University of Zurich
Department of Informatics

s.e.a.l. software evolution & architecture lab
RERG

Software Engineering Übung 4

Architektur, Entwurf

1 Informationen

1.1 Daten

- Ausgabe: Di. 5.12.2006
- Abgabe: So. 17.12.2006
- Besprechung: Fr. 22.12.2006

1.2 Formales

Die Dateien, welche zu Ihrer Abgabe gehören, müssen in eine .zip-Datei gepackt werden (Diagramme und andere Dokumente als PDF, Quellcode als Textdateien in einem separaten Unterverzeichnis). Die Abgabe erfolgt per Email an cramer@ifi.unizh.ch und an smeier@ifi.unizh.ch.

1.3 Gruppen

Die Übung ist in 3er Gruppen zu lösen. Falls die Aufgaben aufgeteilt wurden, muss klar ersichtlich sein, wer welchen Teil bearbeitet hat. Alle Gruppenmitglieder müssen über alle Teile Auskunft geben können.

2 Ausgangslage und Aufgabenstellung

Anmerkung: Um die folgenden Aufgaben zu Lösen, laden Sie bitte von der Übungswebseite die Datei *CodeU4.zip* herunter. Diese Datei enthält weitere Informationen und den benötigten Quellcode zum Lösen der Übung.

Die PinkAir hat ein bestehendes System für die Buchung von Flügen im Einsatz, das in Java geschrieben ist. Dieses System soll im Zuge einer Markterweiterung bzw. wegen neuer regulatorischer Bestimmungen (separates Ausweisen von Flügen in Länder mit speziellen Sicherheitsvorkehrungen) umgebaut und erweitert werden. In der Reisebranche gibt es zur Zeit zwei führende Standard-Software Produkte (*TravelAgent* und *Earthtrotter*), die unter anderem zur Buchung von Flügen eingesetzt werden können. Buchungen werden durch diese Software direkt an die Buchungsserver der Fluggesellschaften weitergeleitet. Bisher arbeitet das PinkAir-Buchungssystem

aber nur mit der TravelAgent-Software zusammen. Die vorher erwähnten neuen regulatorischen Bestimmungen wurden in der TravelAgent-Software bereits eingebaut, jedoch werden die zusätzlichen Daten noch nicht vom PinkAir-Buchungsserver verarbeitet. Daher sieht der Ablauf bei einer Buchung zur Zeit wie folgt aus:

1. Eine Buchung wird vom Reisebüro mit der TravelAgent-Software erfasst. Dabei werden verschiedene Angaben aufgenommen.
2. Die Einträge der Buchung werden in ein durch die TravelAgent-Software definiertes Format gebracht (siehe weiter unten).
3. Die Einträge werden über das Internet an den PinkAir Buchungsserver geschickt.
4. Auf dem PinkAir-Server werden die einkommendenden Daten geloggt, dazu werden die Daten zu einem Logeintrag umformatiert.
5. Schliesslich werden die Daten in eine Datenbank geschrieben, von wo aus andere Applikationsteile des PinkAir-Servers auf die Daten zugreifen und diese weiterverarbeiten.

Die Daten, welche von der TravelAgent-Software übermittelt werden, haben folgendes Format:

1. Agentur-Identifikationsnummer (*Agency ID*). Dies ist eine beliebige 7stellige Nummer, welche die buchende Agentur eindeutig identifiziert.
2. Nachname (*Name*): Beliebige Buchstabenfolge, bis zu einer Länge von maximal 30 Zeichen.
3. Vorname (*First Name*): Beliebige Buchstabenfolge, bis einer Länge von maximal 30 Zeichen.
4. Nationalität (*Nationality*): Dreistelliger ISO-Buchstabenländercode (CHE für Schweiz).
5. Passnummer (*Passport No*): Beliebige Folge von Buchstaben und Nummern, max. 10 Zeichen.
6. Startflughafen (*From*): Dies ist der Startflughafen als dreistelliges Kürzel in Grossbuchstaben.
7. Zielflughafen Eingabefeld (*To*): Dies ist der Zielflughafen als dreistelliges Kürzel in Grossbuchstaben.
8. Name der Fluggesellschaft (*Airline*): Dieses Feld sollte immer das 3stellige Buchstabenkürzel PKA (PinkAir ICAO-Kürzel) enthalten.
9. Nummer des Flugs (*Flight No*): Eine vierstellige ganzzahlige Nummer.
10. Datum des Abflugs (*Date*): Das Datum im Format YYYY-MM-DD, wobei YYYY das Jahr mit vier Ziffern, MM die Nummer des Monats im Jahr mit zwei Ziffern und DD die Nummer des Tages im Monat mit zwei Ziffern darstellt. Die Monatsnummer liegt zwischen 01 und 12, die Nummer des Tages zwischen 01 und 28 bzw. 31, abhängig vom Monat.
11. Spezielle Sicherheitsvorschriften (*SpecialSecurity*): Falls diese Option gesetzt ist, heisst dies, dass auf einem Anschlussflug oder im Zielland spezielle Sicherheitsvorschriften gelten.
12. Die Ticketnummer: Diese wird durch die TravelAgent-Applikation generiert. Eine Ticketnummer hat das Format XXX*YYY*ZZZ, wobei XXX,YYY und ZZZ Ziffernblöcke darstellen.

Die Firma TravelAgent-Soft, welche die TravelAgent Software entwickelt, hat PinkAir für die Implementierung der ersten Version ihres Buchungsservers einen einfachen Prototypen zur Verfügung gestellt. Mit diesem Prototypen ist es möglich in vereinfachter Form Daten im TravelAgent-Format zu generieren. Die Firma hat diesen Prototypen aktualisiert und die Datenfelder für die Umsetzung der neuen Sicherheitsbestimmungen (Feld *SpecialSecurity*) eingebaut. Es wurde eine einfache graphische Benutzeroberfläche mitgeliefert, um das Generieren der Daten leichter zu handhaben. Dieser Prototyp ist im Quellcodeverzeichnis *src_travelagent* zu finden. Die Klasse *com.travelagent.Booking* enthält die Daten, welche an den PinkAir-Buchungsserver übermittelt werden.
Um nicht alle komplexen Einzelheiten des bestehenden PinkAir-Systems beim Ausarbeiten einer neuen Architektur einbeziehen zu müssen, hat die Software-Entwicklungsabteilung von PinkAir begonnen einen einfachen experimentellen Prototypen zu entwickeln. Eine erste Version des prototypisch implementierten Buchungsservers von PinkAir findet sich im Quellcodeverzeichnis *src_server*.

Die vom TravelAgent-Prototypen generierten Daten werden zum PinkAir-Buchungsserver mittels einer in Java implementierte Kommunkationsschnittstelle namens CSCC geschickt. Die Komponente CSCC ermöglicht es, asynchron Nachrichten bzw. Daten zwischen bliebigen Klienten und Server mit Hilfe des Serialisierungsmechanismus von Java über ein TCP/IP-Netzwerk auszutauschen. Diese Kommunikationsbibliothek soll auch im zukünftigen produktiven System Anwendung finden. Der Übermittlungsmechanismus wurde bereits im TravelAgent- und im PinkAir-Buchungsserver-Prototypen eingebaut. Weitere Einzelheiten zu dieser Kommunikationskomponente finden Sie im Unterverzeichnis *cscc/docs/*.
Um mit dem Buchungsserver- und TravelAgent-Prototypen zu experimentieren, müssen Sie die beiden zuerst starten, indem Sie die Klasse *com.pinkair.Main* im Verzeichnis *src_server* starten. Nun können Sie zwischen *0..n* TravelAgent-Klienten ausführen. Den TravelAgent-Prototypen starten sie durch die Klasse *com.travelagent.Main* auf. Der TravelAgent-Prototyp zeigt eine graphische Benutzeroberfläche, mit welcher Sie Daten generieren können. Nach dem Ausfüllen der Eingabefelder und dem Drücken von *Send* werden die Daten zum Server geschickt, wo diese weiterverarbeitet werden.
Bei der Ausführung des TravelAgent- und Buchungsserver-Prototypen wird angenommen, dass Klienten und Server auf dem selben Rechner ausgeführt werden. Möchten Sie dass der Server auf einem entfernten Computer ausgeführt wird, dann haben Sie die entsprechenden Einstellungen für den Zielcomputer und den TCP/IP-Port im TravelAgent-Prototypen (Klasse *com.travelagent.BookingManager*) vorzunehmen. Achten Sie bei der Ausführung von Klienten und Server darauf, dass Sie Firewalls und entsprechende Schutzprogramme auf den richtigen Port konfiguriert haben oder diese deaktiviert sind. Achten Sie zudem darauf, dass alle Quellverzeichnisse, sowie die Bibliothek cscc.jar im Classpath Ihrer Entwicklungsumgebung eingebunden sind.

2.1 A Architekturüberblick (3 Punkt)

Verschaffen Sie sich zunächst einen Überblick über die Software, wie sie Ihnen vorliegt. Versuchen Sie die Architektur der Software zu visualisieren, indem Sie ein geeignetes Modell erstellen, welches einen Überblick über die Software gibt. Dies können Sie machen indem Sie ein UML-Klassendiagramm oder ein UML-Composite-Structure-Diagram (http://en.wikipedia.org/wiki/Unified_Modeling_Language) verwenden. Versuchen Sie die Software in Komponenten einzuteilen und beschreiben Sie diese.

2.2 B Analyse der verwendeten Architekturstile (2 Punkte)

Untersuchen Sie die Architektur der Anbindung zwischen der CSCC, dem TravelAgent-Klienten und dem Server hinsichtlich der verwendeten Architekturstile. Diskutieren Sie für jeden Architekturstil, den Sie aus der Vorlesung kennen, ob dieser in den genannten Teilen der Software enthalten ist oder nicht.

2.3 C Entwurfsmuster in der Software (2 Punkte)

Untersuchen Sie die folgenden Teile der Software bezüglich der verwendeten Entwurfsmuster, die Sie aus der Vorlesung kennen. Beschreiben Sie die verwendeten Entwurfsmuster. Welchen konkreten Zweck erfüllt jedes der identifitierten Entwurfsmuster? Untersuchen Sie die folgenden Teile auf Entwurfsmuster:

- Paket *com.pinkair.processing*
- Schnittstelle zwischen CSCC, TravelAgent-Klienten und die Schnittstelle zwischen Buchungsserver und CSCC.

2.4 D Architekturanpassungen (8 Punkte)

Wie bereits erwähnt soll der PinkAir-Buchungsserver umgebaut werden. Die Änderungen bestehen hauptsächlich aus zwei Punkten: Erstens soll es weiteren Softwareprodukten ermöglicht werden Buchungen an den PinkAir-Buchungsserver zu schicken. Zum Teil liegen bei denen Buchungsdaten jedoch in einem anderen Datenformat vor. Zweitens sollen zusätzlich neue regulatorische Bestimmungen und weitere Geschäftsregeln im Buchungsserver umgesetzt werden.
PinkAir plant ihren Buchungsserver so zu erweitern, dass Buchungen auch mit der zweiten führenden Standardsoftware für Reisebüros, nämlich *Earthtrotter* von der Firma Earth-Traveller, verarbeitet werden können. Der PinkAir-Buchungsserver soll die Buchungsdaten wie beim TravelAgent-Klienten über die CSCC-Kommunikationsschnittelle empfangen. Intern soll der PinkAir-Buchungsserver wie bisher das Datenformat der TravelAgent-Software verwenden. Auch Earth-Traveller hat PinkAir einen vereinfachten Prototypen zur Verfügung gestellt, mit welchem Buchungen im Datenformat der Earthtrotter-Software erzeugt werden können. Die entsprechenden Quellcodes finden Sie im Quellcodeverzeichnis *src_earthtrotter*.
Leider sind die Datenformate von Earthtrotter und TravelAgent inkompatibel. Das Datenformat von Earthtrotter besteht im Wesentlichen aus einem Java-Vektor (java.util.Vector), welcher mit Instanzen der Klasse *com.earthtrotter.VectorEntry* gefüllt wird. Jede dieser Instanzen enthält ein Feld des Buchungseintrages, wobei jede Instanz von VectorEntry den Namen und den Inhalt des entsprechenden Feldes enthält. Die Namen der Felder sind ähnlich wie im TravelAgent-Klienten, jedoch nicht gleich. Neben der Repräsentation der einzelnen Felder sind zwischen dem TravelAgent und dem Earthtrotter-Format die folgenden Felder unterschiedlich formatiert:

1. Die Ticketnummer: Diese wird durch die Applikation generiert. Sie hat das Format ZZZ–YYY–XXX, d.h. verglichen mit der TravelAgent-Buchungsnummer hat diese andere Trennzeichen und die einzelnen Zahlenglieder (ZZZ,YYY,XXX) sind in umgekehrter Weise angeordnet.
2. Datum des Abflugs: Eingabefeld *Date*. Das Datum im Format DD/MM/YYYY, d.h. das Datum ist anders formatiert als beim TravelAgent-Klienten.

Weiter ist geplant, dass in naher Zukunft auch Buchungen, welche durch verschiedene Internetplattformen von Drittanbietern gemacht werden, ebenfalls mit dem System verarbeitet werden können. Dabei ist die Wahrscheinlichkeit sehr gross, dass die einzelnen Buchungsplattformen ebenfalls eigene Formate haben, welche nicht mit dem von PinkAir verwendeten Standard übereinstimmen.
Zusätzlich sollen einige neue regulatorische Anforderungen und weitere Geschäftsregeln umgesetzt werden, die im bisherigen System noch nicht realisiert sind, was gesamthaft zu folgendem neuen Ablauf im Buchungsserver der PinkAir führt:

1. Die empfangenen Daten werden im Buchungsserver geprüft und sofern nötig ins TravelAgent-Format umkonvertiert. Die Funktionalität für die Umkonvertierung der Datenformate muss noch umgesetzt werden.
2. Die Buchung wird im Transaktions-Log niedergeschrieben. Das Schreiben des Logs für die Transaktionen ist bereits in der Klasse *com.pinkair.logging.TransactionLogger* implementiert und wird auch schon (für die TravelAgent-Daten) verwendet.
3. Es wird geprüft, ob der Passagier, für den der Flug gebucht wurde, in der Blacklist der PinkAir eingetragen ist. Ist der Passagier auf der schwarzen Liste, so wird der Buchungseintrag in eine Datei, welche Buchungen von Passagieren auf der schwarzen Liste enthält, geschrieben. Die Buchung wird danach nicht mehr weiterverarbeitet. Für die Prüfung, ob ein Passagier auf der schwarzen Liste steht, existiert bereits eine Funktion im Prototypen [*com.pinkair.dbconnectorDbConnector.isOnBlackList(String passportNo)*]. Der Rest ist noch nicht umgesetzt.

4. Ist der gebuchte Flug ein Transitflug mit einem Anschlussflug mit besonderen Sicherheitsbestimmungen, oder hat die Destination besondere Sicherheitsbestimmungen, so wird eine Kopie des kompletten Buchungseintrages in eine weitere Log-Datei geschrieben. Diese wird periodisch an die zuständigen Stellen für Sicherheit weitergeleitet. Danach wird der Buchungseintrag normal weiterverarbeitet. Die Funktionalität um einen Eintrag ins Security-Log zu schreiben ist bereits in der Klasse *com.pinkair.logging.SecurityLog* implementiert. Der Rest ist noch nicht umgesetzt.
5. Die Buchung wird wie bisher in die Datenbank geschrieben, von wo aus andere Teile des PinkAir-Informationssystems Zugriff auf die Daten haben. Diese Funktion ist bereits mit der Methode [*com.pinkair.dbconnector.DbConnector.storeInDatabase(com.travelagent.Booking b)*] implementiert und wird bereits verwendet.

Aufgabe: Diskutieren Sie, welche Architekturstile geeignet sind, um die Daten der verschiedenen Buchungsplattformen auf dem Server zusammenzuführen. Entwerfen Sie anschliessend auf Grund dieser Diskussion eine Architektur, welche die im Prototypen noch nicht umgesetzten Anforderungen abdeckt. Erweitern Sie dazu die Architekturbeschreibung aus Aufgabe A) und beschreiben Sie in Worten und mit geeigneten Modellen die hinzugefügten Architekturteile.

2.5 E Implementierung (5 Punkte)

Hinweise: Implementieren Sie Ihre Lösung in die Struktur, die mit der Aufgabenstellung ausgegeben wurden. Geben Sie sämtliche Verzeichnisse (*src_server*, *src_earthtrotter*, *src_travelagent* und *cscc*) Ihrer Implementierung in einer ZIP-Datei gepackt ab. Zusätzlich müssen Sie in einem separaten Lösungsdokument (Format PDF) Ihre Implementierung beschreiben. Sie erleichtern uns damit die Arbeit beim Korrigieren.

a) Im *Earthtrotter*-Prototyp muss noch die fehlende Kommunikationsschnittstelle mit dem Server-Prototyp implementiert werden. Machen Sie dies, indem Sie die entsprechenden Änderungen im Quellcodeverzeichnis *src_earthtrotter* vornehmen.

b) Beurteilen Sie, wie weit Ihnen die Implementierung im Paket *com.pinkair.processing* hilft, um die unter F) entwickelte Architektur umzusetzen?

c) Implementieren Sie Ihren Architekturentwurf, den Sie unter F) erstellt haben. Erstellen Sie dazu zuerst ein detailliertes Design und implementieren Sie dieses anschliessend. Nehmen Sie nötigenfalls Anpassungen im Quellcode des bestehenden Prototypen vor. Ihre Implentierung darf jedoch nichts an den Datenformaten des Earthtrotter- und TravelAgent-Klienten verändern.