Universität zu Köln
Institut für Informatik
Manuel Molina Madrid

Weyertal 121,
D-50931 Köln
Tel.: +49-221-70-89791
E-Mail: molina@informatik.uni-koeln.de

Programmierpraktikum 2012

Pflichtenheft

Version 1.0

1. Allgemeines

Das vorliegende Dokument listet Architektur, Features und Qualitätskriterien auf, die zu implementieren bzw. zu berücksichtigen sind. Architektur, Features und Qualitätskriterien sind Bewertungsgrundlage für Ihr Softwareprojekt, welches auf dem in ILIAS zu findenden Quellcode von „Hindi Bones“ beruht. Die Features müssen bis zur Abgabefrist des Projektes (3. Meilenstein) lauffähig implementiert sein.

Abgaben der Meilensteine

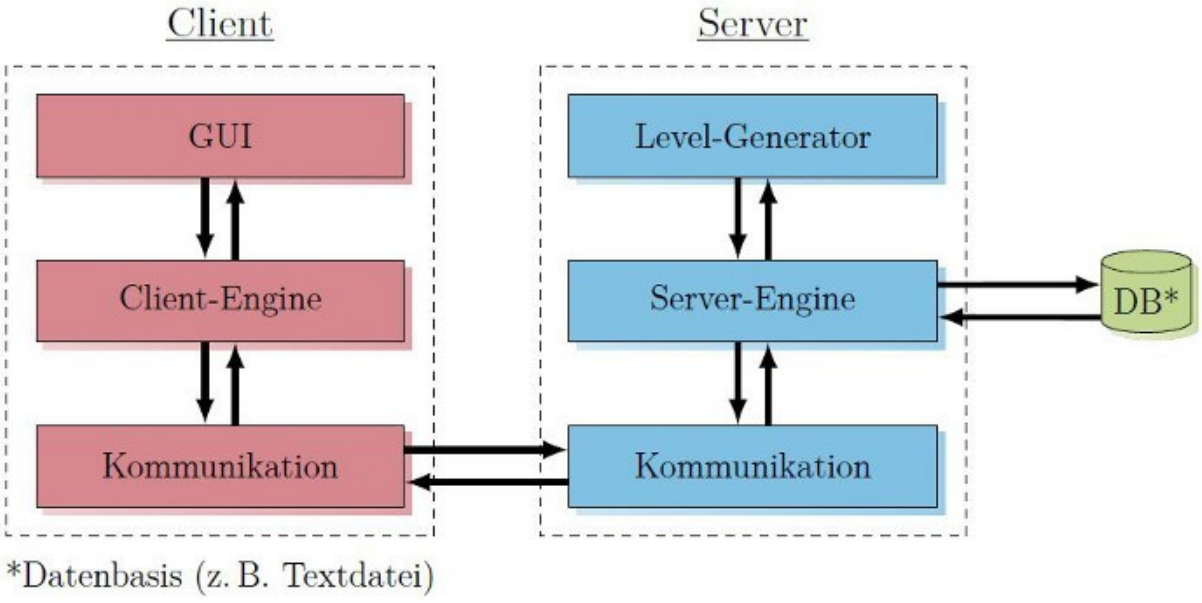
Die Abgaben der Meilensteine werden zu den im Folgenden angegebenen Fristen in ILIAS hochgeladen:

1. Meilenstein: Konzeptioneller Softwareentwurf
   Abgabefrist: 07.06. / 09.06.
2. Meilenstein: Komponente & Testumgebung
   Abgabefrist: 21.06. / 23.06.
3. Meilenstein: Software & Dokumentation
   Abgabefrist: 09.07. / 11.07.

Gruppe01 gibt zum ersten Termin ab. Gruppe02 gibt zum zweiten Termin ab.

2. Architektur

Hier die Architektur, die in ihrer Software umgesetzt werden soll:



Bei der Bewertung wird auf die Umsetzung geachtet. Außerdem müssen die Schnittstellen im Team abgestimmt und umgesetzt werden.

3. Features

Im Folgenden die für das Softwaresystem und die einzelnen Komponenten zu implementierenden Features. Sie sind nach den zu bearbeitenden Aufgabenbereiche durch die Teammitglieder strukturiert.

3.1 GUI

1. Anzeige
    - Öffnen eines Spielfensters mit einem Spielpanel
    - Anzeigen des Levels
    - Anzeige des Charakters und dessen Umgebung (mitscrollend)
    - Anzeige von Items & Gegnern
    - Anzeigen anderer Charaktere (NPC[1] & Mitspieler)
    - Fenster zum Einloggen (mit Verschlüsselung)
    - Fenster für Highscore
    - Öffnen eines Spielfensters mit einem Statuspanel
    - Anzeige der Spielerattribute (z. B. Lebensenergie, Zeit, Level, Tränke, usw.)
    - Anzeige von Systemnachrichten (z. B. Client angemeldet, usw.)
    - Anzeige von Chatnachrichten
    - Anzeigen einer Minimap (Levelanzeige in kleinerem Maßstab)
    - Isometrische Kartenansicht
2. Event-Verarbeitung
    - Spielfenster schließbar
    - Steuerung: Laufen im Level (per Maus)
    - Item-Aktionen: Nehmen (per Leertaste)
    - Angriff: Spieler greift Gegner an (durch Gegenlaufen bzw. draufklicken)
    - Funktion zum geordneten Ausloggen (nicht einfach nur Anwendung schließen)
    - Versenden von Chatnachrichten
    - Item-Aktionen: Benutzen
    - Das Spiel soll über eine Menüleiste verfügen. Die Menüs müssen mindestens die Menüpunkte „Beenden“, „Neustart“, „Highscore“ und „Karte aufdecken“ haben

3.2. Client-Engine

1. Datenstrukturen werden angelegt (Shared)
    - des Message-Systems (inkl. Cheats)
    - des Charakters

1 non player character

Universität zu Köln
Institut für Informatik
Manuel Molina Madrid

Weyertal 121,
D-50931 Köln
Tel.: +49-221-70-89791
E-Mail: molina@informatik.uni-koeln.de

2. Message-Handling von Serverdaten

2.1. Leveldaten

- Level speichern und verwalten
- Leveldaten vom Server empfangen
- Leveldaten werden beim Start (Login) komplett empfangen
- Leveldaten gemäß inkrementeller Updates vom Server anpassen
- Level wechseln

2.2. Verarbeiten von Spielerkommandos

- Laufen (inkl. Konsistenzchecks)
- Items aufnehmen und benutzen (inkl. Konsistenzchecks)

3.3. Kommunikation

Clientseitig

1. Nachrichten (Sende-/Empfangs-Schlange)

- Nachrichten vom Server empfangen
- Nachrichten an Server senden (inkl. Spielerkommandos)

2. An- und abmelden beim Server

- Verbindung zum Server aufbauen
- Leveldaten vom Server empfangen
- Abmelden beim Server (z. B. beim Logout)
- Inkrementelle Updates vom Server empfangen
- Periodisches Senden von „Ich bin noch da!“-Nachrichten

Serverseitig

1. Nachrichten (Sende-/Empfangs-Schlange)

- Nachrichten an Clients senden
- Nachrichten von Clients empfangen

2. An- und abmelden beim Server

- Eigener Thread für jede Client-Verbindung
- Abgebrochene Verbindungen werden erkannt und geordnet beendet
- Clients werden geordnet abgemeldet
- Verbindung von mindestens 2 Clients

3.4. Server-Engine

Für den Aufgabenbereiche der Server-Engine gilt folgendes:

- Spielweltverwaltung: Punkte 1, 2, 3 und 8
- Monstersystem: Punkte 4, 5, 6 und 7
- Levelgenerator: Punkte 9, 10 und 11

1. Message-Handling von Clientdaten (inkl. Konsistenzchecks)

    1.1. Nachrichten verarbeiten

        • Spielernachrichten
        • Item- & Kampfnachrichten

    1.2. Leveldaten

        • Level speichern und verwalten (inkl. Event-Server)
        • Leveldaten werden komplett an den Client übertragen (beim Start)
        • Inkrementelle Levelupdates werden an die Clients verschickt
        • Spieler kann Level wechseln (komplettes übertragen beim ersten betreten)

2. Verwaltung

    • Chat-Nachrichten verarbeiten
    • Verwaltung aller aktueller Level
    • Verwaltung von mindestens 2 Spielern

3. Datenstrukturen werden angelegt (Shared)

    • der Gegner
    • des Charakters

4. Gegnersteuerung

    • A*-Algorithmus für die Bewegung
    • Endlicher Automat: Taktische Steuerung durch direktes Hin-/Weglaufen
        – Gegner greifen an
        – Gegner spazieren herum
        – Gegner flüchten
        – Gegner sterben und verpuffen
        – Trank wird hinterlassen

5. Mehrere verschiedene Gegnertypen werden eingesetzt, diese werden in höheren Leveln stärker und es wird eine größer werdende Anzahl verteilt

6. Verwaltung und Berechnung von Kämpfen
   (z. B. beim Tod dieses aus der Liste austragen)

7. Item-System (Trank & Schlüssel)

    • Datenstrukturen sind fertig (Shared)
    • Aufnehmen/Hinlegen von Items
    • Benutzen der Items wird berechnet (z. B. Lebensenergie)

8. Datenbasis (z. B. als Textdatei)

9. Datenstrukturen werden angelegt (Shared)

    • der Map
    • der Tiles

10. Räume (Labyrinthe)

    • Jeder Raum hat zwei Türen (Eingang/Ausgang)
    • Anlegen eines Startpunkts für Spieler in Startlevel
    • Aufbau eines gesamten Levels als Labyrinth
      (Generierung mit Hilfe das Flood-Fill-Algorithmus)

Universität zu Köln
Institut für Informatik
Manuel Molina Madrid

Weyertal 121,
D-50931 Köln
Tel.: +49-221-70-89791
E-Mail: molina@informatik.uni-koeln.de

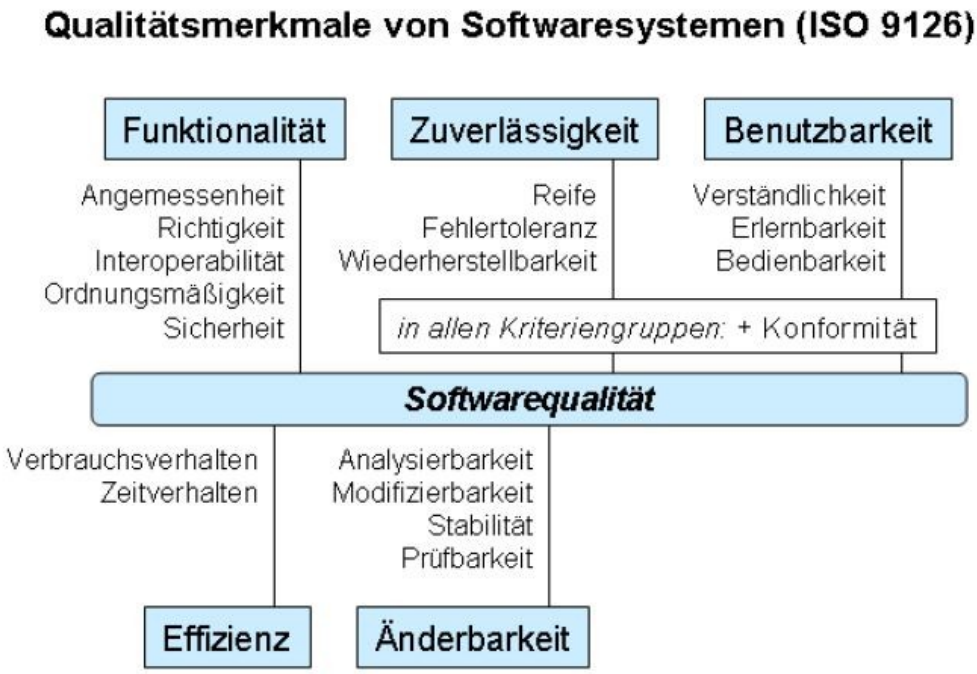
- Labyrinthe miteinander verbinden (Türen einsetzen)
- Raumwechsel nur mit Schlüssel möglich (dieser wird zufällig an ein Gegner verteilt)

11. Zufällig im Labyrinth (in anderen Räumen freigestellt) verteilen
    - Heiltränke
    - Gegner

4. Softwarequalitätskriterien

Im Folgenden werden die Qualitätskriterien aufgeführt, die an das Softwaresystem gestellt werden:

1. Funktionalität (z. B. A* berechnet Weg korrekt, Flood-Fill generiert Labyrinth)
2. Zuverlässigkeit (z. B. Wie Fehlertolerant ist das Programm, stürzt es oft ab)
3. Benutzbarkeit (z. B. Benutzerfreundlichkeit, Spielfreude, Bedienbarkeit)
4. Effizienz (z. B. Zeitverhalten „Latenzzeiten“)
5. Änderbarkeit (z. B. Programmierstil, JavaDoc, Coding Convention, Modularität)



Weitere Anforderungen, die sich auf die Bewertung auswirken, sind:

- Beachten Sie bei der Implementierung, dass Sie keine vom Betriebssystem abhängigen Systemelemente (wie z. B. Fullscreen, Sound) verwenden.
- Denken Sie auch daran **keine Umlaute(!)** zu verwenden.
- Ihr Spiel sollte für alle Rechnersysteme (Windows, Linux und MacOS) lauffähig sein.
- Die Serveradresse soll im Client nach Programmstart geändert werden können, wobei ein entsprechend sinnvoller Standardwert („localhost“) gesetzt ist

Beim Verfassen dieser Aufgabenstellung haben wir uns sehr darum bemüht sie fehler- und widerspruchsfrei zu gestalten. Sollten Sie in diesem Dokument Inhalte vermissen, Fehler oder Widersprüche finden, teilen Sie uns dies bitte mit und schreiben an progprak@informatik.uni-koeln.de oder besuchen Sie die Sprechstunde des Dozenten (dienstags und donnerstags von 14:00 - 15:00 Uhr).