technische universität
dortmund

Aktuelle Themen der Dienstleistungsinformatik

Projektbeschreibungen
WS 2012/2013

Bernhard Steffen
Markus Doedt

Lehrstuhl für Programmiersysteme

Generelle Bemerkungen

- Es werden zwei Projekte in jeweils in Gruppenarbeit bearbeitet.
- Die beiden Projekte sind thematisch verwandt und sind beide Teil eines gemeinsamen Szenarios.
- Jede Gruppe verfasst nach der praktischen Arbeit am Projekt einen schriftlichen Bericht über das Projekt, d.h. über die Aufgabe, die Herangehensweise, die theoretischen und praktischen Grundlagen, die aufgetretenen Probleme, die Lösungen zu diesen Problemen, etc.

  Der Text ist für eine Leserschaft zu schreiben, die zwar aus Informatikern besteht, die jedoch keine Kenntnisse im konkreten Anwendungsbereich haben muss (Webservices, SOA, BPM).

  Der Text ist in LaTeX zu verfassen.

Gemeinsames Szenario: Migration SAP → Google

Die Firma „IDES AG“ verwaltet alle Daten des Unternehmens mittels SAP-Software, konkret mit einem **SAP-ERP**. Dieses System hat sich auf der einen Seite als sehr komplex in der Bedienung erwiesen, auf der anderen Seite bedeutet der Einsatz von SAP-ERP einen erheblichen Kostenaufwand. Um in Zukunft Kosten zu senken, sich weniger abhängig von SAP zu machen und eine einfacher zu bedienende Software zu haben, plant die IDES AG nun eine schrittweise Migration auf kostenlose **Cloud-Dienste der Firma Google**. So können z.B. Personendaten in „Google Contacts“, Aufgaben in „Google Tasks“, Termine in „Google Calendar“ und Dokumente in „Google Docs“ verwaltet und bearbeitet werden.

Projekt A (Seite 1 von 2)

Die in SAP-ERP verwalteten Kundendaten sollen nach und nach in eine Kundendatenbank auf Basis von „Google Contacts“ migriert werden. Dabei sollen nicht einfach alle Datensätze kopiert werden, da viele Kundendatensätze in SAP-System veraltete „Karteileichen“ darstellen. Stattdessen stellt sich das Unternehmen folgenden Migrationsprozess vor:

Bei einer Geschäftsbeziehung mit einem Kunden (z.B. wenn dieser eine Anfrage an das Unternehmen stellt) wird der Kundenname aufgenommen. Falls schon ein Kunde mit dem Namen im Google-System vorhanden ist, muss der Bearbeiter entscheiden, ob dies genau der Kunde ist. Falls ja, ist der Prozess beendet, der Kunde wurde gefunden. Falls nein, wird im SAP-ERP nach dem Kunden gesucht. Hier muss der Bearbeiter ebenfalls evtl. entscheiden, ob ein gefundener Kunde genau der ist, der gesucht wird. Falls dem so ist, sollen die Kundendaten vom SAP-ERP zu Google kopiert werden (bei SAP lesen, bei Google schreiben). Ansonsten werden die Daten manuell eingegeben und bei Google gespeichert.

Projekt A (Seite 2 von 2)

- Recherchieren Sie, wie die benötigten Services (bei Google und SAP) aufgerufen werden. Zum Testen bietet sich hier das Tool SOAP-UI an.
- Als SAP-System nehmen Sie die Installation im ES Workplace von SAP: http://esworkplace.sap.com
- Information zur API von Google Contacts finden Sie hier: https://developers.google.com/google-apps/contacts/v3/
- Die Umsetzung der Prozesse soll mit jABC erfolgen. Erstellen Sie für die Services entsprechende SIBs und modellieren Sie den oben genannten Prozess. Testen Sie den Prozess mit dem Tracer.
- Entwickeln Sie noch zwei Varianten des Prozesses: Sowohl für Lieferanten als auf für Mitarbeiter sollen analoge Prozesse abgearbeitet werden.

Projekt B (Seite 1 von 2)

Die im SAP-ERP verwalteten Aufträge von Kunden („Sales Order“) sollen ebenfalls (siehe Projekt A) ab jetzt mit Google verwaltet werden. Dazu soll zu einem bestehenden Auftrag eine entsprechende Tabellenkalkulation in „Google Spreadsheets“ angelegt werden. Der Prozess sieht folgendermaßen aus:

Der Bearbeiter möchte gerne Informationen zu einem Auftrag. Dazu gibt er die Auftragsnummer ein. Nun wird bei Google Spreadsheets geguckt, ob dort zu dem Auftrag schon eine Tabellenkalkulation existiert. Wenn ja, wird diese heruntergeladen und angezeigt. Andernfalls wird der entsprechende Auftrag aus dem SAP-System gelesen und anschließend als Tabellenkalkulation in Google Spreadsheets angelegt.

Projekt B (Seite 2 von 2)

- Recherchieren Sie, wie die benötigten Services (bei Google und SAP) aufgerufen werden. Zum Testen bietet sich hier das Tool SOAP-UI an.
- Als SAP-System nehmen Sie die Installation im ES Workplace von SAP: http://esworkplace.sap.com
- Information zur API von Google Spreadsheets finden Sie hier: http://code.google.com/apis/spreadsheets/
- Information zur API von Google Drive finden Sie hier: https://developers.google.com/drive (hat die ehemalige „Google Docs API“ ersetzt https://developers.google.com/google-apps/documents-list/)
- Die Umsetzung der Prozesse soll mit jABC erfolgen. Erstellen Sie für die Services entsprechende SIBs und modellieren Sie den oben genannten Prozess. Testen Sie den Prozess mit dem Tracer.

SIB-Implementierung

- Ein SIB ist eine Java-Klasse, die mit der Annotation @SIBClass markiert wurde. Eine minimale Implementierung sieht also so aus:

```
package sib.tutorial;

import de.metaframe.jabc.framework.sib.annotation.SIBClass;

@SIBClass("4711")
public class MinimalSIB {

}
```

- Der Wert in der Annotaion @SIBClass ist eine eindeutige ID für den SIB.
- Dieses SIB ist noch nicht ausführbar.

SIB-Implementierung: Parameter

- Die Parameter eines SIBs sind alle Attribute der dazugehörigen Klasse, die als “public“ aber nicht als “static“ deklariert sind.
- SIB-Parameter müssen mit einem Wert initialisiert werden.
- SIB-Parameter sind zur Laufzeit Konstanten.
- Für Laufzeitdaten benutzt man den Kontext (z.B. per ContextKey).
- Beispiel:

```
package sib.tutorial;
import de.metaframe.jabc.framework.sib.annotation.SIBClass;

@SIBClass("4713")
public class ParameterSIB {
    public String StringParameter = "default value";
    public Integer IntegerParameter = new Integer(1);
    public ContextKey someContextKey = new ContextKey("someKey");
    private String dummy; // this is not a SIB Parameter !!!
}
```

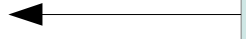
SIB-Implementierung: Ausführbare SIBs

- Damit ein SIB ausführbar ist, muss es das Interface “Executable“ implementieren. Die dazugehörige Methode “trace“ enthält den Code, den der Tracer für das SIB ausführt. Die Methode bekommt den Context als Parameter übergeben und gibt den zu wählenden Branch als String zurück. Ein Beispiel könnte also so aussehen:

```
package sib.tutorial;

import de.metaframe.jabc.framework.execution.ExecutionEnvironment;
import de.metaframe.jabc.framework.sib.annotation.SIBClass;
import de.metaframe.jabc.sib.Executable;

@SIBClass("1234567890")
public class MySIB implements Executable {
        public final String[] BRANCHES = { "default", "error" };
        public String trace(ExecutionEnvironment env) {
                boolean success = doSomethingUseful();
                return success ? "default" : "error";
        }
}
```



Branches müssen genau so deklariert sein!

SIB-Implementierung: Deployment

- SIBs können auf verschiedene Weise bereitgestellt werden:
  - projektspezifisch:

    Ein jar-Archiv oder ein Ordner mit den kompilierten SIB-Klassen wird im Projekt als SIB-Pfad hinzugefügt.

  - global:

    Ein jar-Archiv mit den kompilierten SIB-Klassen wird in den „sib“-Ordner der jABC-Installation abgelegt. Dabei muss sich die Datei an die Konvention halten, dass deren Namen die Zeichenfolge „sibs“ enthält (Beispiel: „sap-sibs.jar“).