Aufgabe 3: Buchhaltungskomponente
Vorführung am 30.10.2013

Lernziele
In dieser Aufgabe geht es darum, das Konzept von Komponenten und deren Architektur, Komponenten-übergreifenden Operationen, komplexeren Persistenz-Mappings, Komponenten- und Integrationstests zu verstehen und am Beispiel einer trivialen Fachlichkeit anwenden zu können.

Aufgabenstellung
In dieser Aufgabe soll die Buchhaltungskomponente mit Basisfunktionalität für Frachtabrechnungen und zugehörigen Gutschriften erstellt werden. Frachtabrechnungen werden von den Frachtführern nach erbrachter Leistung erstellt. In dieser ersten Ausbaustufe pflegen wir diese Abrechnungen allerdings noch „manuell“ über eine entsprechende Schnittstellenoperation in das System ein. Denkbar wäre hier später, dass Frachtführer diese Abrechnungen automatisiert an das HLS schicken.

Die Aufgabe umfasst Folgendes:

- Erstellen Sie die **Buchhaltungskomponente** mit den **Entitäten/DTOs** Frachtabrechnung und Gutschrift (siehe Fachliches Datenmodell in Visio) und den jeweiligen **NHibernate-Mappings**. Das Attribut „Inhalt : PDFTyp“ der Entität Frachtabrechnung können Sie ignorieren. Für die Attribute RechnungsNr und GutschriftNr verwenden Sie den Typ int.
- Erstellen Sie **Fachliche Datentypen (FDT)** für:
    - WährungsTyp (inklusive Operationen zur Umrechnung zwischen EUR und USD)
    - KontodatenTyp (als IBAN/BIC)
- Erstellen Sie eine **Schnittstelle IBuchhaltungServices** für die Komponente, welche die folgenden **Funktionalitäten** definiert:
    - *Erstellen einer Frachtabrechnung*. Die Frachtabrechnung bezieht sich auf einen konkreten Frachtauftrag der Unterbeauftragungkomponente. Falls keine entsprechende Frachtabrechnung dort existiert (entsprechende Prüf-Operation in der dortigen Schnittstelle!), soll ein fachlicher Fehler als Exception zurückgeliefert werden. Denken Sie daran, dass die Kopplung der Entitäten lose mittels Ids erfolgt und nicht als direkt Entitätsreferenzen!
    - *Bezahlen einer Frachtabrechnung*. Hierbei wird eine Gutschrift erstellt und an den BankAdapter geschickt (siehe unten). Des Weiteren soll der Frachtauftrag in der Unterbeauftragungkomponente in diesem Zuge als „abgeschlossen“ markiert werden. Fügen Sie dazu ein entsprechendes neues Attribut in die Entität Frachtauftrag ein und erweitern Sie die Schnittstelle der Unterbeauftragungkomponente um eine entsprechende Operation.
    - *Löschen einer Frachtabrechnung*. Hierbei soll auch eine evtl. vorhandene Gutschrift gelöscht werden (durch eine Cascade-Angabe im Mapping).
- Erstellen Sie eine **Adapterkomponente** „BankAdapter“, die die Kommunikation mit der Bank implementiert (ähnlich wie der bereits vorhandene FrachtführerAdapter). Die Kommunikati-

on soll allerdings **nicht** das Messaging-Framework verwenden, sondern der Adapter fungiert rein als „Dummy“, der die an ihn übergebenen Daten auf einer Konsole ausgeben soll.

- Erstellen Sie für die Buchhaltungskomponente entsprechende **Komponententests**. Die Unterbeauftragungkomponente und der Bankadapter werden in den Komponententests als Mock-Objekte implementiert.
- Erstellen Sie für den Test der Zusammenarbeit der drei Komponenten entsprechende **Integrationstests**.
- Passen Sie die **Projekt-Dokumentation** an (Visio-Modelle, Architekturdokument, etc.), so dass Ihre Entwürfe dort korrekt reflektiert werden. Das Architekturdokument ist nun als Worddatei auf emil vorhanden.

Hinweise:

- Das Visual Studio Projekt der Buchhaltungskomponente muss das Projekt der Unterbeauftragungkomponente referenzieren („Verweise“-Ordner in dem Projekt), um eine Funktion der dortigen Schnittstelle aufrufen zu können.
- Versuchen Sie, vorab die Aufwände für die Tätigkeiten zu **schätzen („Soll“)**. Teilen Sie die obigen Aufgaben in sinnvolle Pakete und notieren Sie Ihre Schätzung (in Stunden/Minuten) in einer Tabelle. Notieren Sie anschließen in der Tabelle ebenfalls die **realen Aufwände („Ist“)**.
- Halten Sie sich unbedingt an die **Architektur-, Codierungs- und Testvorgaben**!

Abnahme

Sie sollten folgende **Fragen im Praktikum** beantworten können:

- Wie ist eine Komponente intern strukturiert?
- Wozu dient ein Adapter?
- Was sind Mock-Objekte und wozu werden sie benutzt?
- Wozu dienen Integrationstests?
- Wie werden Komponenten in dem Architekturdokument dokumentiert?
- Was waren Ihre Schätzungen und wie viel Zeit haben Sie letztendlich benötigt?
- Überlegen Sie sich, über welches Konzept des HLS Sie in der nächsten Vorlesung diskutieren möchten.

Viel Spaß!