Mobile App für Foto-Überweisungen

# Versionshistorie

|  |  |  |
| --- | --- | --- |
| Datum | Kürzel | Änderung |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Inhalt

[Versionshistorie 2](#_Toc509476585)

[Einleitung 4](#_Toc509476586)

[Entwicklung Mobiler Applikationen 4](#_Toc509476587)

[Anforderungsmodellierung 4](#_Toc509476588)

[Modellierung 5](#_Toc509476589)

[Testen 5](#_Toc509476590)

[Requirements Engineering 5](#_Toc509476591)

[Modellierungsphase 5](#_Toc509476592)

[Mobile Development 6](#_Toc509476593)

[Texterkennung 6](#_Toc509476594)

[Implementierung 6](#_Toc509476595)

[Fazit und Ausblick 6](#_Toc509476596)

[Literaturverzeichnis 6](#_Toc509476597)

# Einleitung

Das Scannen von Dokumenten erfolgt meist als Bilddatei, was dazu führt, dass der digitalisierte Text weder kopiert, noch bearbeitet werden kann. Um dies zu ermöglichen kann der Text als Buchstaben erkannt und ausgegeben werden. Diese Methode der Texterkennung wird als Optical Character Recognition, kurz OCR, bezeichnet.

Moderne Banking Applikationen, beispielsweise die der Sparkasse, bieten die Möglichkeit an aus Fotos Überweisungsdaten wie IBAN, Betrag oder Verwendungszweck zu erkennen um den Überweisungsprozess zu vereinfachen.

Ziel dieses Projekts ist eine Applikation für Mobilgeräte zu entwickeln, die anhand eines Fotos Überweisungsdaten erkennt und diese auf dem Bildschirm ausgibt.

# Entwicklung Mobiler Applikationen

Der Entwicklungsprozess von Applikationen für mobile Geräte lehnt sich an die konventionelle Art des Softwareentwicklungsprozesses an, besitzt jedoch wichtige Unterschiede in beispielsweise Benutzeroberflächendesign oder der Testphase.

Der Aufbau des Projekts folgt dem V-Modell. In der Anforderungsspezifikation werden die Anforderungen des Kunden an die Software notiert. (SE Pressman, S.42)

(Bild) (SE Pressman S.43)

## Anforderungsmodellierung

Die Anforderungsmodellierung dient zur Kommunikation der Kundenanforderungen an den Softwareentwickler in Form von Text, sowie Diagrammen, sodass diese einfach zu verstehen sind. (SE Pressman, S.166)

Das V-Modell zeigt, dass es oft schwierig ist alle Anforderungen von Anfang an zu spezifizieren, weshalb durch verschiedene Testphasen die Anforderungen möglicherweise angepasst werden müssen. (SE Pressman, S.42)

Für dieses Projekt werden Szenariobasierte-, (Verhaltens-?) und Klassenmodelle verwendet, um die Anforderungen darzustellen.

Ein Beispiel für ein Szenariobasiertes Modell ist das Use-Case-Diagramm:

(unser UseCase)

Ein Beispiel für ein Verhaltensdiagramm ist das Aktivitätsdiagramm. Es soll die schon im Use-Case-Diagramm vorhandenen Informationen auf kompaktere Weise darstellen, indem es einen Interaktionsfluss zeigt. (SE Pressman, S.99f)

(unser Aktivitätsdiagramm)

Folgendes Klassendiagramm soll das Klassenmodell darstellen:

(unser Klassendiagramm)

## Modellierung

(Unsere Anforderungen)

## Testen

Das Testen einer Mobilen Applikation kann wie bei konventioneller Software durch Unit-, Integration- und Systemtests durchgeführt werden (SE Pressman, S.483). Die Unittests sollen möglichst automatisiert durchgeführt werden und Fehler in einzelnen Funktionen oder Modulen aufdecken (SE Pressman, S.473). Durch Integrationstest wird geprüft, ob einzelne Komponenten korrekt miteinander zusammenarbeiten. Beispielsweise ob die Kameraansicht der Applikation das aufgenommene Foto korrekt an das Texterkennungsmodul weiterleitet (SE Pressman, S.475f).

Da die Entwickler der Software unbewusst dazu tendieren Tests durchzuführen, die zeigen, dass die Software fehlerfrei funktioniert ist ein weiterer wichtiger Bestandteil die Applikation von Nutzern testen zu lassen und deren Feedback bezüglich Benutzerfreundlichkeit und Navigation in den Softwareentwicklungsprozess einfließen zu lassen.

Da mobile Geräte Eigenschaften aufweisen, die auf Desktop Computer nicht zutreffen, beispielsweise viele verschiedene Formfaktoren, unterschiedliche Betriebssystemversionen und limitierte Akkukapazität und Speicherplatz, ist es notwendig spezifische Tests in Bezug auf Gerätekompatibilität, Performance und Netzwerkverfügbarkeit durchzuführen. (SE Pressman, S.483)

# Requirements Engineering

Pflichtenheft, Richtlinien, Testkonzept, Funktionsspezifikation

# Modellierungsphase

Anwendungsfälle, Klassendiagram, (Aktivitäts- und Sequenzdiagramme)

# Mobile Development

Auswahl der Plattform, Besonderheiten bei mobilen Applikationen,

Designrichtlinien, Welche IDE?, Layout-Vorlagen?, Wieso Android ?

# Texterkennung

Das Ziel dieser Studienarbeit ist es eine Rechnung mit einem mobilen Endgerät zu erkennen und vorhandene Kontoinformationen auszulesen. Dafür muss ein Dokument zunächst digitalisiert vorliegen. Dies geschieht beim Mobiltelefon mithilfe einer eingebauten Kamera. Das Bild liegt als Matrix von Pixelinformationen vor, in dem für jeden Pixel ein Farbwert gespeichert ist. Für die Texterkennung wird diese Matrix weiterverarbeitet.

## Klassischer Ablauf

Es wird zunächst generell die Funktion der klassischen Texterkennung ohne Deep Learning beschrieben. Der erste Schritt für die Weiterverarbeitung ist eine Binarisierung (OCR, Chapter 2 S.5). Dafür werden zunächst aus den Farbwerten Grauwerte berechnet und im nächsten Schritt wird ein Threshhold-Algorithmus angewandt. Ganz einfach kann zum Beispiel der Wert w = 127 als Schwelle festgelegt werden. Alle Werte kleiner/gleich diesem Wert würden dann zu einer 0 und alle Werte höher als w würden zu einer 1. Mit der geschickten Wahl und Parametrisierung des Threshhold-Algorithmus wird der spätere Mustervergleich verbessert. Nach diesem Schritt können eventuell noch weitere Filterungen durchgeführt werden um Störungen wie Rauschen zu eliminieren und die Qualität der Zeichen zu verbessern. Die Binarisierung des Bildes vereinfacht auch die darauffolgende Layout-Analyse. Hier wird der semantische Inhalt des Dokuments verstanden und in Tabellen oder Felder aufgeteilt. [BILD eines Beispiel-Feldes] Es kann eine Datenbank mit Modellen genutzt werden, die Parameter für unterschiedliche Dokumenttypen speichert und die Form- und spätere Syntaxanalyse unterstützt. Die erkannten Felder werden dann einzeln auf den Inhalt untersucht. Das Feld kann zunächst anhand der Formelemente, wie zum Beispiel begrenzende Linien ausgerichtet werden. Dafür wird untersucht, ob Linien ungerade sind und das Feld wird dementsprechend gedreht. Dann werden Formelemente herausgeschnitten um für jedes Feld nur noch die Zeichen zu untersuchen. Ein Text wird dann in die einzelnen Zeichen aufgeteilt.

Für die möglichen Zeichen muss es eine Datenbank mit Mustern geben. Diese Datenbank kann nach Sprache und Zeichensatz aufgeteilt sein. Jedes Zeichen wird dann Pixel für Pixel mit den Mustern verglichen und es wird eine Wahrscheinlichkeit für jedes Muster berechnet. QUELLE Außerdem kann noch eine Syntaxanalyse durchgeführt werden. Wird zum Beispiel für ein Feld im ausgewählten Modell eine Email-Adresse verlangt, kann dies überprüft werden. Mit einer weiteren Bibliothek kann auch eine Kontextanalyse durchgeführt werden. Wenn zum Beispiel das Wort „8aum“ erkannt werden, kann daraus das Wort „Baum“ abgeleitet werden.

[Mustererkennung Beispiel]

Es fehlt:

Andere Applikation anschauen

Maschinelles Lernen

Beispiele/Bilder

# Implementierung

Was haben wir genau gemacht? Anwendung vorstellen! Bezug auf Modellierungsphase nehmen

# Fazit und Ausblick

Wo gab es Probleme ? Was hat gut geklappt, was könnte man besser machen? Was könnte in Zukunft noch umgesetzt werden?

# Literaturverzeichnis

[1] Buch, Author, Jahr – Titel (Abk.)