikationServer**“, die Sicherstellung und der Erhalt der Kommunikationsfähigkeit zwischen der Server- und der Clientseite.

Die „**Spiellogik**“ Komponente beinhaltet alle Klassen und Methoden, die benötigt werden, um Daten von der Komponente „**Spieldaten**“ entgegenzunehmen und an die Komponenten „**GUISever**“ und „**KommunikationServer**“ weiter zu reichen. Dies wird durch die Schnittstellen „**ISpiellogik**“ zur Komponente „**KommunikationServer**“, „**ISpielkontrolle**“ zur Komponente „**GUISever**“ und „**ISpieldaten**“ zur Komponente „**Spieldaten**“ ermöglicht. Insgesamt betrachtet ist die Aufgabe der Komponente, die Sicherstellung des regelkonformen Spielablaufes.

Die „**Spieldaten**“ Komponente beinhaltet alle Klassen und Methoden, die benötigt werden, um Daten im Spiel zu verwalten und zu verteilen. Dies wird durch ihre Schnittstelle „**IAnzeigedaten**“ zu den Komponente „**GUISever**“ und Kommunikationsserver sowie „**ISpieldaten**“ zur Komponente „**Spiellogik**“ ermöglicht. Insgesamt betrachtet ist die Aufgabe der Komponente „**Spieldaten**“ die Sicherstellung der korrekten Speicherung sowie Verteilung von im Spiel entstehender Daten.

## 4 IT-Konzept

### 4.1 Detaillierte Schnittstellenbeschreibung

Die detaillierte Schnittstellenbeschreibung wird in den Java-Dateien mit JavaDoc umgesetzt.

### 4.2 Dynamische Beschreibung exemplarischer Anwendungsfälle

### 4.3 Detaillierte Beschreibung des internen Ablaufs