ENTWICKLUNG EINES DATEITRANSFER-KLIENTEN MITTELS METHODEN DER MODELLGETRIEBENEN SOFTWARE-ENTWICKLUNG

Mario Kaulmann[1], Naoufel Frioui[1] and Carole Noutchegueme[1]

[1] *Fachbereich Informatik und Medien, Technische Hochschule Brandenburg, Magdeburger Straße 50, Brandenburg an der Havel, Deutschland*

Keywords: MDSD, modellgetriebene Software-Entwicklung, md2, Dateitransfer, REST, Android

Abstract: Das wird später geschrieben

1 EINLEITUNG

1.1 Motivation

Modellgetriebene Software-Entwicklung (MDSD) soll dazu dienen das Entwickeln von Anwendungen so zu vereinfachen, dass Menschen ohne Programmierkenntnisse, Anwendungen erstellen können, die für einen speziellen Anwendungsbereich eingesetzt werden können.
Das Entwickeln einer Smartphone-App ist eine komplexe Aufgabe. Wenn diese App mit einem Server kommunizieren soll, dann wird die Komplexität dieser Aufgabe noch erhöht.
Um trotzdem Ergebnisse erzielen zu können gibt es bereits Ansätze, mit deren Hilfe man durch Beschreibungen des Problems Code erzeugen kann, durch den eine lauffähige Anwendung entsteht.

1.2 Ziel

Das Ziel besteht darin eine Android-App mit Hilfe von Methoden der MDSD zu erstellen. Die App soll über eine REST-Schnittstelle Dateien auf einen Server laden können und Dateien von diesem Server runterladen können. Damit verschiedene Nutzer den Dateitransfer-Dienst nutzen können, soll auch eine Funktion bereitgestellt werden, die es ermöglicht, dass Nutzer sich registrieren, anmelden und abmelden können. Diese Funktionen werden von dem Server bereitgestellt und sind auch über eine REST-Schnittstelle nutzbar.
Bei der Umsetzung dieses Vorhabens wird herausgestellt, was mit den zum Zeitpunkt der Arbeit verfügbaren Mitteln möglich ist und welche Grenzen es noch gibt.

1.3 Aufgabenstellung

Zum Vergleichen des Ergebnisses, das mit Methoden der MDSD erstellt wird, wird vorher ein Prototyp erstellt, der den vollen Funktionsumfang bietet und auf klassische Weise programmiert wird.
Im ersten Schritt der Entwicklung werden Mittel zur modellgetriebenen Entwicklung eines Android-Klienten ausgesucht. Im zweiten Schritt wird versucht den Prototyp des Klienten zu entwickeln. Dabei wird der erzeugte Code anschließend in Android Studio geöffnet und kompiliert. Anschließend wird die App getestet und mit der klassisch programmierten App verglichen, um die aktuellen Grenzen aufzuzeigen und herauszufinden, ob der Prototyp die Anforderungen erfüllt.
Um den Funktionsumfang erfüllen zu können werden einige Teile bei der MDSD-App mit selbstgeschriebenen Code ergänzt.

1.4 Abgrenzung

In dieser Arbeit soll nur ein Android-Klient erstellt werden, der mit einem Server über eine REST-Schnittstelle kommuniziert. Es ist nicht Bestandteil dieser Arbeit eine Server-Anwendung zu erstellen, die die REST-Schnittstelle zur Verfügung stellt.
Das Produkt dieser Arbeit ist ein Prototyp, der durch verschiedene MDSD-Methoden erstellt wird. Der Anteil des selber geschriebenen Codes wird dabei versucht möglichst klein zu halten.

1.5 Ergebnis

2 VORGEHEN

3 WERKZEUGE ZUR ENTWICKLUNG

3.1 MD^2

MD^2-DSL ist eine domänenspezifische Sprache (DSL), die entwickelt wurde um datengetriebene Business-Apps in textueller Form beschreiben zu können (Heitkötter et al., 2013).
MD^2 ist ein akademischer Prototyp, der auch Anwendung in praktischen Projekten findet und dabei auch weiterentwickelt wird (Majchrzak et al., 2015).

3.2 Androidstudio

Wir benutzen diese freie Integrierte Entwiclungsumgebung fr Android. Damit testen wir unsere app auf Android gerte. Hier gibt es auch die möglichkeit, virtuelle geräte zu erstellen (Eingabe wie Betriebsysteme und die Gößе der Geräte einzugeben)

3.3 FeatureProvider

4 ENTWICKLUNG

4.1 Schnittstellen

Hier werden die Schnittstellen beschrieben, die mithilfe vom Restservice vom Server zur verfügung stehen.

1. Operationen auf einen Nutzer
   - Post (erstellung von einem Nutzer)
     Hier ist die Erstellung von einem Nutzer möglich. Wenn ein Nutzer unsere App. benutzen will, muss er erstmal sich registrieren lassen. Dafür braucht man:
     - Ein Username
     - Ein Passwort
     Antwort 200: Die Operation war erfolgreich
     Antwort 400: Schlechte Anfrage, falls der Nutzer ungültige oder schon vorhandene Parameter einträgt.
   - Post (Nutzer im System einloggen)
     Hier ist das Einloggen von einem Nutzer im System möglich. Diese Funktionnalität ist nur möglich, wenn der Nutzer schon registriert ist. Dafür braucht man:
     - Ein Username
     - Ein Passwort
     Antwort 200: Die Operation war erfolgreich
     Antwort 400: Schlechte Anfrage, falls der Nutzer ungültige oder schon vorhandene Parameter einträgt.
   - Delete (Meldet die aktuelle angemeldete Benutzersitzung ab) Hier ist die Abmeldung von der Sitzung möglich. Dafür braucht man den Token vom Nutzer
     Antwort 200: Die Operation war erfolgreich
     Antwort 400: Schlechte Anfrage, falls der Nutzer ungültige oder schon vorhandene Parameter einträgt.
2. Operationen auf eine Datei
   - Post (Hochladen von Dateien) Hier ist das Hochladen von einer Datei möglich. Dafür braucht man:
     - Der Token
     - Der FolderId
     - Die Datei
     Antwort 200: Die Operation war erfolgreich
     Antwort 400: Schlechte Anfrage, Im Fall die Größe der Datei größer als 30Mb ist.
     Antwort 403: Verboten
     Antwort 404: Nicht gefunden
   - Get (Herunterladen von Dateien) Hier ist das herunterladen von einer Datei möglich. Dafür braucht man:
     - Der Token
     - Der FolderId (Id vom übergeordneten Ordner)
     - Die DateiId (Id von der Datei)
     Antwort 200: Die Operation war erfolgreich
     Antwort 403: Verboten
     Antwort 404: Nicht gefunden
   - Put (Bearbeitung vom Dateiname) Hier ist die Bearbeitung von einer Datei möglich. Dafür braucht man:
     - Der Token
     - Der FolderId (Id vom übergeordneten Ordner)
     - Die DateiId (Id von der Datei)
     - Der Name von der Datei
     Antwort 200: Die Operation war erfolgreich
     Antwort 400: Schlechte Anfrage
     Antwort 403: Verboten
     Antwort 404: Nicht gefunden
   - Delete (Lschen von einer Datei) Hier ist das Löschen von einer Datei möglich. Dafür braucht man:
     - Der Token
     - Der FolderId (Id vom übergeordneten Ordner)
     - Die DateiId (Id von der Datei)
     Antwort 200: Die Operation war erfolgreich
     Antwort 403: Verboten
     Antwort 404: Nicht gefunden
3. Operationen auf ein verzeichnis
   - Get (Herunterladen von einem Ordner) Hier ist das herunterladen von einem Ordner möglich. Dafür braucht man:
     - Der Token
     - Der FolderId (Id vom Ordner)
     Antwort 200: Die Operation war erfolgreich
     Antwort 403: Verboten
     Antwort 404: Nicht gefunden
   - Post (Hochladen von einem Ordner)
     Hier ist das Hochladen von einem Ordner möglich. Dafür braucht man:
     - Der Token
     - Der FolderId
     - Der Ordner
     Antwort 200: Die Operation war erfolgreich
     Antwort 400: Schlechte Anfrage, Im Fall die Größe der Datei größer als 30Mb ist.
     Antwort 403: Verboten
     Antwort 404: Nicht gefunden
   - Put (Bearbeitung von einem Ordner)
     Hier ist die Bearbeitung von einem Ordner möglich. Dafür braucht man:
     - Den Token
     - Den FolderId (Id vom übergeordneten Ordner)

- Den Folder (z.B ein neur Name)
Antwort 200: Die Operation war erfolgreich
Antwort 400: Schlechte Anfrage
Antwort 403: Verboten
Antwort 404: Nicht gefunden

- Delete Hier ist das Löschen von einem Ordner möglich. Dafür braucht man:
- Der Token
- Der FolderId (Id vom übergeordneten Ordner)
Antwort 200: Die Operation war erfolgreich
Antwort 403: Verboten
Antwort 404: Nicht gefunden

5 ZUSAMMENFASSUNG

QUELLEN

Heitkötter, H., Majchrzak, T. A., and Kuchen, H. (2013). Md2-dsl eine domnenspezifische sprache zur beschreibung und generierung mobiler anwendungen. *Proceedings der 6. Arbeitstagung Programmiersprachen (ATPS), Software Engineering*.

Majchrzak, T. A., Ernsting, J., and Kuchen, H. (2015). Model-driven cross-platform apps: Towards business practicability. *Proceedings of the CAiSE 2015 Forum at the 27th International Conference on Advanced Information Systems Engineering (CAiSE 2015)*.