SEP 2019

– Aufgabenblatt 2b –

spätester Abgabetermin: 27.05.2019 12 Uhr

Dieses Blatt richtet sich ausschließlich an die Teilnehmer des ***Softwareentwicklungsprojekts****.*

Ziel des Blattes

Auf dem letzten Aufgabenblatt wurde die Analysephase mit der Erstellung eines Analysemodells abgeschlossen. Auf diesem Aufgabenblatt geht es nun um die Entwurfsphase. Dabei wird das Analysemodell erweitert, konkretisiert und um technische Aspekte ergänzt.

Ihr Entwurfsmodell soll die Verbindung zu den verwendeten Frameworks bzw. zu den verwendeten APIs aufzeigen. Außerdem wird für Konzepte, die bisher nur fachlich definiert waren, sich so aber technisch nicht 1:1 umsetzen lassen, eine technische Lösung gesucht. Oftmals kommen dabei Entwurfsmuster (*Design Patterns*) zum Einsatz. Dies ist besonders interessant, da hier Server und Client getrennt werden.

Aufgaben

1. Erstellen Sie um technische Aspekte verfeinerte Klassendiagramme, die die statische Struktur der Software veranschaulichen.

2. In Ihrer Software soll die Netzwerkkommunikation mittels RMI[1] umgesetzt werden. Achten Sie darauf, dass dies bereits in Ihren Modellen ersichtlich ist.

3. Überlegen Sie sich, welche Exceptions Ihre Klassen werfen werden. Definieren Sie eine Hierarchie fachlicher Exceptions und veranschaulichen Sie diese in einem Klassendiagramm.

4. Beschreiben Sie für einen Computerspieler (Bot) für alle Entscheidungen Handlungsvorgaben. Es kann hierbei gegebenenfalls hilfreich sein einen Zustandsautomaten zu modellieren.

5. Erstellen Sie ein Gantt-Diagramm für die Implementierungsphase. Achten Sie hierbei darauf, dass aus Ihrem Diagramm hervorgeht wer für welchen Teil verantwortlich ist. Diese Informationen sollen auch in Ihrem Diagramm ersichtlich sein.
   Implementierung und Optimierung erstrecken sich über zwei Phasen:

[1] http://openbook.rheinwerk-verlag.de/java7/1507_12_001.html

a) Grundgerüst, Dokumentation, Tests und Implementierung von Chat-, Login- und GUI-bezogenen Funktionen
(Späteste Abgabe: 24.06.)

   i. In dieser Phase sollen die Interfaces/Klassen jeweils mit ihren Methodensignaturen und die Beziehungen der Klassen umgesetzt werden

   ii. Ferner sind Unit-Tests und Java-Doc für alle öffentlichen Methoden zu schreiben.

   iii. Schließlich müssen Chat-, Login- und GUI-bezogenen Funktionen implementiert werden.

b) Erweiterung, Validierung und Optimierung
(Späteste Abgabe: 15.07.)

   i. In dieser Phase soll die Implementierung abgeschlossen werden, indem alle verbleibenden Klassen/Methoden implementiert werden.

   ii. Hier sollen dann auch die auf diesem Blatt entworfenen Bots umgesetzt werden.

6. Bereiten sie sich auf eine Zwischenabnahme vor. Hierbei sollen sie als Team vor ihrem HiWi sowie einem weiteren HiWi präsentieren was sie bisher im Projekt geleistet haben. Jeder Teilnehmer in ihrer Gruppe soll seine eigene Leistung selbst vorstellen. Hierzu ist nicht zwingend eine Beamer Präsentation notwendig, aber falls diese von ihnen als hilfreich angesehen wird können sie eine solche vorbereiten.

Hinweise

1. Wenn Ihr Modellierungstool Beschränkungen hinsichtlich der Gestaltung der Diagramme aufweist, sodass Sie von dem Abweichen müssen, was in der SE2 gelehrt wurde, schreiben Sie das bitte dazu.

2. Wenn Sie absichtlich von dem abweichen, was in der SE2 gelehrt wurde, ist das auch kein Problem, solange Sie das sinnvoll begründen.

3. Das Analysemodell passt vielleicht gerade noch in ein Diagramm. Für das Entwurfsmodell sollten Sie mehrere Diagramme verwenden. Eine Möglichkeit ist, pro Paket ein separates Diagramm zu erstellen.

4. Mittels Stereotypen lassen sich bestimmte Informationen leicht an Klassen „anheften“. So ist es beispielsweise möglich, Remote-Interfaces durch einen Stereotyp `«remote»` zu kennzeichnen, anstatt jedes Mal einen Generalisierungspfeil zu `java.rmi.Remote` zu zeichnen. Auf diese Weise können Sie Ihre Diagramme übersichtlicher gestalten.

5. Bedenken Sie, dass bei RMI alle Daten, die zwischen Client und Server ausgetauscht werden, entweder `Remote` oder `Serializable` sein müssen.

6. Sehen Sie sich die *Java Collection API*[2] an. Sie werden einige dieser Klassen bzw. Interfaces brauchen.

7. Wenn Sie von dem abweichen, was Sie in der Analyse spezifiziert haben, dokumentieren Sie das. Passen Sie das Analysemodells entsprechend an. Bedenken Sie, dass gewisse Änderungen, wie beispielsweise das Aufteilen von Funktionalität von einer Klasse in mehrere, normale Verfeinerungen darstellen können und demnach nicht zwangsweise eine Abweichung vom Analysemodell sind.

8. Es ist nicht nötig alle GUI-Klassen von Client zu modellieren. Es reicht die Klassen zu modellieren, die unmittelbar in der Kommunikation mit dem Server agieren.

9. java.util.concurrent kann einige Hilfe darstellen, sowohl zum Verständnis von Nebenläufigkeit als auch um deadlocks zu vermeiden.[3]. Insbesondere lohnt es sich die Klassen CountDownLatch, CyclicBarrier, and Semaphore anzuschauen.

10. Einige (relativ) kurze Einleitungen zur Nebeläufigkeit auf Deutsch kann man sehr einfach von Google bekommen.

11. Die Verwendung von Design Patterns kann wesentlich hilfreich sein. Insbesondere können die folgenden Patterns relevant sein: Observer (Publish/Subscribe), Facade, Iterator und Strategy.

12. Für Gantt-Diagramme haben sich folgende Programme bewährt:
    a) Microsoft-Project
    b) GanttProject
    c) Planner

13. Versuchen Sie die Botstrategie in einem realen Spiel anzuwenden und testen Sie so die Tauglichkeit.

14. Es kann sich als hilfreich erweisen, einen rudimentären Bot zu entwerfen und darauf aufbauend erst einen intelligenten Bot zu gestalten.

Checkliste zur Vermeidung typischer Fehler

- ☐ Benutzung von RMI wird bei Modellierung berücksichtigt.
- ☐ Modellierung wird mit Beachtung von Java-spezifischen Einzelheiten erledigt.
- ☐ UML-Syntax wird beachtet.
- ☐ Gui und Client sind zwar auf einer Seite, sind aber nicht dasselbe.

[2] http://docs.oracle.com/javase/8/docs/api/java/util/Collections.html
[3] https://docs.oracle.com/javase/8/docs/api/java/util/concurrent/package-summary.html

Kontakt

Website	http://hci.uni-kl.de/~ebert/SEP
Leitung	apl. Prof. Dr. Achim Ebert ebert@cs.uni-kl.de
Organisation	Dr. Taimur Khan
Hiwis	Samir Bouchama
	Roman Reimche
	Matthias Müller
	Jonas Noglik
	Mail an alle sep-support@cs.uni-kl.de