Modul 326 – Projekt 22 – Auftrag 2

Musterlösung

Die Musterlösung zu Auftrag 2 dient als Vorlage für den Auftrag 3.

3 Klassenfindung

Für die Klassenfindung wird eine Textanalyse des Auftrags durchgeführt, um Klassenkandidaten zu identifizieren. Dazu wird das Nomenverfahren angewendet.

Das Ergebnis wird mit den UseCase-Beschreibungen verglichen.

3.1 Textanalyse

Die Firma SuperTouperFullComputer ist in einem extremen Wachstum begriffen. Es kommen laufende neue MitarbeiterInnen dazu. Nachteil: man kennt sich nicht mehr, wenn man einander in den Gängen und Räumen der Firma begegnet.

Die Geschäftsleitung möchte dem mit einer Personal-Kennenlern-Anwendung begegnen. Ziel ist es, von jeder Person folgende Daten zu haben

- Foto (Passfoto = Brustbild)
- Vor- und Nachname
- Abteilung
- Funktion
- Projektteam (mehrere Teams möglich)

Die Anwendung kennt zwei Benutzergruppen, nämlich die HR-Personen, die Personen erfassen und Zuteilungen vornehmen können sowie die MitarbeiterInnen, die bestehende Daten nach diversen Kriterien sortieren und durchsuchen können.
Die Stammdaten Abteilung, Funktion, Team werden in einem Administrator-Modus (dritter Modus) erfasst. Dieser Modus ist durch einen Geheimcode geschützt.

Die Anwendung soll grundsätzlich alle Modi umfassen. Administrator- und Erfassen-Modus müssen dabei explizit angefordert und durch einen Nutzercode (Geheimcode) freigeschaltet werden.

Jede Benutzeraktion wird in einem Logbuch eingetragen. Das Logbuch lässt sich im Administrator-Modus anzeigen.

Das Löschen von Daten entspricht bezüglich der Zugänglichkeit jenem des Erfassens (Administrator- und Erfassen-Modus)

Die ausgewählten Daten sollen in einer tabellarischen Form angezeigt werden.

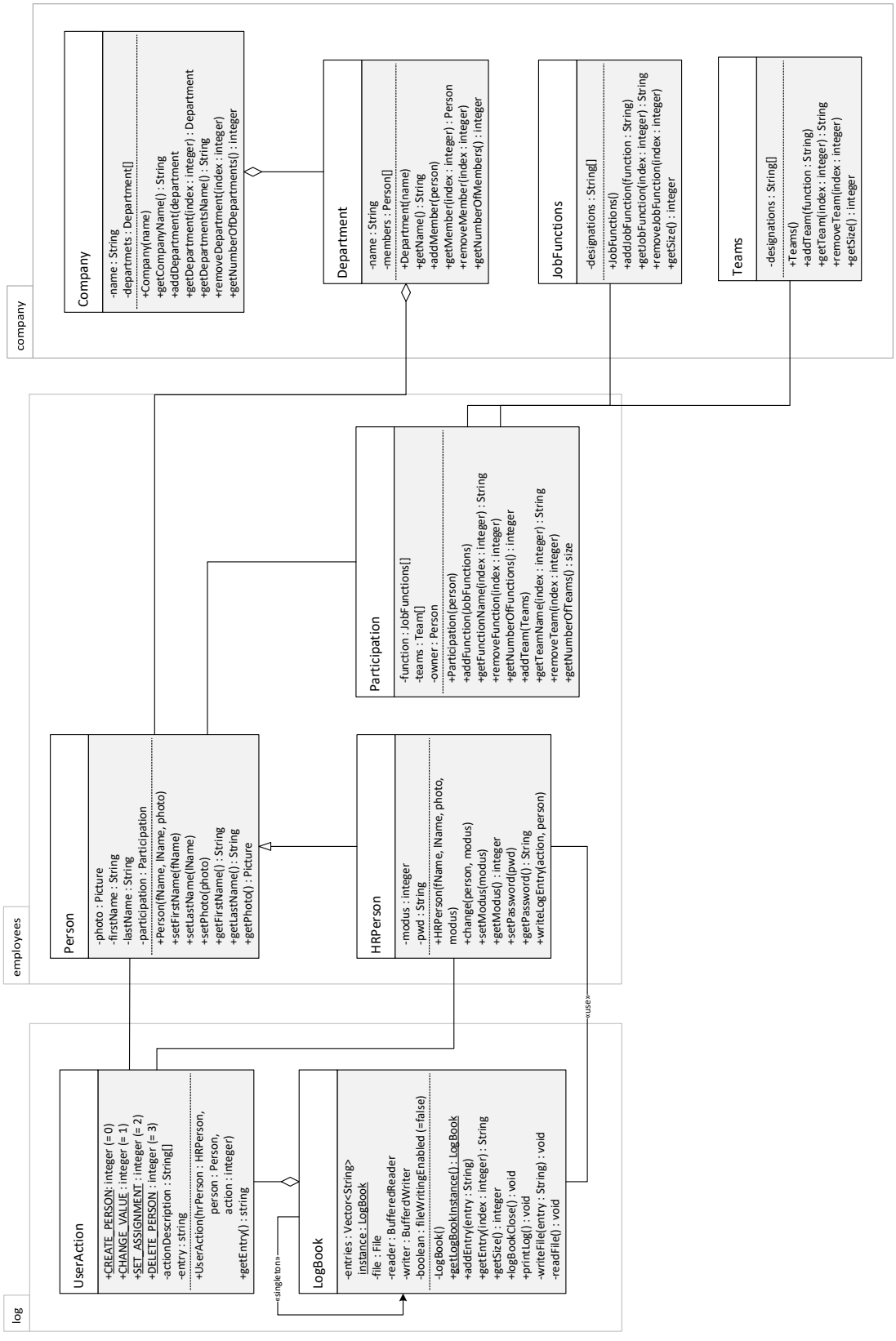
Klassenkandidaten	***Mögliche Attribute***	***Ohne Bedeutung***
Firma	Foto, Passfoto, Brustbild	Wachstum, Nachteil, Gänge, Räume, Geschäftsleitung, Daten, Anwendung, Zuteilungen, Stammdaten, Löschen, Zugänglichkeit, Erfassens, Form
MitarbeiterInnen, Person	Vor-, Nachname	
Benutzergruppen	Abteilung	
HR-Personen	Funktion	
Kriterien	Projektteam, Team	
Modus, Modi	Administrator-Modus	
Benutzeraktion	Geheimcode, Nutzercode	
Logbuch	Erfassen-Modus	

3.2 Abgleich mit UseCase-Beschreibungen

Klassenkandidaten	***Mögliche Attribute***	***aus UseCase***
Firma	Foto, Passfoto, Brustbild	
MitarbeiterInnen, Person	Vor-, Nachname	
Benutzergruppen	Abteilung	
HR-Personen	Funktion	
Kriterien	Projektteam, Team	Suchmuster, Filter
Modus, Modi	Administrator-Modus	
Benutzeraktion	Geheimcode, Nutzercode	
Logbuch	Erfassen-Modus	

Basierend auf der Textanalyse wird ein erstes Klassendiagramm festgelegt. Dieses muss im Verlauf der Software-Entwicklung den jeweiligen Erkenntnissen angepasst werden.

4 Klassendiagramm

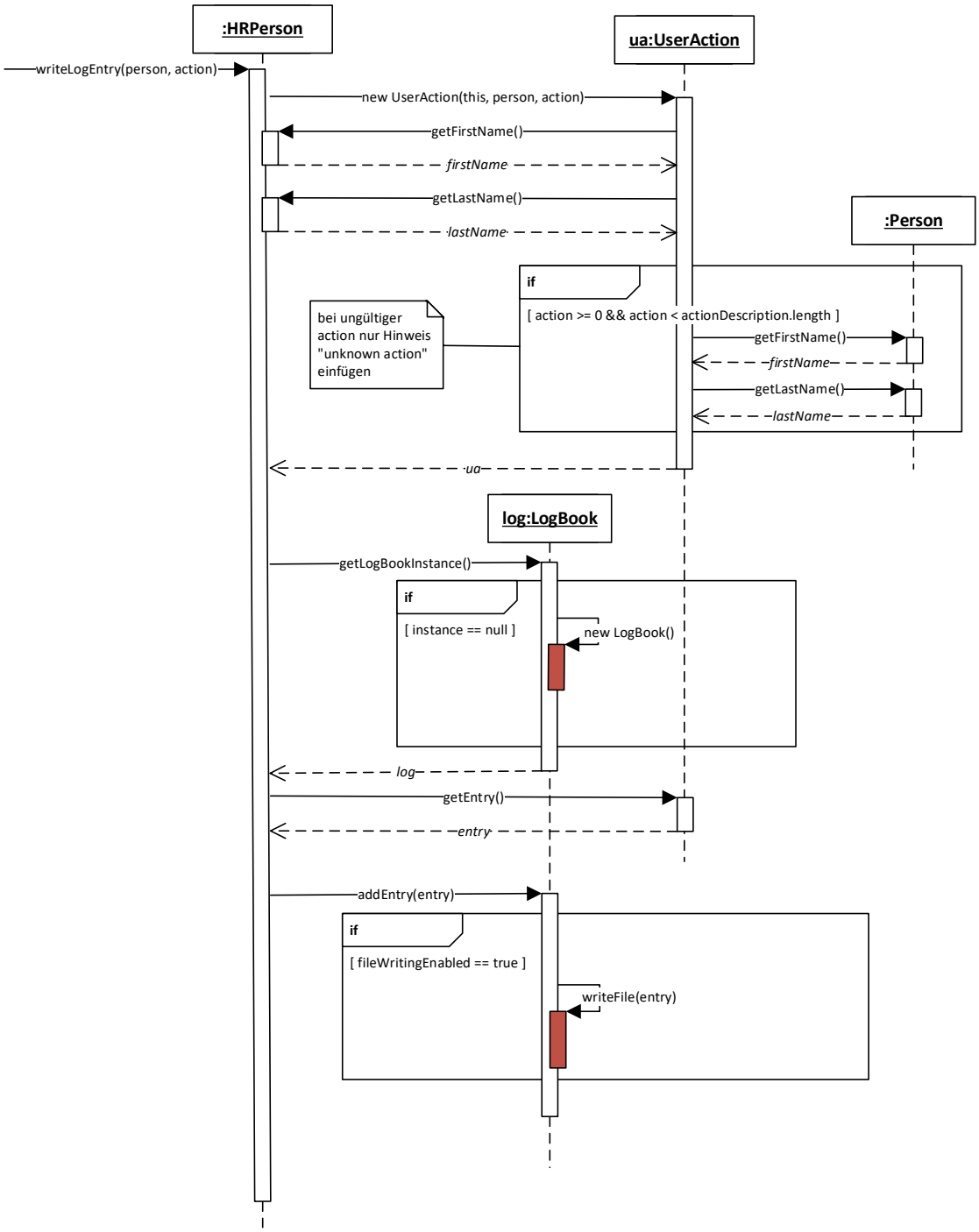


Anmerkung:

- Die Attributkandidaten «Abteilung», «Funktion» und «Team» wurden zu eigenen Klassen. Das wird bei der Erfassung der Daten von Vorteil sein, wenn entsprechende GUI-Komponenten zugeordnet werden.
- Der Klassenkandidat «Modus» wird in ein Attribut der Klasse HRPerson überführt.
- Der Klassenkandidat «Benutzergruppen» findet sich nur indirekt wieder in der Vererbung von HRPerson zu Person. Ob später eine Klasse die Zuordnung der Person zu einer Benutzergruppe realisiert, wird sich bei der Umsetzung zeigen.
- Der Klassenkandidat «Kriterium» entfällt vorerst ganz. Er steht in Beziehung mit der Such- und Sortierfunktion des GUI. Hier wird wohl erst später erkennbar, ob es eine solche Klasse braucht.
- Die Beziehung HRPerson ←→ LogBook ist noch nicht geklärt. Sie hängt – wie auch die Bearbeitung der Stammdaten – davon ab, dass der Benutzer der Klasse HRPerson angehört und über den entsprechenden Modus (HR-Admin) verfügt.
- HR-Admin wird nicht als eigen Klasse implementiert. Es ist das Attribut «modus», das die konkrete Rolle (HR-Person bzw. HR-Admin) signalisiert.

5 Sequenzdiagramme

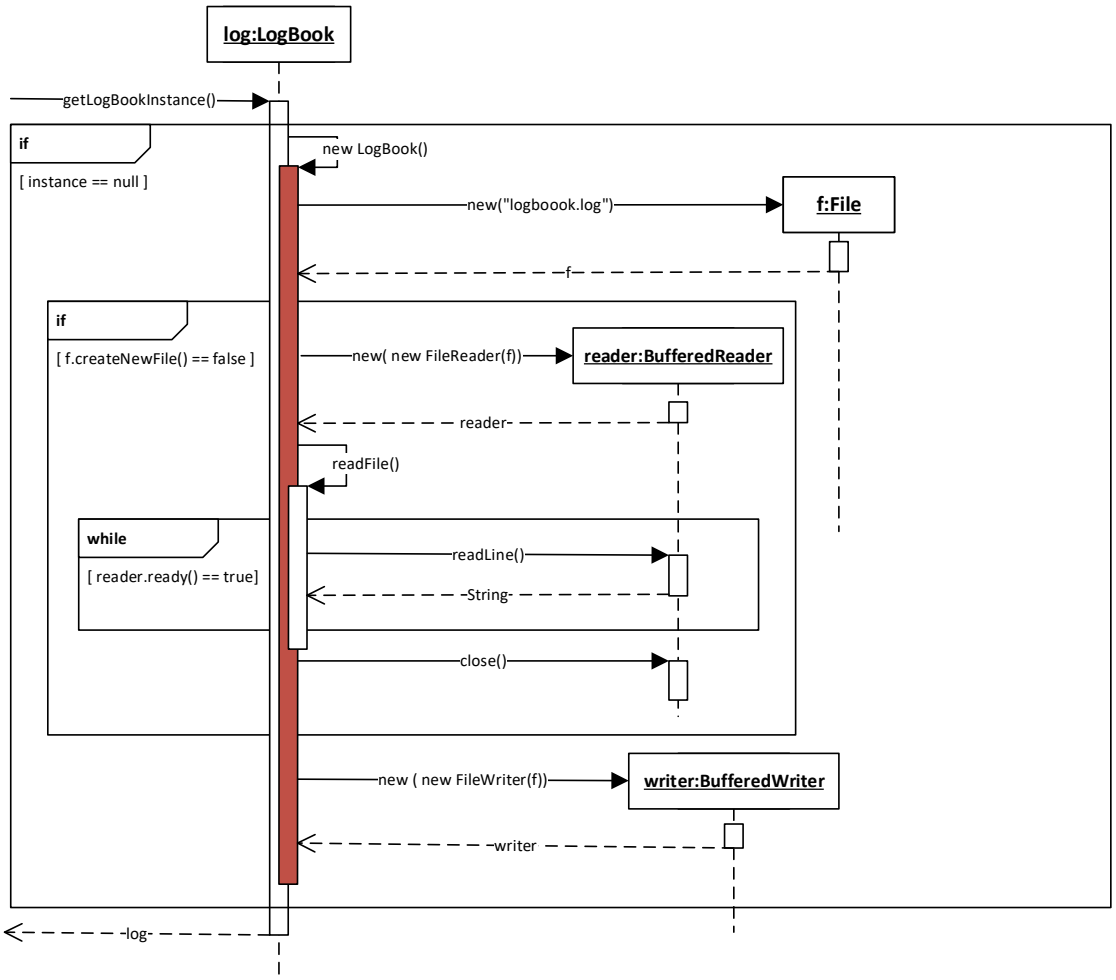
5.1 Übersicht für das Erstellen eines Logbuch-Eintrags



Anmerkung:

- Die Musterlösung ist sehr umfassend ausgeführt. Ihre Lösung muss nicht in diesem Detailierungsgrad vorliegen.
- Das gezeigte Sequenzdiagramm dient dazu, die Klassen `UserAction` und `LogBook` möglichst effizient und korrekt zu implementieren.
- Da der Umgang mit Dateien im Unterricht kein Thema war, finden Sie zusätzlich weiter Sequenzdiagramme, welche die rot markierten stellen im Detail noch zeigen.
- Sie müssen aber auch die Beschreibungen der Klassen `File` und `BufferedReader` bzw. `BufferedWriter` in der Java-API genau studieren.

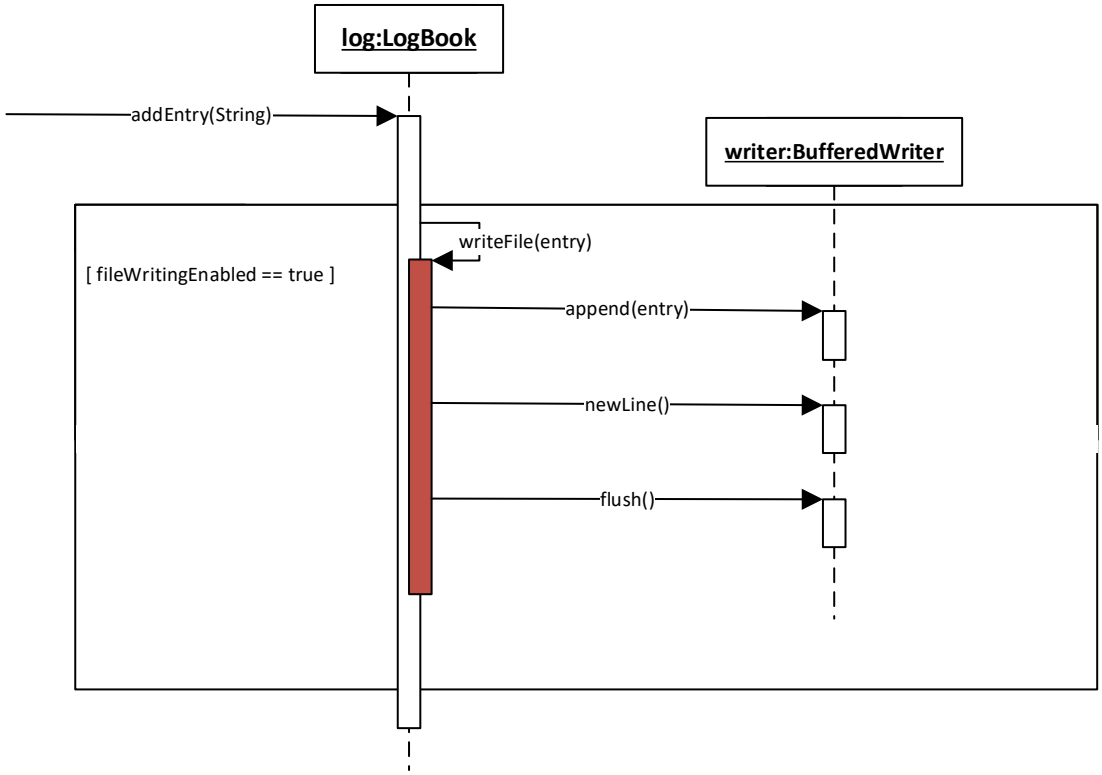
5.2 Logbuch öffnen bzw. neu erstellen



Erklärung:

- Die Datei wird in eine Collection eingelesen und so für die Nutzung innerhalb der Anwendung bereitgestellt (Admin kann Logbuch einsehen)
- Ist die Logbuch-Datei ausgelesen, wird der Reader geschlossen (close-Befehl) und die Datei für den Schreibzugriff geöffnet (Writer).

5.3 Logbucheintrag erstellen



Erklärung:

- Jeder Eintrag wird als eine Textzeile am Ende der Datei abgelegt.
- Damit kein Datenverlust entsteht, sollte die Anwendung unerwartet terminieren, wird das Schreiben der Datei durch `flush` erzwungen.
  (Programme schreiben ihre Daten in einen Buffer, der erst in die Datei übertragen wird, wenn der Buffer voll ist.)
- Wird das Programm beendet, muss der `writer` geschlossen werden. Das erfolgt in der Methode `logbookClose` und den Befehl `close` auf das Objekt `writer`.