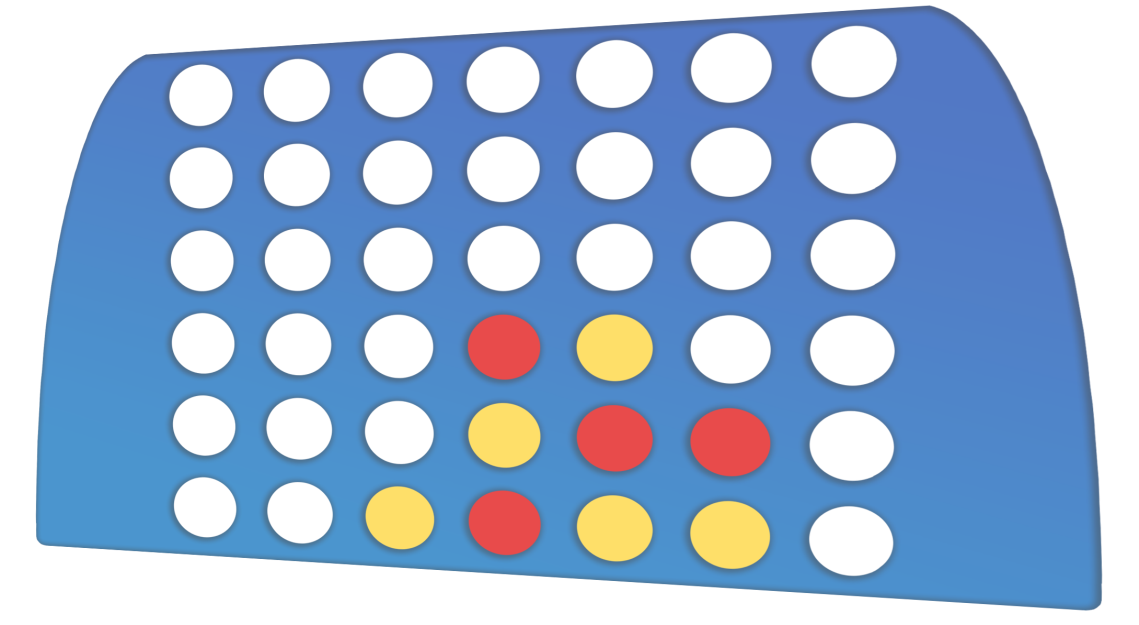
# Entwicklungsdokumentation

Duale Hochschule Baden-Württemberg Mannheim

- Vier gewinnt -

Bearbeitungszeitraum xx. September 2015 – xx. April 2016

****Verfasser: Marcel Cornesse

Dominik Efinger

Irena Männle

Luis Oberste

Shakira Steger

Kurs: WWI 13 SCA

Vorlesung: WI-Projekt

Dozent: Prof. Dr. Matthias Lauterbach

Inhalt

[Entwicklungsdokumentation 0](#_Toc431220323)

[1 Spezifikation 2](#_Toc431220324)

[1.1 Was leistet sie? 2](#_Toc431220325)

[1.1.1 UML um use case 2](#_Toc431220326)

[1.2 Sicht des Anwenders und Auftraggebers 2](#_Toc431220327)

[1.2.1 Mit Anforderungsbeschreibung 2](#_Toc431220328)

[2 Design 3](#_Toc431220329)

[2.1 Prozess der Softwareentwicklung 3](#_Toc431220330)

[2.1.1 Vorgehensmodell (agiles/scrum) 3](#_Toc431220331)

[2.1.2 Meilensteine und verantwortlichkeiten 3](#_Toc431220332)

[2.2 Aufbau, Entwurfsentscheidungen 5](#_Toc431220333)

[2.2.1 Files & folders 5](#_Toc431220334)

[2.2.2 Schichtenarchitektur 5](#_Toc431220335)

[2.2.3 aktivitätendiagramm 5](#_Toc431220336)

[2.3 Konzept 5](#_Toc431220337)

[2.3.1 Wie wird was umgesetzT? 5](#_Toc431220338)

[2.3.2 Gui -> swing 5](#_Toc431220339)

[2.3.3 DB > hsqldb 5](#_Toc431220340)

[2.4 Struktur 5](#_Toc431220341)

[2.4.1 Entitäten (Komponenten, Module, Pakete, Klassen) 5](#_Toc431220342)

[2.4.2 Deren Beziehungen und Abhängigkeiten 5](#_Toc431220343)

[2.4.3 Datenbank (ER-Diagramm) 5](#_Toc431220344)

[2.4.4 Klassen beschreiben 5](#_Toc431220345)

[3 Installationsanleitung 5](#_Toc431220346)

[4 Bedienungsanleitung 5](#_Toc431220347)

[4.1 Illustrierungen und Beispiele 5](#_Toc431220348)

Abschnitt: Entwicklung

# 1 Spezifikation

## Was leistet sie?

//

## UML um use case

## Sicht des Anwenders und Auftraggebers

## Mit Anforderungsbeschreibung

Nach Angaben des Dozenten, galt es folgende Anforderungen innerhalb des Bearbeitungszeitraumes eigenverantwortlich umzusetzen.

Um einen ersten Überblick des Projektumfangs zu erhalten ist im Folgenden die Kurzfassung der vorgegebenen Aufgabenbeschreibung aufgelistet:

* Bauen sie einen Software-Agenten, der autonom das Spiel „4 gewinnt“ gegen einen anderen Spieler spielen kann.
* Ihr Agent kommuniziert mit einem Server-Programm, mit dem auch der Gegenspieler, d. h. der Agent des anderen Teams, verbunden ist.
* Den Bau des Agenten führen sie - im Rahmen der vorgegebenen Anforderungen - selbstständig im Team durch. Das bedeutet insbesondere:
  + Sie managen ihr Projekt eigenverantwortlich
  + Sie gestalten eine Lösung entsprechend ihren Vorstellungen.
  + Sie wenden eine Methodik ihrer Wahl an.
  + Implementierungsdetails liegen weitgehend in ihrem Ermessen.
* Der Rahmen der vorgegebenen Anforderungen wird maßgeblich durch die wichtigsten zu liefernden Ergebnisse bestimmt (genauer siehe nächste Seiten):  
  + Agent (Prototyp, Beta und Release)
  + Dokumentation
  + Turnierteilnahme

Ebenfalls wurden die für das Projekt zugelassenen Hilfsmittel im Voraus durch den Dozenten festgelegt und setzten sich aus den folgenden sechs Quellen zusammen:

* Java-Tutorial
* Java-Bücher
* Java-Api-Doc
* XML-Spezifikation
* HSQLDB-User Guide
* bereitgestellte Bausteine

**Funktionale Anforderungen**

Der entsprechende Lieferumfang umfasst jeweils:

* Prototyp:
* Import muss entsprechend Importanleitung für Dozenten ohne Compilerfehler möglich sein.
* Ein auf Studentenrechner lauffähiger und vorzeigbarer Oberflächenprototypist gefragt. Die eine oder andere Funktionalität oder
* Optional: ein automatischer Demonstrationsablauf

* Beta
* Wie Prototyp zzgl.: mindestens 5 Züge über Server spielen können. Beide Schnittstellen müssen funktionieren.
* Zufallszug des eigenen Agents genügt; Siegmustererkennung oder Spalte-ist-voll-Erkennung nicht notwendig.
* Release
* auslieferungsfähiges Bundle (zusammen mit der Dokumentation und dem Quellcode als Menge von Dateien auf CD/DVD)
* mit dessen Hilfe ein Anwender auf der Basis der Dokumentation den Agenten installieren, konfigurieren und bedienen kann und ein Entwickler das Programm warten und erweitern kann (Eclipse-Import muss funktionieren).

Entsprechend zu den verschiedenen Themen werden jeweils eigene Anforderungen gestellt, die bis zum jeweiligen Termin fertig gestellt sein sollen.

# Design

* 1. Prozess der Softwareentwicklung
     1. Vorgehensmodell (agiles/scrum)

* + 1. Meilensteine

Folgende Abbildung zeigt eine Zeitleiste inklusive der allgemeinen Terminplanung und den festgelegten Meilensteinen.

Meilenstein

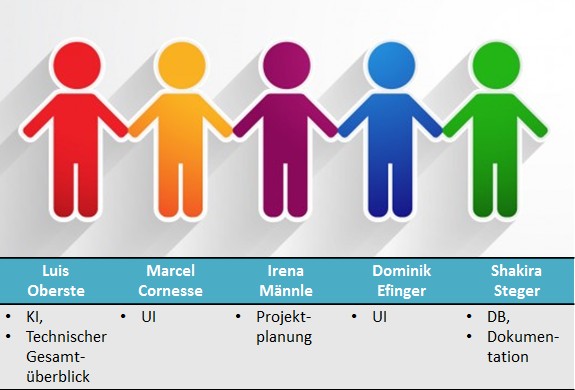
Der Zeitstrahl zeigt den Start des Projektes am 07.09.2015 sowie dessen Ende am 09.11.2015. Ebenfalls zu sehen sind die drei gesetzten Meilensteine. Die Prototyp-Abgabe erfolgt am 28.09.2015. Die Abgabe der Beta-Version erfolgt wiederum zwei Wochen später, am 12.09.2015. Als letzter Meilenstein gilt die letzte Abgabe, das Release am 09.11.2015.

* + 1. Verantwortlichkeiten

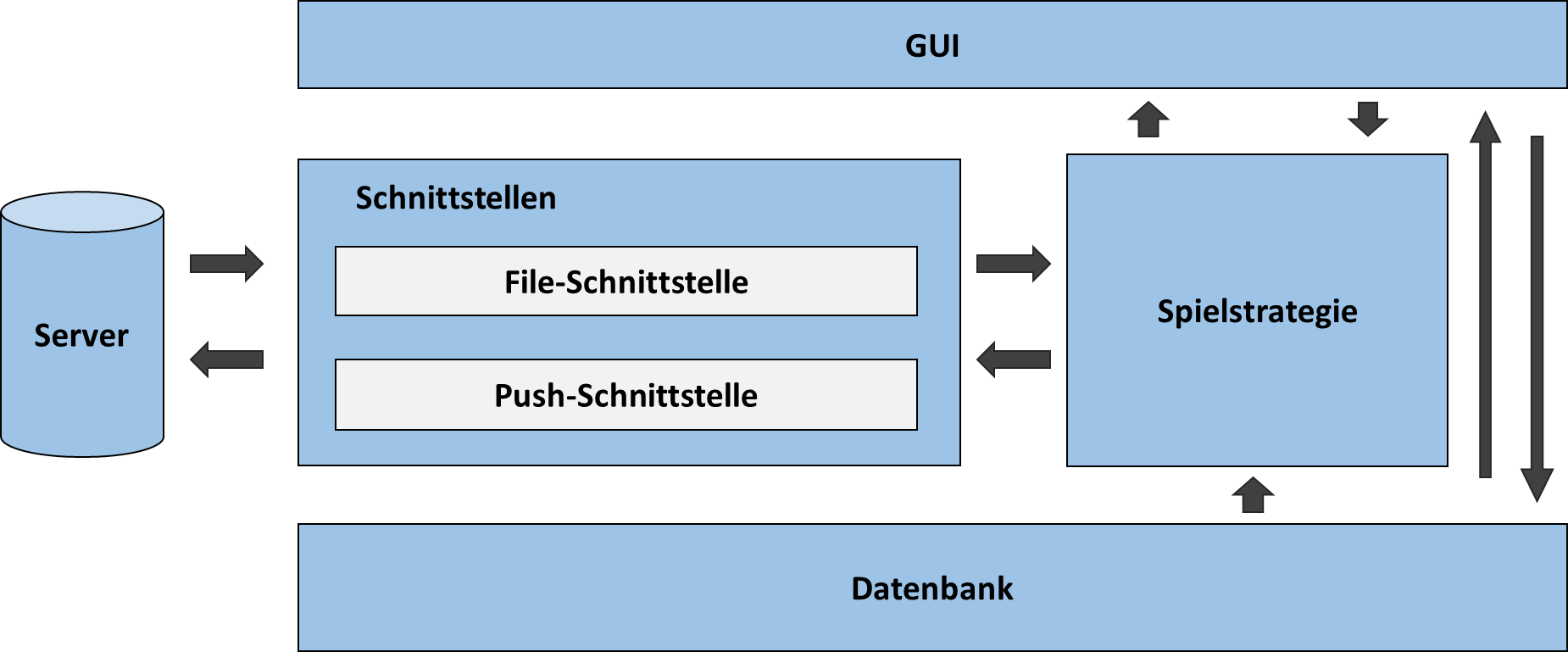
Die Verantwortlichkeiten wurden gleich zu Beginn des Projekts gemeinsam innerhalb des Teams verteilt. Je nach Stärken der jeweiligen Teammitglieder konnten die Verantwortungsbereiche bestmöglich verteilt werden. Auch wenn jedes Mitglied selbst dafür zuständig ist seine Anforderungen zeitgerecht und funktionsfähig abzuliefern, werden auftretende Probleme gemeinsam in der Gruppe angegangen und gelöst.

Im nachfolgenden werden die Aufgabenbereiche der einzelnen Projektmitglieder tabellarisch dargestellt.

**Team fungi**



* 1. Aufbau, Entwurfsentscheidungen
     1. Files & folders
     2. Schichtenarchitektur



* + 1. Aktivitätendiagramm
  1. Konzept
     1. Wie wird was umgesetzt?
     2. Gui -> swing
* Mockups

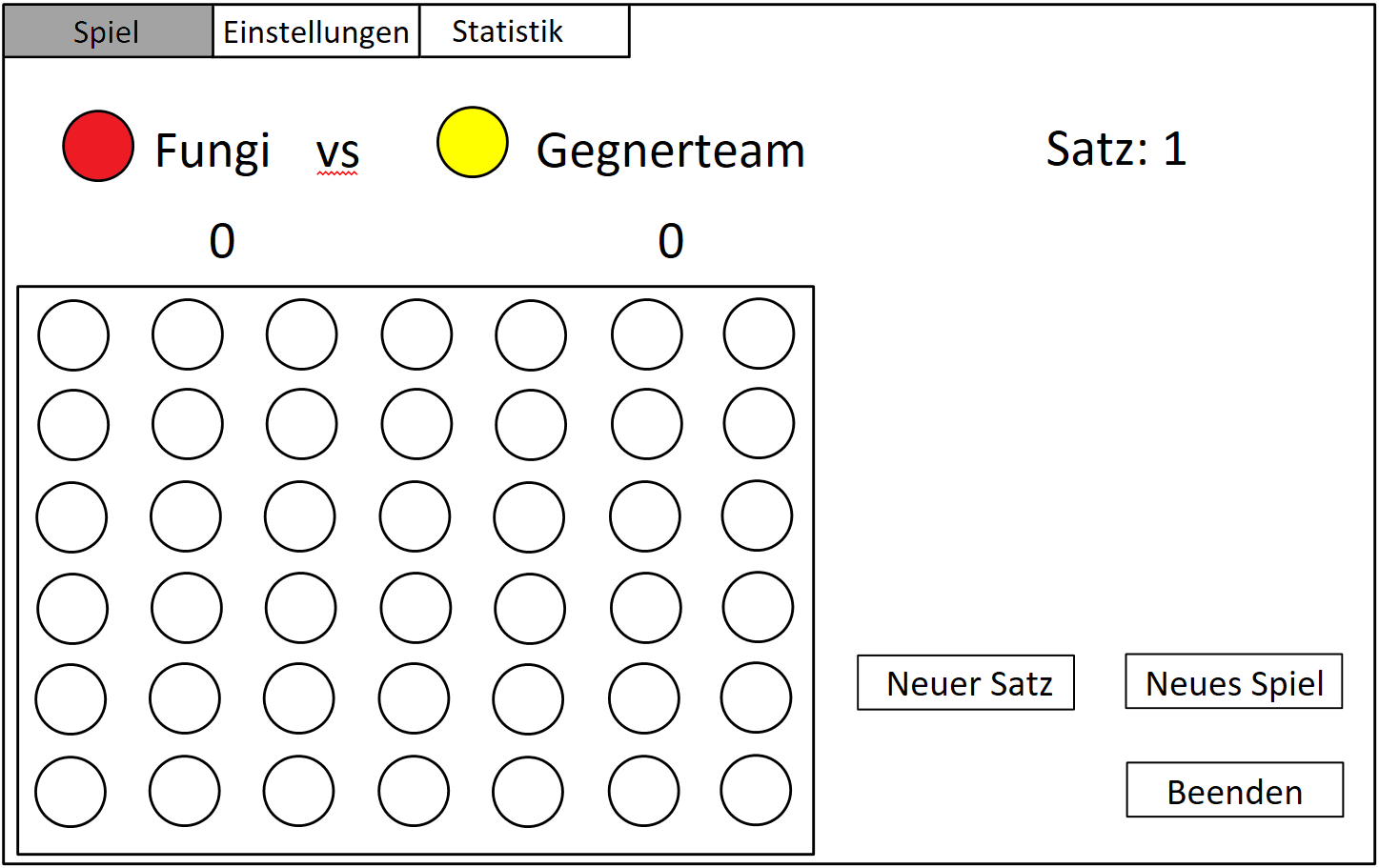


Abbildung 1: Spieloberfläche

Die GUI besteht aus den 3 Registerkarten "Spiel", "Einstellungen" und "Statistik". Im oberen Bereich werden die Teams angezeigt, die das aktuelle Duell bestreiten, sowie der Spielstand und der aktuelle Satz. Der untere Bereich beinhaltet das Spielfeld auf dem die Spielzüge visuell angezeigt werden. Des Weiteren gibt es die Buttons "Neuer Satz", "Neues Spiel" und "Beenden"

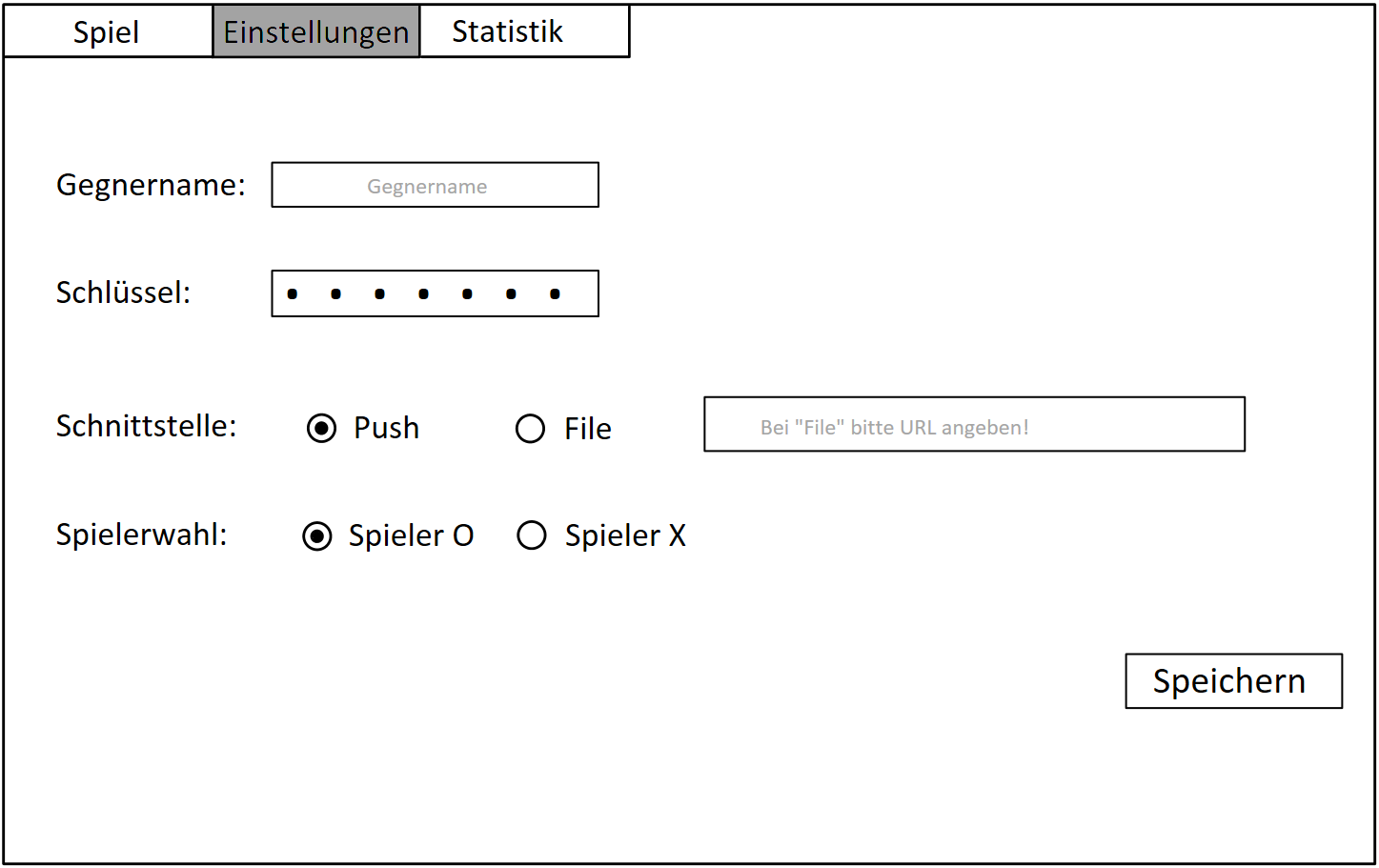
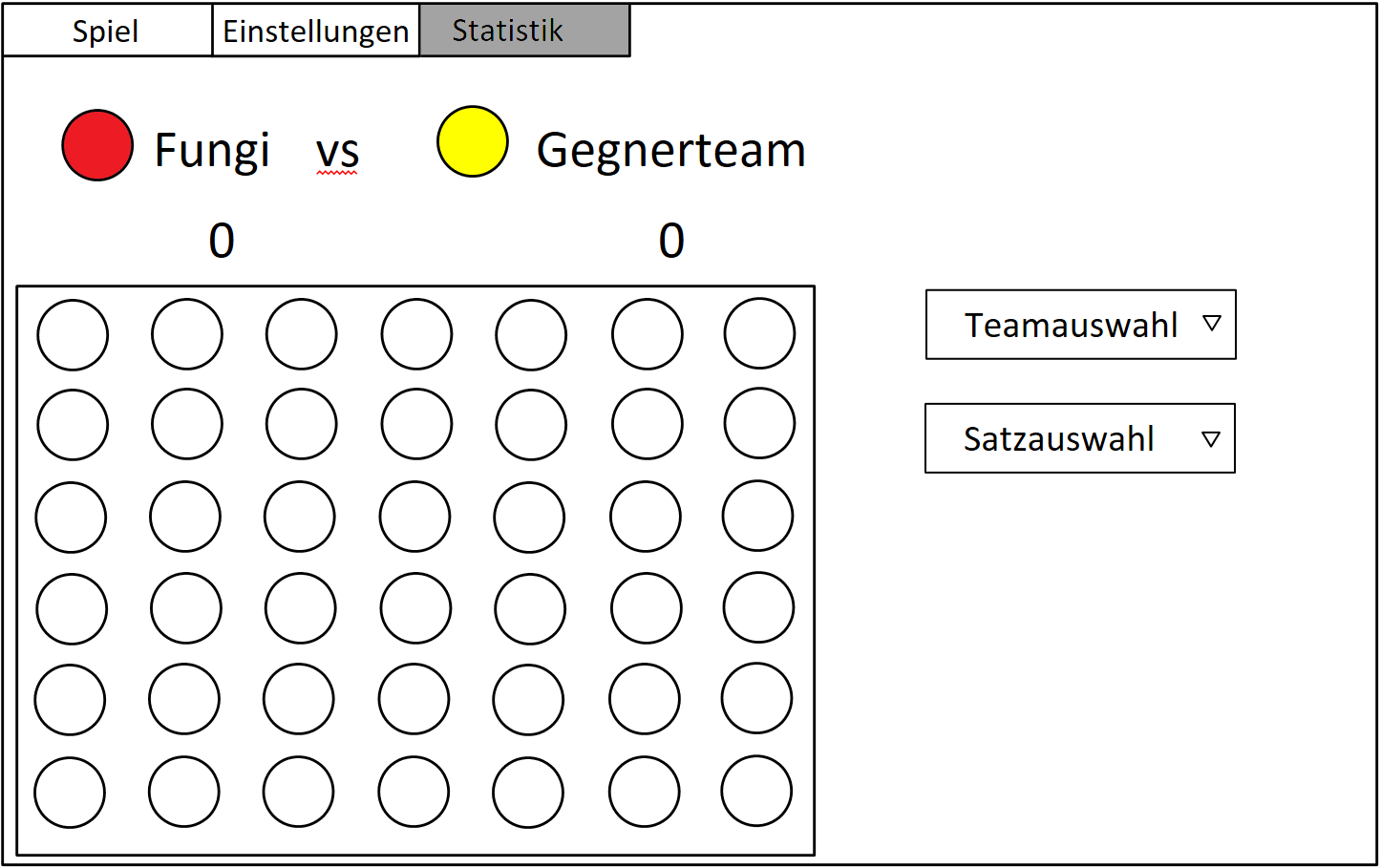
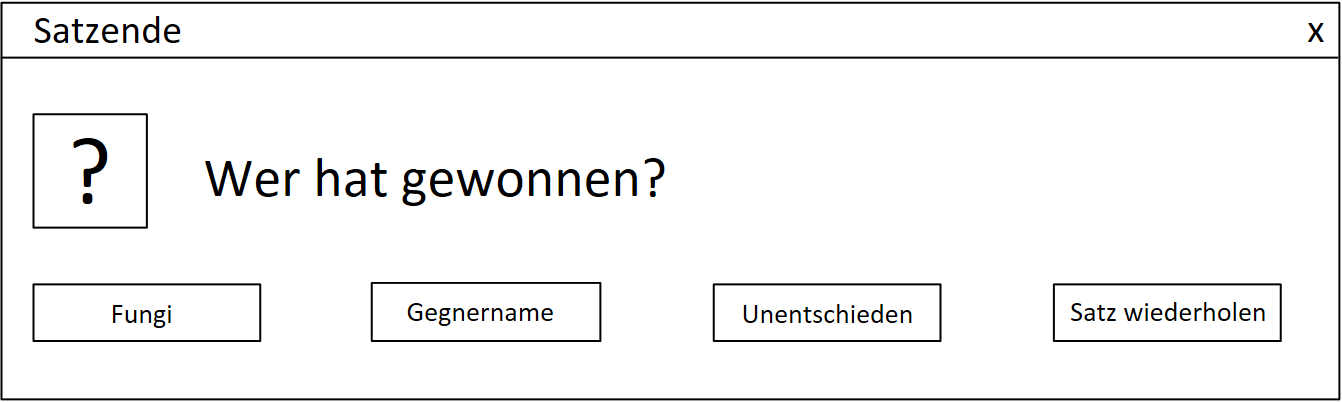


Abbildung 2: Konfigurationsoberfläche

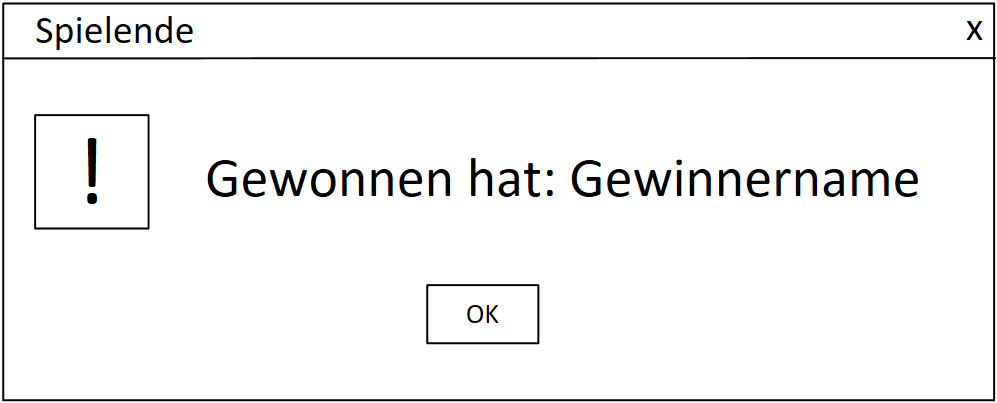
Die "Einstellungs-Seite" dient zur Konfiguration vor dem Spielstart. Einzustellen sind Gegnername, Schlüssel, Schnittstelle und die Spielerwahl. Im Falle der Schnittstelle "File", muss noch der dazugehörige Pfad angegeben werden. Der "Speichern-Button" dient zur Sicherung der Daten in einer lokal gespeicherten ".ini-Datei"



Die "Statistik-Seite" bietet die Möglichkeit den letzten Spielzug vergangener Sätze erneut grafisch darzustellen. Selektiert wird hierfür nach Gegner und Satz. Dies ist möglich, durch zwei Dropdown-Menüs auf der rechten Seite der GUI. Wie bereits auf der "Spiel-Seite" wird auch hier wieder das Spielfeld, sowie die Teamnamen und der Spielstand angezeigt.



Nach jedem Satz erscheint das "Satzende-Popup", dass nach dem Gewinner des Satzes fragt. Auswahlmöglichkeiten sind neben den spielenden Teams sind "Unentschieden" und "Satz wiederholen". "Satz wiederholen" wird benötigt, wenn beispielsweise ein Spieler die Verbindung zum Server verliert und der Satz erneut gespielt wird.



Nach jedem Spiel erscheint das "Spielende-Popup", dass anzeigt, wer das Spiel gewonnen hat. Dies dient lediglich zur Information des Spielers.

* Für gui auf swing zugegriffen
* Keinerlei plugins
* Alles aus akutellen eclipse version
* V0.1 version : GUI entwurf
* Vo.12 : infofenster spielende satzende hinzugefügt
* V0.15: spielfeld angelegt, array-tabelle agiert im Hintergrund eines png Spielfelds
* Tabelle die mittels array befüllt wird
* V0.2 : steuerungslogik hinterlegt save der Konfiguration via filewriter
* hier als Prototyp eingereicht
* V0.3 anpassung der spielfläche und der statistik, konfigurationseinstellungen bleiben beim programmende erhalten, optimierung des quellcodes (säubern, novh mehr beschreibungen ergänzen)
* Dann an luis übergeben zur verknüofung der Anwendungslogik
  + 1. DB > hsqldb
  1. Struktur
     1. Entitäten (Komponenten, Module, Pakete, Klassen)
     2. Deren Beziehungen und Abhängigkeiten
     3. Datenbank (ER-Diagramm)
     4. Klassen beschreiben

Abschnitt: Benutzung

# Installationsanleitung

# Bedienungsanleitung

* 1. Illustrierungen und Beispiele