Zentraler Signaturdienst

Bachelorarbeit

Abteilung Informatik
Hochschule für Technik Rapperswil

Frühjahrssemester 2012

Autor(en):	Eric Aebi
Betreuer:	Prof. Dr. Andreas Steffen
Projektpartner:	Keyon AG; Jona
Experte:	Ralf Hauser, PrivaSphere AG, Zürich
Gegenleser:	Prof. Dr. Luc Bläser

0. Dokumentinformationen

0.1. Änderungsgeschichte

Datum	*Version*	*Änderung*	*Autor*
31.05.12	1.0	Dokument erstellt	ea
31.05.12	1.1	Einfügen der einzelnen Dokumente in die Kapitel Anforderungsanalyse, Sicherheitsszenarien, Schnittstellenbeschreibung, Design, Projektplan	ea
01.06.12	1.2	Einfügung GUI Analyse Einfügen Benutzeranleitung	Sr
05.06.12	1.3	Überarbeitung der Kapitel, korrektur lesen	ea
07.06.12	1.4	Überarbeitung der Kapitel	ea, sr
11.06.12	1.5	Überarbietung der Kaptiel	
13.06.12	1.6	Einfügen Systemtest	ea
14.06.12	2.0	Korrektur Lesen, Überarbeitung Kapitel Design und Installationsanleitung, Einfügen des Anhanges.	ea, sr

0.2. Inhalt

1. Aufgabenstellung

Das rechtsgültige digitale Signieren von PDF-Dokumenten nach dem Schweizer Bundesgesetz über Zertifizierungsdienste im Bereich der elektronischen Signatur (ZertES) ist heute bereits möglich. Dazu benötigt der Benutzer eine SuisseID, ein Lesegerät sowie die entsprechende Software zum Erstellen und Prüfen der digitalen Signaturen. Den meisten Anwendern ist dies jedoch zu kompliziert oder sie scheitern bereits an der Installation. In Organisationen ergibt sich ein hoher Supportaufwand bezüglich der Installation von Treibern sowie der Unterstützung der Benutzer.

Das Ziel dieser Arbeit ist es, das digitale Signieren von Daten zu vereinfachen. Bisher mussten rechtsgültige Signaturen lokal auf einem PC durchgeführt werden. Seit dem 1. August 2011 besteht die Möglichkeit, zentrale Signaturdienste innerhalb der eigenen Organisation oder bei vertrauenswürdigen Dritten sicher zu betreiben. Der Benutzer kann sich so ganz auf seinen Geschäftsprozess konzentrieren und muss sich nicht mehr um die Technik kümmern. Das Projekt sieht einen zentralen Signaturdienst vor, bei welchem die kryptografischen Schlüssel der Anwender auf einem Hardware Security Module (HSM) gespeichert werden. Somit benötigt der Enduser keine Smartcard. Die gesetzlich vorgeschriebene Zwei-Faktor-Authentisierung wird erreicht, indem sich der User zuerst bei diesem Signaturdienst einloggt. Anschliessend kann er das zu signierende File hochladen. Dieses wird ihm dann auf dem Smartphone zur Kontrolle angezeigt. Bestätigt der User dies, wird das File mit seinem Signaturschlüssel signiert.

Für das Signieren der Dokumente kann auf eine bestehende Lösung der Firma Keyon AG zurückgegriffen werden. Diese muss um das Webportal, sowie um den Versand des zu signierenden Dokumentes an ein Smartphone erweitert werden. Dabei gilt es einige konzeptuelle Probleme zu klären (beispielsweise wie die Registrierung des Benutzers resp. des Smartphone bei diesem Service gelöst werden kann). Die technischen und organisatorischen Prozesse sollen zudem sicherheitstechnisch bewertet werden.

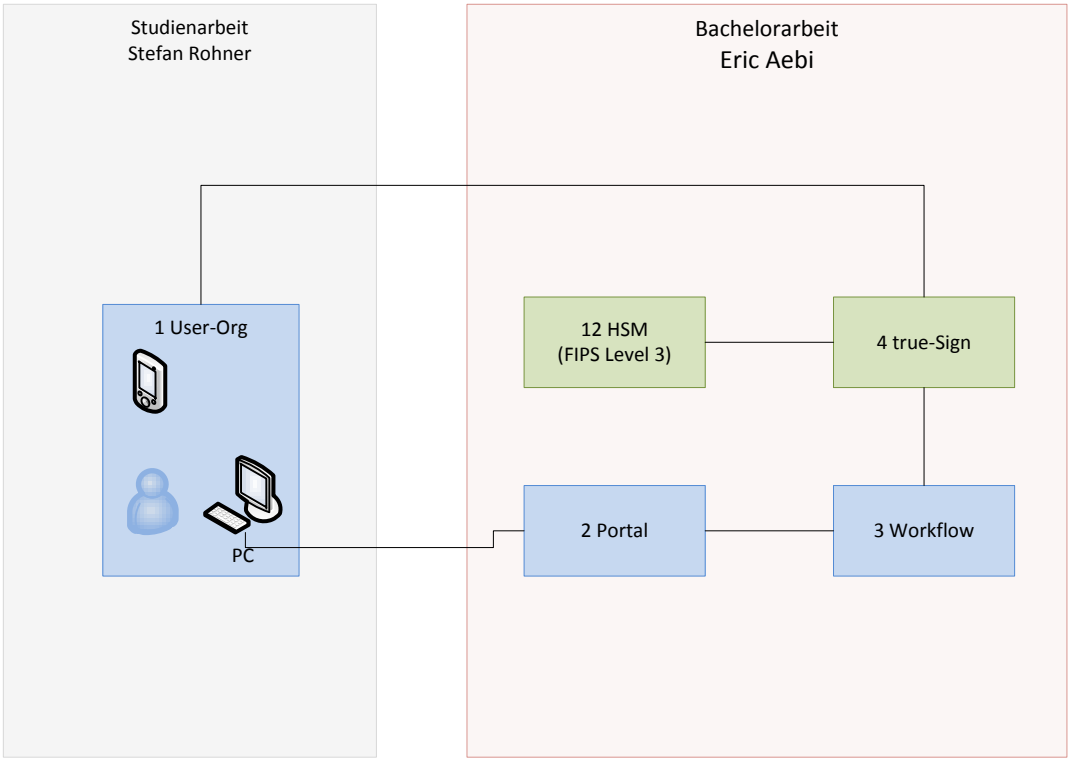


Abbildung 1 Übersicht Arbeitsteilung

1.1. Aufgaben

Folgende Punkte sollen innerhalb der Arbeiten analysiert und beschrieben sowie in einem proof of concept umgesetzt werden:

- Beschreibung der Registrierung und Verwaltung der Benutzer und deren Authentifizierungsmerkmale
- Beschreibung der Sicherheitsmerkmale der Applikation (organisatorisch und technisch)
    - Welche in- / externe Attacken sind möglich, wie werden sie verhindert
- Analyse und Umsetzung der folgenden Anwendungsfälle
    - Aufruf der Applikation mittels PC, Zwei-Kanal-Authentisierung mittels Smartphone
    - Aufruf der Applikation mittels Pad, Zwei-Kanal-Authentisierung mit demselben Gerät (optional)

2. Abstract

Die entwickelte Applikation ermöglicht das Signieren von Dokumenten via Webportal. Der User kann das zu signierende Dokument im Webportal hochladen, dieses wird ihm anschliessend zur Bestätigung an sein Smartphone geschickt, wo der User die Datei kontrollieren und die Signatur bestätigen kann. Nach erhaltener Bestätigung durch das Smartphone, signiert der Server das Dokument und stellt es im Webportal zum Download bereit.

Dank der Bestätigung via Smartphone wird die für qualifizierte Signaturen vom ZertES vorgeschriebene Zwei-Faktor-Authentisierung eingehalten. Das Login zum Webportal wurde zusätzlich durch eine TAN gesichert, welche ebenfalls in der Mobile-App angezeigt wird.

Bei der vorwiegend in Java entwickelten Server-Applikation, handelt es sich um einen SOAP Webservice, welcher die Kommunikation mit dem Webportal und der trueSign Mobile-App behandelt. Das Webportal ist eine ASP.Net-Webapplikation. Die Datenhaltung wird in einer MS SQL Express Datenbank gemacht, welche von der Webapplikation kontrolliert wird.

Die Mobile-App wird in einer ersten Version für Android entwickelt. Bei der Entwicklung wurde jedoch auf das PhoneGap-Framework gesetzt, dieses ermöglicht, die Applikation einfach auf andere mobile Betriebssysteme zu portieren. Im Speziellen kann das gesamte GUI ohne Anpassungen übernommen werden. Einzig spezifische Funktionen, wie z.B. das Empfangen von Push-Nachrichten müssen nativ für die verschiedenen Betriebssysteme entwickelt werden.

3. Erklärung von Eric Aebi und Stefan Rohner

Erklärung

Ich erkläre hiermit,

- dass ich die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt habe, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit dem Betreuer schriftlich vereinbart wurde,
- dass ich sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben habe.

Ort, Datum: Jona, 15.06.2012

Name, Unterschrift: Eric Aebi,

Name, Unterschrift: Stefan Rohner

4. Management Summary

4.1. Ausgangslage

Mit der Suisse ID ist es bereits möglich, Dokumente rechtsgültig digital zu signieren. Hierfür wird jedoch eine Smartcard benötigt. Viele User sind mit der Installation und Anwendung einer Smartcard jedoch überfordert. Die Firma Keyon AG in Jona hat bereits einen Signaturserver entwickelt, welcher nun entsprechend erweitert werden soll, um gesetzliche Vorgaben, wie die Zwei-Faktor-Authentisierung zu erfüllen.

4.2. Vorgehen, Technologien

In einer ersten Phase wurden Sicherheitsszenarien auf Grundlage des ZertES analysiert. Dabei galt es in erster Linie zu verhindern, dass ein Angreifer eine Signatur mit einem Zertifikat einer Drittperson ausführen kann. Dies konnte durch eine durchgehende Verschlüsselung mittels SSL sowie der Zwei-Faktor-Authentisierung mittels einer Smartphone-App bewerkstelligt werden.

Da bei dieser Arbeit eine bereits bestehende Applikation erweitert wurde, war die Technologie weitgehend vorgegeben. Der Serverteil wurde in Java entwickelt. Beim Web-Frontend handelt es sich um eine ASP.Net (C#) Applikation. Bei der Mobile-App handelt es sich um eine neue Entwicklung. Diese soll, um den Entwicklungsaufwand für alle mobilen Plattformen möglichst gering zu halten, mit dem PhoneGap-Framework erstellt werden. PhoneGap basiert auf HMTL5 und JavaScript und ermöglicht es so, die Applikation für alle Plattformen nur einmal zu entwickeln.

4.3. Ergebnisse

Die entwickelte Applikation ermöglicht das Signieren von Dokumenten via Webportal. Der User kann das zu signierende Dokument im Webportal hochladen, dieses wird ihm anschliessend zur Bestätigung an sein Smartphone geschickt, wo der User die Datei kontrollieren und die Signatur bestätigen kann. Nach erhaltener Bestätigung durch das Smartphone, signiert der Server das Dokument und stellt es im Webportal zum Download bereit.

Dank der Bestätigung via Smartphone wird die für qualifizierte Signaturen vom ZertES vorgeschriebene Zwei-Faktor-Authentisierung eingehalten. Das Login zum Webportal wurde zusätzlich durch eine TAN gesichert, welche ebenfalls in der Mobile-App angezeigt wird.

4.4. Ausblick

Für eine nächste Version wäre die Entwicklung der Mobile-App für weitere Plattformen lohnenswert. Zudem wird in einer produktiven Umgebung noch ein Entry-Server eingesetzt, dies würde einige Anpassungen erfordern.

5. Anforderungsspezifikation

5.1. Allgemeine Beschreibung

5.1.1. Produktperspektive

Diese Applikation soll die Möglichkeit bieten, PDF-Dateien digital rechtsgültig zu signieren. Dies jedoch ohne eine Smartcard, wie die SuisseID, besitzen zu müssen. Das Signieren soll über eine Webapplikation erfolgen. Die benötigte Zwei-Kanal-Authentisierung wird mittels einer Bestätigung der Signatur auf einem Smartphone erreicht.

5.1.2. Produktfunktion

Die zu signierende PDF-Datei wird vom Rechner des Kunden zur Webapplikation hochgeladen. Anschliessend wird es an das Smartphone des Kunden geschickt. Bestätigt dieser auf dem Smartphone die Korrektheit der Datei, wird diese signiert und dem Kunden zum Download zur Verfügung gestellt.

5.1.3. Benutzercharakteristik

Personen, welche ihre Geschäfte online tätigen möchten und dafür digital signierte Dateien benötigen, allerdings über zu wenig IT-Wissen verfügen, um eine Smartcard für die Signatur zu verwenden oder bereits an der Installation des Smartcard-Lesers scheitern.

5.1.4. Einschränkungen

Für die volle Funktionalität ist ein modernes Mobilgerät mit einer Internetverbindung nötig.

5.1.5. Abhängigkeiten

Für die Signatur der PDF-Dateien wird die bereits bestehende Lösung der Firma Keyon AG verwendet. Diese muss jedoch um ein Webportal erweitert werden.

5.2. Spezifische Anforderungen

Das Projekt besteht aus zwei Applikationen. Einerseits gibt es die Webapplikation zur Signierung der PDF-Dateien und andererseits die Mobile-App zur Bestätigung der Korrektheit der Datei. Die folgenden genannten Anforderungen gelten für beide Teile des Projektes, ausser es ist anders erwähnt.

5.2.1. Nichtfunktionale Anforderungen

5.2.1.1. Angemessenheit

Die Mobile-Applikation soll möglichst wenig Speicherplatz benötigen. Der Ladevorgang soll möglichst schnell ablaufen, dies ist allerdings zu einem grossen Teil von der Grösse der zu signierenden Datei und der Geschwindigkeit der Internetverbindung abhängig.

5.2.2. Bedienbarkeit

5.2.2.1. Verständlichkeit

Das GUI soll ohne jeglichen Lernaufwand und absolut intuitiv bedienbar sein.

5.2.2.2. Bedienbarkeit

Das GUI der Mobile-Applikation soll vollständig über den Touchscreen des mobilen Geräts bedienbar sein.

5.2.3. Zuverlässigkeit

5.2.3.1. Reife

In 99% soll ein Dokument, das via trueSign Webportal hochgeladen wurde, auch in der trueSign Mobile-App angezeigt werden. Ausgenommen sind Fehler bei der Übertragung der Push-Notification, da diese in der Hoheit des Anbieters liegen und eine Auslieferung vom Service Provider nicht garantiert wird.

Zu 99% soll ein von der trueSign Mobile-App gesendete Signatur-Request vom trueSign Service erfolgreich bearbeitet werden.

5.2.3.2. Fehlertoleranz

Besteht keine oder nur eine schlechte Verbindung zur Mobile-Applikation, wird der Benutzer darüber informiert und er kann einen erneuten Versuch starten. Ist jedoch keine Bestätigung der Datei mittels Smartphone möglich, kann die Signatur nicht erstellt werden, da in diesem Fall die gesetzlich vorgeschriebene Zwei-Kanal-Authentisierung nicht erreicht werden kann.

5.2.4. Leistung

5.2.4.1. Antwortzeit

Die Mobile-Applikation soll beim Start nach maximal zwei Sekunden vollständig geladen sein. Ausgenommen davon ist das Herunterladen der zu signierenden PDF-Datei. Ausserdem soll eine Sanduhr oder eine Statusanzeige den Benutzer darüber informieren, dass der Download im Gange ist bzw. wie lange der Prozess noch dauern wird. Das GUI sollte möglichst ohne Verzögerung, aber nach maximal 150 Millisekunden auf eine Berührung des Touchscreens reagieren.

Das GUI der Webapplikation soll ebenfalls über eine Statusanzeige verfügen, um dem User anzuzeigen, wie lange der Upload der zu signierenden PDF Datei noch dauern wird.

5.2.5. Wartbarkeit

5.2.5.1. Anpassbarkeit

Die Applikation muss ohne Probleme erweitert werden können.

5.2.5.2. Stabilität

In 95% soll der gesamte Prozess der Signatur ohne Fehler ablaufen. Ausgenommen hiervon sind Fehler in der Kommunikation, welche nicht unter der Kontrolle der Applikation liegen.

5.2.6. Schnittstellen

5.2.6.1. Benutzerschnittstelle

- Das GUI der trueSign Mobile-App soll zu 100% über den Touchscreen des Mobiltelefons bedienbar sein.
- Das GUI des trueSign Webportals soll nach aktuellen Webdesign-Standards designt werden und intuitiv bedienbar sein.

5.2.6.2. Softwareschnittstellen

- PhoneGap 1.6.0 – Zur nativen Entwicklung von Mobile-Apps für mehrere Plattformen
- Google API Level 15 (Android 4.0) – spezifische Anpassungen an Android
- iOS 5.1 – spezifische Anpassungen an iOS.

5.2.6.3. Kommunikationsschnittstelle

Es gibt in diesem Projekt drei Kommunikationsschnittstellen, welche im Kapitel 9. Schnittstellenbeschreibung erläutert werden.

5.2.7. Lizenzanforderungen

In der Entwicklungsphase werden keine Lizenzen benötigt, da alle benutzten IDEs und Libraries opensource bzw. lizenzfrei sind. Für die spätere Veröffentlichung in den jeweiligen App Stores werden Developer-Lizenzen benötigt.

5.2.7.1. Verwendete Standards

- UML 2.0
- HTML5 / CSS / JS
- SOAP

5.3. Use Cases

5.3.1. Use Case Diagramm

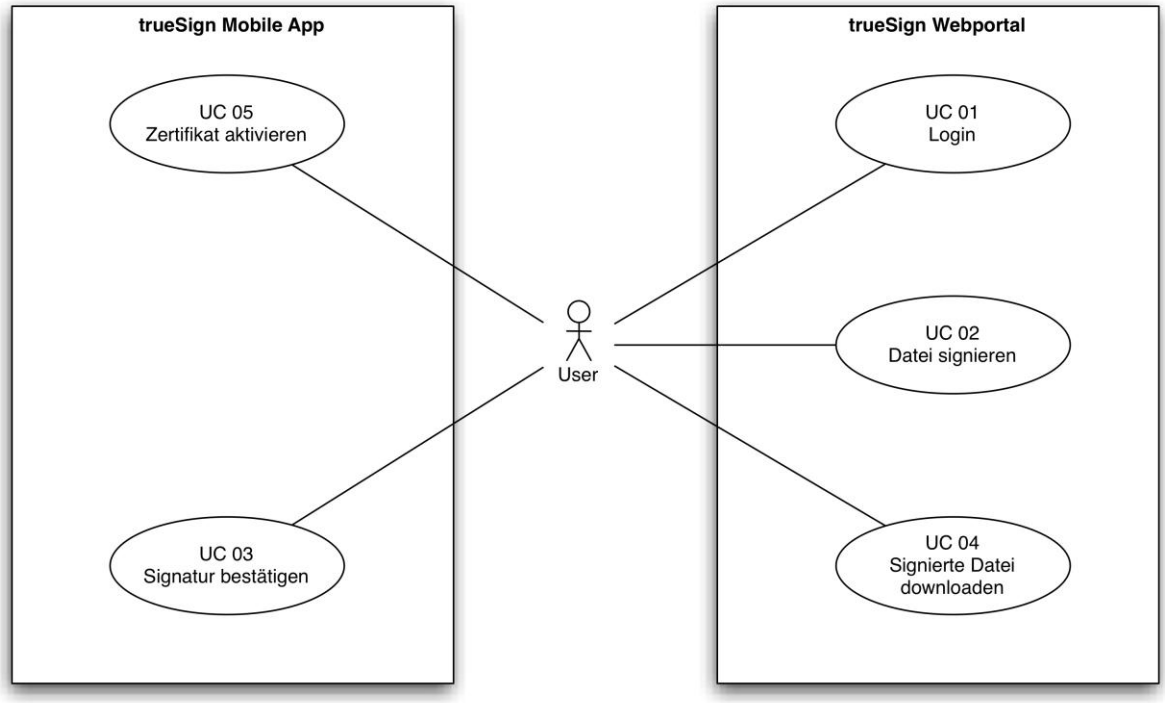


Abbildung 2 Use Cases Diagramm

5.3.2. Aktoren & Stakeholders

Der primäre Akteur ist der Benutzer der Applikation. Er bedient die Applikation über das Webportal sowie über den Touchscreen des mobilen Gerätes.

5.3.3. Use Cases brief

5.3.3.1. UC 01: Login

Ruft der Benutzer die Applikation über das Webportal auf, muss er sich als erstes authentifizieren, um die Applikation benutzen zu können

5.3.3.2. UC 02: Datei signieren

Dies ist der Main-Use-Case. Der Benutzer kann die PDF-Datei, welche er signieren möchte, hochladen und den Signierungsprozess starten.

5.3.3.3. UC 03: Signatur bestätigen

Die PDF-Datei wird auf dem Smartphone des Benutzers angezeigt. Der User muss die Signatur bestätigen.

5.3.3.4. UC 04: Signierte Datei downloaden

Wurde der Signaturprozess erfolgreich abgeschlossen, kann der Benutzer die signierte PDF-Datei vom Webportal herunterladen.

5.3.3.5. UC 05 Zertifikat aktivieren

Beim erstmaligen Anmelden am trueSign Webportal, muss das Zertifikat via trueSign Mobile-App aktiviert werden.

5.3.4. UC 01 Login

Use-Case-Abschnitt	Kommentar
Scope	trueSign Webportal
Level	Anwenderziel
Primary Actor	Benutzer
Stakeholders & Interests	Benutzer: sich beim trueSign Webportal authentifizieren. Dies auf einem einfachen, jedoch sicheren Weg.
Preconditions	Der Benutzer hat sich bereits einmalig für den Service registriert
Success Guarantee	Der Benutzer wurde authentifiziert
Main success scenario	1. Aufruf des trueSign Webportals 2. Benutzer gibt seine Credentials ein 3. TAN wird auf sein Smartphone geschickt 4. Mit Eingabe der TAN im Webportal authentifiziert sich der Benutzer 5. Hauptmenü des Webportals wird angezeigt
Postconditions	Der Benutzer kann die Funktionen des Webportal benutzen
Extensions	1. Benutzer gibt die TAN nicht innerhalb von 2 min. ein 2. Die Gültigkeit der TAN verfällt 3. Der Zugriff auf die Applikation wird verwehrt
Special Requirements	-
Technology and Data Variations list	-
Frequency of occurence	1-mal pro Applikationsausführung

5.3.5. UC 02 Datei signieren

Use-Case-Abschnitt	Kommentar
Scope	trueSign Webportal
Level	Anwenderziel
Primary Actor	Benutzer
Stakeholders & Interests	Benutzer: Die zu signierende PDF-Datei kann hochgeladen - und nach der Bestätigung - signiert wieder heruntergeladen werden.
Preconditions	Der Benutzer hat sich erfolgreich für das Webportal authentifiziert.
Success Guarantee	Die PDF-Datei kann signiert heruntergeladen werden.
Postconditions	Der Benutzer kann über einen Link die signierte PDF-Datei herunterladen.
Main success scenario	1. Der Benutzer lädt die PDF-Datei im Webportal hoch 2. Der Benutzer bestätigt die Signatur -> UC 03 3. Die PDF-Datei wird signiert 4. Der Benutzer lädt die signierte Datei herunter.
Extensions	Wird die Bestätigung auf dem Smartphone nicht innerhalb von 5 min. ausgeführt, wird die Signatur der Datei nicht erstellt.
Special Requirements	-
Technology and Data Variations list	-
Frequency of occurence	Ca. 1-mal pro Applikationsausführung.
Notes	

5.3.6. UC 03 Signatur bestätigen

Use-Case-Abschnitt	Kommentar
Scope	trueSign Mobile-App
Level	Anwenderziel
Primary Actor	Benutzer
Stakeholders & Interests	Benutzer: Um die gesetzlich vorgeschriebene Zwei-Faktor-Authentisierung zu erreichen, muss der Benutzer die Signatur mit dem Smartphone bestätigen.
Preconditions	Die zu signierende PDF-Datei wurde im trueSign Webportal hochgeladen.
Success Guarantee	Die PDF-Datei wird bestätigt und die Signatur kann erstellt werden.
Postconditions	Das trueSign Webportal erhält die Bestätigung und erstellt die Signatur
Main success scenario	1. Der Benutzer erhält auf dem Smartphone eine Nachricht. 2. Der Benutzer kontrolliert die Echtheit der PDF-Datei auf dem Smartphone. 3. Der Benutzer gibt zur Bestätigung der Signatur die PIN für seinen Private Key ein. 4. Die Bestätigung, inkl. PIN, wird ans trueSign Webportal geschickt
Extensions	3a) Der Benutzer bestätigt die Echtheit der Datei nicht. Trifft die Bestätigung nicht innerhalb von 5 min. beim Webportal ein, wird die PDF-Datei verworfen.
Special Requirements	-
Technology and Data Variations list	-
Frequency of occurence	ca. 1-5-mal pro Applikationsausführung
Notes	

5.3.7. UC 04 Signierte Datei downloaden

Use-Case-Abschnitt	Kommentar
Scope	trueSign Webportal
Level	Anwenderziel
Primary Actor	Benutzer
Stakeholders & Interests	Benutzer: Der Benutzer kann die signierte Datei herunterladen.
Preconditions	Die Bestätigung der Signatur konnte korrekt ausgeführt werden.
Success Guarantee	Der Benutzer hat die signierte Datei auf seinem Rechner.
Postconditions	Der Signierungsprozess wurde erfolgreich abgeschlossen.
Main success scenario	1. Der Benutzer lädt die Datei herunter
Extensions	keine Extensions oder Alternative Flows
Special Requirements	-
Technology and Data Variations list	-
Frequency of occurence	ca. 1-mal pro Applikationsausführung

5.3.8. UC 05 Zertifikat aktivieren

Use-Case-Abschnitt	Kommentar
Scope	trueSign Webportal / trueSign Mobile-App
Level	Anwenderziel
Primary Actor	Benutzer
Stakeholders & Interests	Benutzer: Der Benutzer muss sein Zertifikat nach dem ersten Login beim trueSign Webportal aktivieren.
Preconditions	Der Benutzer hat sich noch nie beim trueSign Webportal angemeldet.
Success Guarantee	Der Benutzer kann mit seinem Zertifikat Dateien signieren.
Postconditions	Das Zertifikat wurde aktiviert.
Main success scenario	1. Der Benutzer meldet sich an 2. Der Benutzer erhält eine Nachricht auf dem Smartphone zur Aktivierung seines Zertifikates. 3. Der Benutzer gibt seine Initialisierungs-PIN im Smartphone ein 4. Die PIN wird an den trueSign Service geschickt 5. Das Zertifikat wird aktiviert
Extensions	3a) Gibt der Benutzer eine falsche PIN ein, wird er zur erneuten Eingabe aufgefordert. 3b) Nach vier Fehlversuchen wird das Zertifikat revoziert.
Special Requirements	-
Technology and Data Variations list	-
Frequency of occurence	ca. 1-mal pro Applikationsausführung

6. Analyse der Sicherheitsszenarien

6.1. Übersicht über die Applikation

Die Applikation besteht aus zwei Teilen, aus dem trueSign Webportal und der trueSign Mobile-App. Das Signieren eines PDF Dokumentes soll über das Webportal erfolgen. Um die Zwei-Faktor-Authentisierung zu ermöglichen, wird die Mobile-App eingesetzt. Das Smartphone ersetzt somit gewissermassen das in anderen Signaturanwendungen übliche Hardware-Token.

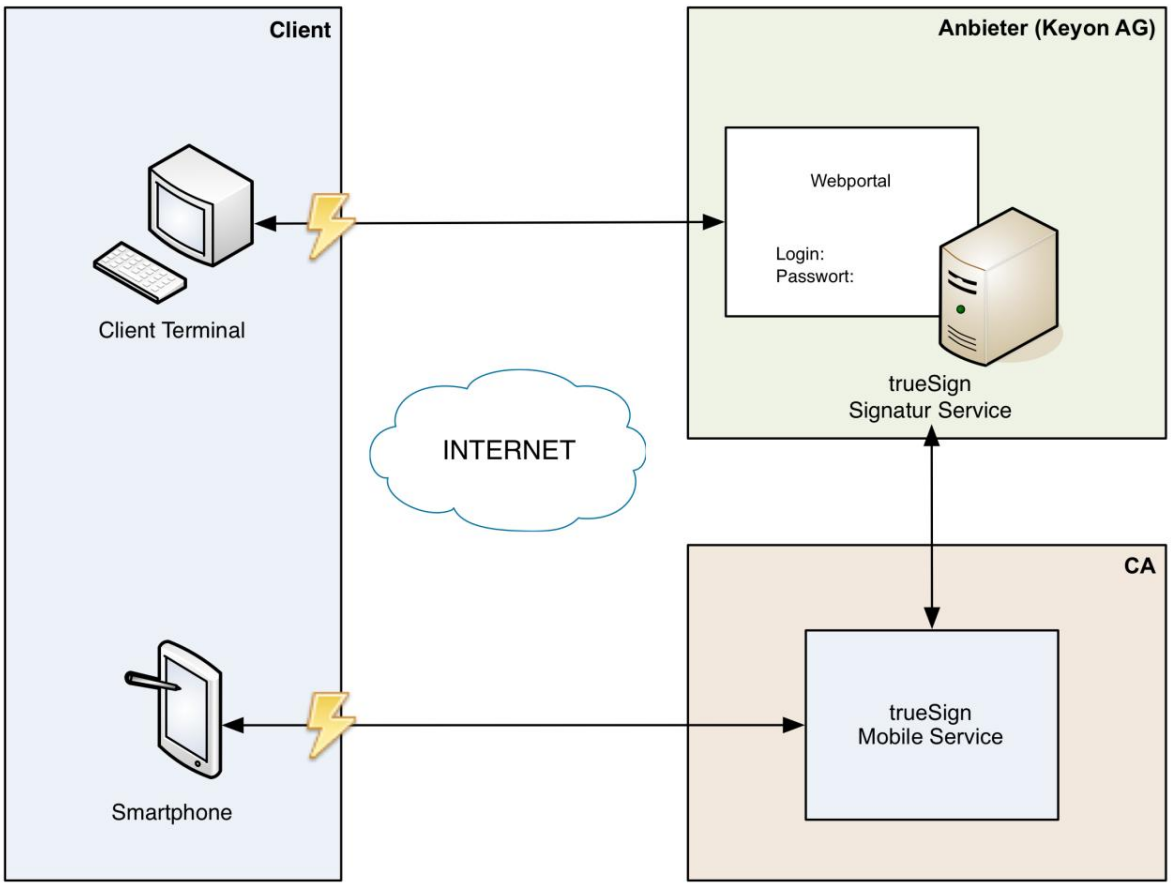


Abbildung 3 Übersicht über die Applikation

6.2. Szenarien

Um zu klären, welche Sicherheitsmechanismen in welche Teile der Applikation eingebaut werden müssen, sollen im Folgenden mögliche Angriffsszenarien untersucht werden. Dabei gilt es zu verhindern, dass eine Drittperson eine Datei im Namen des Users signieren kann.

6.2.1. Szenario: Malware auf Rechner

6.2.1.1. Beschreibung

Der Desktop-Rechner, von welchem auf das trueSign Webportal zugegriffen wird, ist mit Malware verseucht. Somit wird der gesamte Netzwerkverkehr des Rechners mitgelesen und könnte auch verändert werden.

6.2.1.2. Angriff

Ein Angreifer könnte mittels der Malware die Daten, welche an das trueSign Webportal geschickt werden, abfangen und sich somit die Login-Daten des Users beschaffen. Anschliessend könnte er sich mit den so gewonnen Credentials einloggen und eine PDF-Datei signieren lassen.

6.2.1.3. Abwehrmechanismus

Jede Signatur muss via trueSign Mobile-App bestätigt werden. Diese Bestätigungsanfrage wird vom trueSign Service abgeschickt, welcher bei einer Certificate Authority steht. Es ist somit für die Malware

auf dem Rechner nicht möglich, diese Bestätigungsanfrage automatisch abzufangen und zu bestätigen oder abzuändern. Als Bestätigung muss die PIN für den Private Key auf dem Smartphone eingegeben werden. Die PIN wird durch einen gesicherten Kanal direkt an den trueSign Mobile Service geschickt. Somit ist auch die Verwendung des Private Keys durch Drittpersonen (z.B. die Betreiber des trueSign Webportals) nicht möglich. Der Benutzer kann auf seinem Smartphone die Signatur ablehnen.

6.2.1.4. Zusätzliche Sicherheitsaspekte

Das Login beim trueSign Webportal soll zusätzlich durch eine Mobile-TAN gesichert werden. Dies um zu verhindern, dass sich ein Angreifer, der sich via Malware die Login-Credentials beschafft hat, beim Webportal einloggen kann. Denn innerhalb des Webportals können wichtige Daten eingesehen und auch verändert werden.

Es wäre auch möglich, das Login durch eine Client-Authentifizierung mittels SSL zu schützen. Die Sicherheit würde verglichen mit einem Login mit Mobile-TAN jedoch nur unwesentlich erhöht werden. Allerdings würde dann jeder Benutzer ein Clientzertifikat benötigen, was der Idee, die digitale Signatur zu vereinfachen, widersprechen würde. Deshalb wird von einer SSL Client-Authentifizierung abgesehen.

6.2.2. Szenario: Verlust des Smartphone

6.2.2.1. Unberechtigtes Lesen der Bestätigungsanfrage

Beschreibung

Der Benutzer gibt eine Signatur im trueSign Webportal in Auftrag. Das Smartphone, über welches diese bestätigt wird, ist nicht mit einem PIN-Code gesichert.

Angriff

Der Angreifer sieht die Push-Nachricht auf dem Smartphone und liest als unberechtigte Person die PDF-Datei, welche zur Bestätigung an das Smartphone geschickt wurde.

Abwehrmechanismus

Die trueSign Mobile-App könnte den Benutzer vor der Anzeige des PDFs auffordern eine im trueSign Webportal angezeigte TAN einzugeben. Der Angreifer könnte die PDF-Datei somit nicht einsehen. Es ist jedoch davon auszugehen, dass ein Benutzer, welcher eine vertrauliche Datei signieren möchte, darauf achten wird, dass niemand Einblick auf sein Mobilgerät hat. Deshalb wird zugunsten der Usability davon abgesehen, den Prozess durch eine weitere Mobile-TAN zu schützen.

6.2.2.2. Unberechtigtes Signieren einer Datei

Beschreibung

Der Benutzer verliert das für die Bestätigung der Signatur eingetragene Smartphone.

Angriff

Der böswillige Finder des Smartphone kann sich im trueSign Webportal einloggen, eine Signatur auslösen und diese dann mit dem Smartphone bestätigen. Zudem kann er möglicherweise Daten einsehen, welche der Benutzer vor dem Verlust des Smartphone zur Signatur aufgegeben hat.

Abwehrmechanismus

Das trueSign Webportal wird durch ein Login geschützt. Somit kann der Finder des Smartphone keine Signatur in Auftrag geben. Das Passwort hierfür kann vom Benutzer gewählt werden. Es gibt aber gewisse Sicherheitskriterien, die erfüllt werden müssen. Die zur Bestätigung an das Smartphone geschickten Daten werden nicht auf dem Smartphone gespeichert und somit können keine zuvor signierten Dateien eingesehen werden.

6.2.2.3. Zusätzliche Sicherheitsaspekte

Die trueSign Mobile-App könnte durch ein Login (Username / Passwort) gesichert werden. Jedoch bietet die Mobile-App keine Angriffsfläche. Die App zeigt keine Daten an, sofern aktuell kein Signaturauftrag vorliegt. Da die App keine Daten speichert, ist eine Verschlüsselung der Daten ebenfalls nicht nötig.

6.2.3. Szenario: Angriff auf das LAN / WLAN des Benutzers

6.2.3.1. Beschreibung

Der Benutzer verwendet die trueSign Applikation in einem öffentlichen WLAN oder in seinem privaten, schlecht gesicherten Netzwerk.

6.2.3.2. Angriff

Der Angreifer verbindet sich mit dem Netzwerk (häufig WLAN) des Users und kann so die Daten, die an das trueSign Webportal geschickten werden, mitlesen (Man-In-The-Middle). Falls das Smartphone ebenfalls mit dem Netzwerk verbunden ist, wäre es dem Angreifer auch möglich den Verkehr zwischen dem Smartphone und trueSign auszulesen.

6.2.3.3. Abwehrmechanismus

Der Netzwerkverkehr zwischen allen Komponenten wird mittels SSL verschlüsselt. Somit ist es dem Angreifer nicht mehr möglich, den Verkehr zu lesen. Ebenso wenig kann er die Daten unbemerkt verändern. Wichtig dabei ist nicht nur die Verschlüsselung der Daten, sondern auch die Überprüfung des Serverzertifikates. Dies um zu verhindern, dass die PIN für den Private Key, der zur Bestätigung der Signatur eingegeben wird, an eine Drittperson gelangen kann.

6.3. Timeouts bei der Kommunikation

Bei einer mobilen Kommunikation muss immer damit gerechnet werden, dass das mobile Gerät nicht erreichbar ist. Zudem ist es bei einem Webportal gut möglich, dass sich der Benutzer nicht mehr ausloggt. Deshalb sollen verschiedene Timeouts definiert werden.

6.3.1.1. Timeout Bestätigung der Signatur

Wird ein zur Signatur hochgeladenes PDF nicht spätestens nach 5 Min. vom mobilen Gerät abgeholt, verwirft der trueSign Service die Signatur-Anfrage automatisch und teilt dies dem trueSign Webportal mit.

Nachdem das PDF vom Smartphone abgeholt wurde, muss es innerhalb von 10 min. bestätigt werden. Ansonsten wird wiederum die Signatur-Anfrage vom trueSign Service verworfen.

6.3.1.2. Timeout trueSign Webportal

Der Benutzer soll sich während eines Signaturprozesses nicht mehrfach einloggen müssen. Deshalb setzt sich das Timeout des trueSign Webportals aus den beiden Timeouts für die Bestätigung der Signatur zusammen. Somit ergibt sich für das Webportal ein Timeout von 20 min. Nach deren Ablauf ohne eine Aktivität seitens des Benutzers wird er automatisch abgemeldet.

6.4. Registrierung

Um ein Dokument rechtsgültig signieren zu können, muss sich die Person bei der Registrierung einmalig ausweisen. Als Registrierungsprozess wird Folgendes vorgesehen:

6.4.1. Online Anmeldung

Der Benutzer kann sich auf dem trueSign Webportal registrieren. Dabei muss er folgende Angaben machen:

- Email-Adresse
- Vor- und Nachname
- Adresse
- Wohnort
- Mobiltelefonnummer
- Passwort für das trueSign Webportal

Die Email-Adresse wird als Benutzername verwendet. Das Passwort muss bestimmten Mindestanforderungen gerecht werden. Diese können im trueSign Webportal konfiguriert werden. Standard ist eine Mindestlänge von acht Zeichen sowie Gross- und Kleinbuchstaben.

6.4.2. Überprüfung der Identität

Aufgrund der Registrierung wird dem Benutzer eine Bestätigung an die angegebene Adresse gesendet. Der Benutzer muss die darin enthaltenen Angaben kontrollieren und mit seiner Unterschrift

verifizieren. Er ist verpflichtet diese schriftliche Bestätigung anschliessend bei der Registration Authority (RA) mit einem gültigen Ausweis vorzuzeigen. Ein persönliches Erscheinen ist gemäss dem Gesetz[1] vorgeschrieben und kann nicht umgangen werden. Nach dieser Bestätigung wird von der RA der Zertifikatsrequest für den Benutzer erstellt und der Account wird eröffnet.

6.4.3. Aktivierung der trueSign Mobile-App

Der Benutzer erhält einen Initialisierungs-PIN für die trueSign Mobile-App per Post. Beim ersten Start der App muss er seinen Benutzernamen sowie die PIN eingeben. Die trueSign Mobile-App schickt anschliessend diese Daten und zusätzlich die IMEI des Smartphone an das trueSign Webportal. Das mobile Gerät wird fortan mittels der IMEI identifiziert. Die Registration-ID, welche vom Push Notification Provider ausgegeben wird, ist ebenfalls der IMEI zugeordnet.

6.4.3.1. Neuinstallation der trueSign Mobile-App

Falls der Benutzer die trueSign Mobile-App auf demselben Gerät erneut installiert, muss keine neue Initialisierung durchgeführt werden, da die IMEI des Gerätes bereits beim trueSign Webportal gespeichert ist. Wechselt der Benutzer hingegen sein Smartphone durch ein neues Gerät aus, muss er die per Post erhaltene Initialisierungs-PIN erneut eingeben. Aus diesem Grund, muss der Benutzer diese PIN an einem sicheren Ort aufbewahren.

6.4.4. Aktivierung des Signaturschlüssels

Der Benutzer erhält die Initialisierungs-PIN für seinen Private Key per Post. Bei der ersten Anmeldung muss er diesen über sein Smartphone ändern.

6.4.4.1. Inkorrekte Aktivierungsversuche

Nach vier inkorrekten Aktivierungsversuchen wird der Gebrauch des Signaturschlüssels gesperrt. Dies ist gesetzlich vorgeschrieben2. Die Anzahl der möglichen Fehlversuche kann je nach Länge der Aktivierungs-PIN erhöht werden. Ist der Signaturschlüssel gesperrt, ist eine erneute Überprüfung der Identität gemäss 6.4.2 nötig.

6.5. Ungültigkeitserklärung des Zertifikates

Zertifikate müssen aus folgenden Gründen unverzüglich für ungültig erklärt werden[3]:

- Der Inhaber stellt einen entsprechenden Antrag,
- es stellt sich heraus, dass das Zertifikat unrechtmässig erlangt wurde,
- das Zertifikat bietet keine Gewähr mehr für die Zuordnung des Signaturschlüssels zu einer bestimmten Person.

Nach einer Ungültigkeitserklärung eines Zertifikates muss der Inhaber unmittelbar informiert werden. Einerseits mittels einer Email an die bei der Registrierung angegebenen Email-Adresse, andererseits soll der Benutzer auch schriftlich per Post informiert werden.

6.5.1. Ungültigkeitserklärungsantrag vom Benutzer

Da der Benutzer einen Ungültigkeitserklärungsantrag – sollte er das mobile Gerät verloren haben – auch ohne die trueSign Mobile-App erstellen können muss, wird ihm nach der Registrierung eine Revokations-PIN per Post zugestellt. Mit dieser und der Angabe seiner Benutzerdaten kann er den Antrag erstellen. Die PIN ist notwendig um zu verhindern, dass eine unberechtigte Person das Zertifikat sperren lässt.[4]

6.6. Änderung der Benutzerdaten

Da die meisten Daten sicherheitsrelevant sind, können diese nicht im Online-Portal geändert werden.

[1] ZertES Art. 8 Abs. 1

[2] TAV über Zertifizierungsdienste im Bereich der elektronischen Signatur, Kapitel 3.3.3a [3]

[3] ZertES Art. 10 Abs. 1

[4] ZertES Art. 10 Abs. 2

6.6.1. Mobiltelefonnummer

Die Mobiltelefonnummer wird nur für allfällige Rückfragen gespeichert. Die Bestätigung der Signatur läuft mittels der trueSign Mobile-App. Zur Identifikation des mobilen Geräts wird die IMEI genutzt. Somit kann die Telefonnummer über das trueSign Webportal jederzeit geändert werden.

6.6.2. Name des Benutzers

Sollte sich der Name des Benutzers beispielsweise aufgrund einer Heirat ändern, muss ein neues Zertifikat erstellt werden. Dies muss nach der in 6.4.2 beschriebenen Prozedur durchgeführt werden.

6.6.3. Adresse des Benutzers

Die Adresse des Benutzers ist nicht sicherheitskritisch und kann deshalb im Webportal geändert werden. Die Adressänderung wird sofort per Email bestätigt und kann via Link in der Email-Nachricht rückgängig gemacht werden.

6.6.3.1. Angriffsmöglichkeiten

Ein Angreifer kann nach einem Login beim trueSign Webportal die Anschrift des Benutzers ändern, um so Informationen zum Zertifikat per Post zu erhalten. Jedoch wird der rechtmässige Benutzer via Email auf die Adressänderung aufmerksam gemacht und kann diese unverzüglich wieder rückgängig machen. Der Schaden kann somit sehr gut begrenzt werden. Und da eine Adressänderung relativ häufig vorkommt, soll hier von weiteren Sicherheitsmechanismen abgesehen werden.

6.6.4. Passwort

Das Passwort für das trueSign Webportal soll ebenfalls auf einfache Weise geändert werden können. Zudem soll es eine Passwortrecovery-Funktion geben, sollte der Benutzer das Passwort vergessen haben.

6.6.4.1. Angriffsmöglichkeiten

Sollte sich ein Angreifer das Passwort mittels Malware oder auf anderen Wegen beschafft haben, könnte er versuchen, sich im trueSign Webportal einzuloggen und das Passwort zu ändern. Da der Login-Prozess jedoch zusätzlich über eine Mobile-TAN gesichert ist, wird dem Angreifer dies nicht gelingen.

6.6.5. Email-Adresse

Da die Email-Adresse eine gewisse Kontrollfunktion hat und über diese viele wichtige Informationen an den Benutzer gesendet werden, kann sie nicht über das Webportal geändert werden. Ein Angreifer könnte sonst seine Adresse eintragen, ohne dass es der Inhaber des Accounts merken würde.

6.6.5.1. Angriffsmöglichkeiten

Könnte ein Angreifer die Email-Adresse ändern und seine eigene eintragen, würde der rechtmässige Benutzer keine Informationen mehr über den Zustand seines Zertifikates erhalten. Ebenso wenig würde die Passwortrecovery-Funktion korrekt funktionieren, da diese auf der Email-Adresse beruht. Deshalb gilt es, eine Änderung der Email-Adresse zu verhindern.

6.7. Fazit

6.7.1. Sicherheitsmechanismen des trueSign Webportals

Das Webportal wird durch ein Login geschützt, bei welchem der User das Passwort selbst wählen kann. Es muss allerdings eine minimale Sicherheit bieten. Zusätzlich wird das Login mit einer Mobile-TAN gesichert. Von einer stärkeren Authentifizierung wie z.B. SSL Client-Authentifizierung wird abgesehen, da mit dem Webportal allein keine Daten signiert werden können.

Der Datenverkehr zwischen dem Webportal und dem Rechner des Users, sowie dem Smartphone wird mittels SSL verschlüsselt, damit das Abhören der übertragenen Daten verhindert wird und um sicherzustellen, dass die trueSign Mobile-App wirklich mit einem trueSign Service Server verbunden ist.

Um eine unrechtmässige Signatur abzuwenden, können nur einzelne Daten im trueSign Webportal geändert werden. Für eine Änderung von sicherheitskritischen Daten, muss der Benutzer persönlich bei der RA erscheinen und sich ausweisen.

6.7.2. Sicherheitsmechanismen der trueSign Mobile-App

Die trueSign Mobile-App speichert keine der zur Bestätigung empfangenen Daten auf dem Smartphone. Somit ist es nicht möglich Daten einzusehen. Eine Sicherung der trueSign Mobile-App mittels eines Passwortes ist nicht nötig.

6.7.3. Allgemeine Sicherheitsmechanismen

Die Registrierung beim trueSign Signaturservice hat nach einer vorgegebenen und zum Teil auch gesetzlich vorgeschriebenen Prozedur zu erfolgen. Dadurch kann ein Betrug bei der Registrierung praktisch ausgeschlossen werden.

Ein Benutzer kann für sein Zertifikat jederzeit einen Ungültigkeitserklärungsantrag ausstellen.

7. Analyse der Mobilekommunikation

In diesem Kapitel soll die Kommunikation zwischen der trueSign Mobile-App und dem trueSign Mobile Service beschrieben werden. Sowie die in der Entwicklung der Mobile-App verwendeten Frameworks und IDEs beschreiben werden.

7.1.1. Datenverbindungen zum mobilen Gerät

Es wird angenommen, dass sich der Applikationsbenutzer im optimalen Fall über ein Wireless-LAN verbunden ist, welches eine möglichst schnelle Übertragungsgeschwindigkeit zulässt. Ansonsten gehen wir davon aus, dass er sich in einer grösseren Stadt befindet, welche eine möglichst optimale Datenverbindung ermöglicht.

Die Datenübertragung ist auch über die Mobilfunknetze möglich, allerdings ist hier je nach Verbindungsqualität mit Einschränkungen zu Rechnen. Zudem können für den Benutzer je nach Mobilfunkt-Abonnement auch noch zusatzkosten entstehen. Daher ist, speziell bei grösseren Dateien, darauf zu achten, dass eine Wireless-LAN Verbindung zur Verfügung steht.

Dies mag den meisten Mobilen-Applikationen wiedersprechen, da die Mobilität eingeschränkt wird. In unserem Fall wird die Mobile-App aber nur als Sicherheitstoken gebraucht und es gibt keine Use Cases für die alleinige Nutzung der App, sondern nur in Kombination mit dem trueSign Webportal. Daher wird dies den Benutzer der App nicht einschränken.

7.2. Aufbau Mobile-App

7.2.1. Entwicklungsumgebung

7.2.1.1. Software Development Kit (SDK)

Damit Software für Smartphones entwickelt werden kann, werden Werkzeuge und Anwendungen für die spezifischen Mobile-Betriebssystemen verwendet (SDK für Android und iOS).

7.2.1.2. PhoneGap mit Dreamweaver und JQuery

PhoneGap 1.5: PhoneGap ist ein Open-Source-Framework, welches die Entwicklung von Apps für Smartphones auf Basis von HTML5, JavaScript und CSS erlaubt - mit nativer Unterstützung der sieben grössten Plattformen. Somit schließt sich eine Lücke zwischen nativer Entwicklung und Entwicklung mit HTML5, da der Zugriff auf tiefer im System verankerte Elemente wie Sensoren, Systemdaten oder Benutzerdaten bislang für WebApps, die letztendlich im Browser des Geräts laufen mussten, unmöglich war. PhoneGap ermöglicht so auch eine breite Verfügbarkeit einer App auf zahlreichen Plattformen: Die eigentliche App wird einmal geschrieben und kann dank Wrapper nativ für iOS, Android, Windows Phone, Blackberry, Symbian und WebOS realisiert werden.

Dreamweaver CS 5.5: Dreamweaver ist ein HTML-Editor bestehend aus einer Kombination eines WYSIWYG-Editors mit paralleler Quelltextbearbeitung. Auch die Syntaxunterstützung für HTML5, CSS3 und diversen Skriptsprachen ist implementiert. In der neusten Version CS5.5 wurde zudem das OpenSource Projekt PhoneGap in Dreamweaver eingeführt. Hiermit ist eine Applikationsprogrammierung für iOS und Android möglich.

JQuery Mobile: Ist ein Touch-optimiertes Web Framework für Smartphones & Tablets.

Es vereinheitlicht ein HTML5-basiertes GUI für Smartphone-Plattformen. Mit seinem leichten Code bietet dieses Framework ein flexibles und einfach anpassbares Design.

7.2.1.3. Entwicklungsumgebung für spezifische Smartphone-Betriebssysteme

Für Smartphone spezifische Anpassungen, welche nicht mit PhoneGap erreicht werden können, muss auf die Entwicklungsumgebung der Smartphone-Betriebssysteme zurückgegriffen werden.

- Eclipse Indigo für Android
- Xcode für IOS

7.2.1.4. Push-Nachrichten

Um Push-Nachrichten an ein Smartphone zu senden, bieten die Hersteller der Betriebssysteme grösstenteils Frameworks an. Für Android bietet Google das Cloud to device messaging (C2DM) Framework an. Bei iOS bietet Apple den Apple Push Notification Service (APNS). Beide Frameworks funktionieren auf eine sehr ähnliche Weise. Im Folgenden soll das C2DM Framework etwas genauer betrachtet werden, da dies in dieser Applikation benutzt wird.

Aufbau C2DM

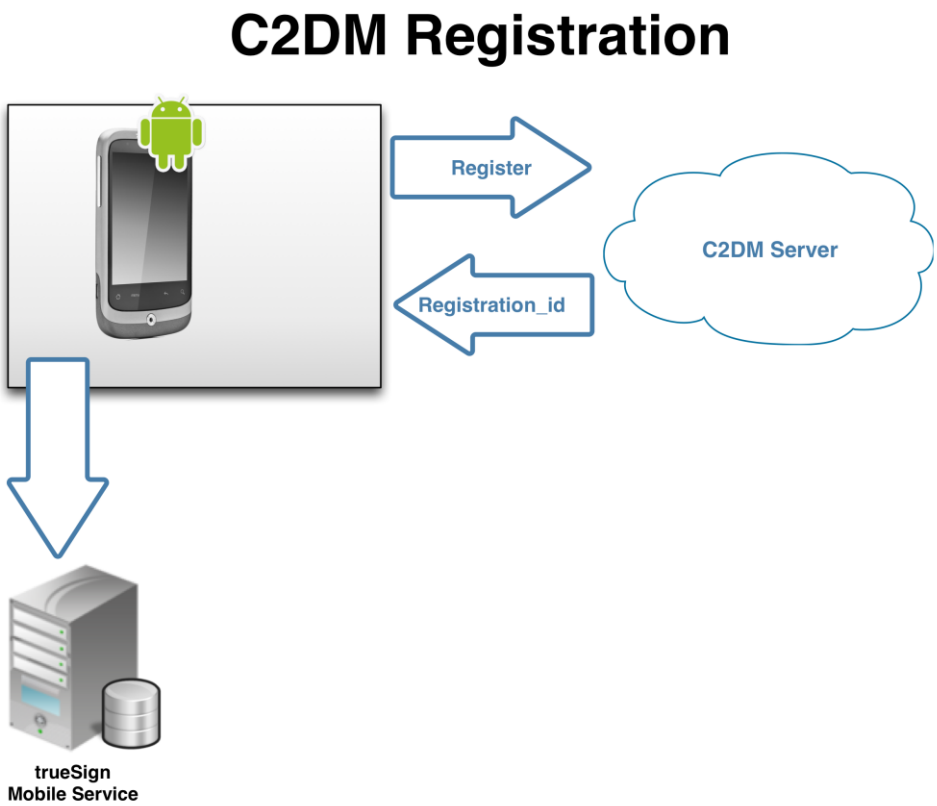


Abbildung 4 Registation bei C2DM

Eine Android-App Registriert sich beim C2DM Server von Google, dieser vergibt für jede App eine RegistrationId welche er der App zurück meldet. Die App muss anschliessend diese RegistrationId ihrem Server mitteilen. Dieser muss diese lokal abspeichern,damit er die App beziehungsweise das Smartphone erreichen kann.

Der Server meldet sich ebenfalls beim C2DM Server an, er erhält von diesem ein Authentization Token.

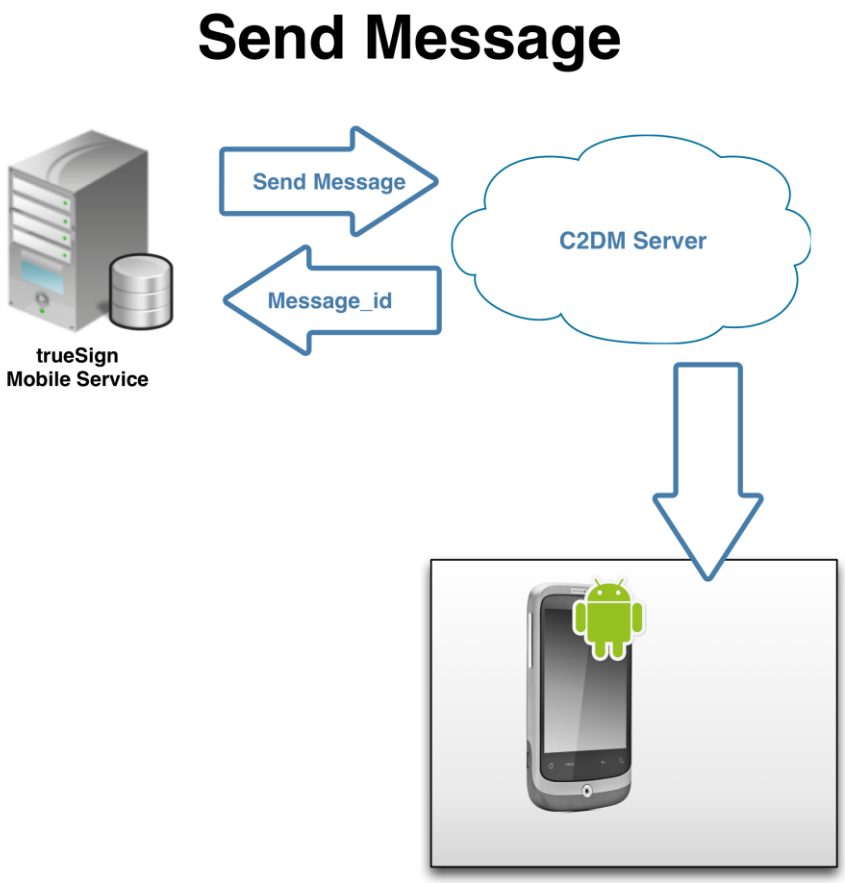


Abbildung 5 Nachricht schicken C2DM

Nun kann der Server mit dem erhaltenen Token als Authentifizierung, sowie der RegistrationId als Empfänger-Adresse eine Nachricht an den C2DM Server übermitteln. Der Google Server leitet diese Nachricht an das Smartphone weiter.

Registration

Bevor sich der Server beim C2DM Server anmelden kann, muss ein Account online[5] eröffnet werden. Dabei müssen gewisse Angaben zur Applikation – wie das Package oder die geschätzte Anzahl Nachrichten pro Tag – angegeben werden. Zudem wird eine Kontakt Email Adresse wie auch eine Sender Email Adresse, welche als Accountname benutzt wird, benötigt. Dieser Accountname muss der Server wie auch die Mobile-App bei der Registrierung beim C2DM Server angeben.

Beschränkungen

Google erlaubt einem Standard Account das Versenden von maximal 200‘000 Nachrichten pro Tag. Dies reicht für die meisten produktiven Anwendungen und ist für unsere Verwendung sicherlich ausreichend. Sollte diese Anzahl überschritten werden, kann bei Google eine Erhöhung der Begrenzung angefordert werden.

Die Payload einer einzelnen Nachricht ist auf maximal 1024 Bytes beschränkt. Daher kann mittels einer solchen Push-Nachricht die Mobile-App nur darüber informiert werden, dass auf dem Server neue Daten zur Verfügung stehen. Diese müssen dann über eine eigene Verbindung abgeholt werden. Für unsere Verwendung stellt dies jedoch keine Beschränkung dar, da wir die Dateien auch ohne diese Beschränkung nicht via Google verschicken würden.

[5] https://developers.google.com/android/c2dm/signup

8. GUI-Analyse

8.1. Personas

In diesem Kapitel sollen zwei imaginäre Personen, Jörg Stadelhofer und Ueli Untermaurer, welche gelegentlich, respektive sehr häufig, digitale Dokumente signieren müssen, als Benutzerrolle repräsentiert werden.

8.1.1. Jörg Stadelhofer

Jörg Stadelhofer

Jörg arbeitet als Dozent unter anderem an der Hochschule für Technik in Rapperswil. Er betreut Studenten bei ihren individuellen Vertiefungsprojekten und verwendet dabei die digitale Unterschrift.

51 Jahre

Dr.iur. HSG (Universität St.Gallen)

Dozent an verschiedenen Hochschulen (HSR Rapperswil, ETH Zürich, Uni Zürich)

Gründer/Mitinhaber verschiedener Firmen (Import/Produktion/DL)

ledig, keine Kinder

Sehr gute Windows-/ Office-Kenntnisse

Ist sehr gewandt mit technischen Geräten und digitale Signaturen sind ihm vertraut

Hauptziele

Er signiert dabei die Projektanträge, welche als PDFs vorliegen, digital und versendet diese an die Projektteilnehmer. Da er nebenbei auch als Unternehmer tätig ist, möchte er auch vermehrt seine geschäftlichen Dokumente digital signieren.

Persönlichkeit, Arbeitsstil & Lernverhalten

Jörg ist sehr organisiert. Alle seine Dokumente werden entsprechend übersichtlich abgelegt. Er arbeitet ausschliesslich mit seinem persönlichen Laptop. Sicherheit ist oberstes Gebot, seine Passwort-Kriterien sind darum sehr vorsichtig. Jörg interessiert sich sehr für IT. Unter anderem auch für die neusten technischen Geräte. Darum hat er auch ein iPhone4S und das neuste Android-Smartphone. Für ihn ist die digitale Signatur die Zukunft.

Pain Points

Obwohl Jörg keine Probleme bei der Installation hat, stört es ihn, immer ein Gerät zur Signatur mit sich herumzutragen. Gerne hätte er den Kartenleser im Notebook integriert, dies ist aber für sein jetziges Apple MacBook leider nicht verfügbar.

8.1.2. Ueli Untermaurer

Ueli Untermaurer

Ueli Untermaurer ist die oberste Instanz der Rüstungsindustrie der Schweiz und kommt vor allem beim Handel mit Verteidigungswaffen nicht um digital signierte Dokumente herum.

Profil

62 Jahre

Kaufmännische Lehre

ehemaliger Parteipräsident der nationalen Rasselbande der Schweiz

Bundesrat (Eidg. Depart. für Verteidigung)

verheiratet, zwei Kinder

wenige Windows-/ Office-Kenntnisse

nicht sehr versiert im Umgang mit technischen Geräten, verwendet digitale Signaturen nur sehr ungern

Hauptziele

In ziemlich allen digitalen Dokumenten, welche bei seinen politischen Aktivitäten entstehen, muss eine Signatur eingebrannt werden, mindestens dreimal täglich. Da er oft unter Zeitdruck ist und ihn der technische Hintergrund nicht interessiert, möchte er so schnell und unkompliziert wie möglich Dokumente unterzeichnen können.

Persönlichkeit, Arbeitsstil & Lernverhalten

Ueli ist leider eher ein Chaot und verliert manchmal den Überblick über seine Termine und Arbeitsgegenstände oder er vergisst seine Login-Daten. Zu bearbeitende Dokumente häufen sich oft an und müssen alle in letzter Minute bearbeitet werden. Er arbeitet zwar täglich mit digitalen Signaturen, interessiert sich aber nicht dafür. Neue Technologien oder Arbeitsprozesse, lehnt er grundsätzlich ab.

Pain Points

Ueli ärgert sich über die komplizierte Installation der SuisseID, da er oft an verschiedenen Geräten seine Dokumente signieren möchte. In den meisten Fällen muss er den Support rufen, um Hilfe und die Rechte für die Hardware zu erhalten.

Er hätte gerne eine Stapelfunktion, mit der er mehrere Dokumente auf einmal signieren könnte. Um Zeit zu sparen, wäre es am einfachsten, alle zu signierenden Dokumente per Checkbox anzuwählen und gleich mit einer Signatur zu versehen, unter Umständen auch ohne das Dokument noch zu öffnen.

8.2. Szenarios

In diesem Kapitel wird eine aktuelle Situation mit konkreten, respektive idealisierten Fällen mit den obigen Personas beschrieben.

8.2.1. IST-Szenario (Problem Scenario)

8.2.1.1. Ausgangssituation

Story #1

Viele Studenten haben sich im Modul IVP eingeschrieben und müssen einen Projektantrag verfassen, welcher von beiden Parteien, Betreuer und Student, unterzeichnet werden muss, damit er danach als Projekt bei der Schulleitung eingegeben werden kann.

Jörg erhält nun ein Dokument per Email eines Studenten, welches er genau studiert. Danach möchte er dieses PDF-Dokument mit seiner SuisseID – diese Smartcard hat er von der Post gekauft – signieren und den unterschriebenen Antrag als Email-Anhang dem jeweiligen Studenten zurücksenden, damit der Student den Antrag einreichen kann.

Story #2

Es wurde entschieden neue, moderne Kampfflugzeuge zu kaufen, da die bestehenden F5-Tiger die Lufthoheit nicht mehr gewährleisten können. Entschieden haben sich Ueli und seine Experten nach langem Hin-und-her für den schwedischen Saab Gripen. Da Ueli die 22 Jets gestaffelt kaufen möchte, wird es für jeden einzelnen Flieger einen eigenen Kaufvertrag geben. Auch die zugehörigen Waffensysteme müssen getrennt beschafft werden, da es sich um einen anderen Lieferanten handelt. Das sind zusätzlich 29 einzelne Verträge.

Da er zeitlich sowieso schon im Verzug ist, muss er nach der Bundesratssitzung dringend 43 der gesamten 51 Verträge unterzeichnen. Zum Glück kann er diese Verträge an seinem Bundes-PC digital signieren lassen und per Email versenden, da der Hersteller in Schweden diese Verträge noch heute unterzeichnet erhalten muss, um eine weiter Verzögerung zu vermeiden.

8.2.1.2. Aktivitäten

User #1

- Jörg liest die per Mail erhaltenen Projektanträge sorgfältig durch. Da die meisten Dokumente im MS Word verfasst wurden, erstellt er daraus PDF-Dokumente.
- Er schliesst den USB-Kartenleser an seinen Laptop an und liest damit die SuisseID ein.
- Er startet das „Sign!“ Programm der Suisse ID, um die Signatur auszuführen.
- Er gibt den persönlichen Private Key ein und bestätigt somit die digitale Unterschrift.
- Nun kann er den Antrag dem jeweiligen Studenten per Mail zurücksenden.

User #2

- Der Benutzer Ueli öffnet das erste seiner 51 PDF-Dokumente. Da er die Verträge nach den etlichen Diskussionen mittlerweile auswendig kann, muss er das Dokument nicht mehr vollständig lesen, er kontrolliert nur kurz die Beträge und das Lieferdatum.
- Er schiebt seine SuisseID in den Kartenleser.
- Er startet das „Sign!“ Programm der Suisse ID, um die Signatur auszuführen.
- Nach Eingabe seines persönlichen Private Key, bestätigt er diesen und das Dokument wird signiert.
- Diesen Prozess wiederholt er für alle restlichen Dokumente.
- Die neuen Dokument mit der digitalen Unterschrift hängt er den schon vorbereiteten Email-Nachrichten an und versendet sie an die Lieferanten.

8.2.1.3. Probleme

Story #1: Blöderweise hat Jörg zu Hause seinen USB-Kartenleser vergessen.

Story #2: Leider funktioniert der Kartenleser wieder nicht und der Helpdesk im Bundeshaus Bern ist zurzeit nicht erreichbar. Erst am nächsten Tag kann ihm ein Arbeitskollege aus einem anderen Department helfen den Kartenleser einzurichten.

8.2.1.4. Abschluss

Story #1: Der Signatur-Prozess verschiebt sich auf den Abend, kann aber doch noch ausgeführt werden.

Story #2: Die signierten Verträge sind verspätet im Postfach der Lieferanten angekommen, was den Liefertermin um ein Jahr nach hinten verschiebt.

8.2.2. SOLL-Szenario (Future Scenario)

Story #1: Da Jörg keine Smartcard mehr benötigt, kann er sie auch nicht mehr zu Hause vergessen. Somit kann er die Projektanträge der Studenten jederzeit mit seinem Laptop und seinem Smartphone signieren.

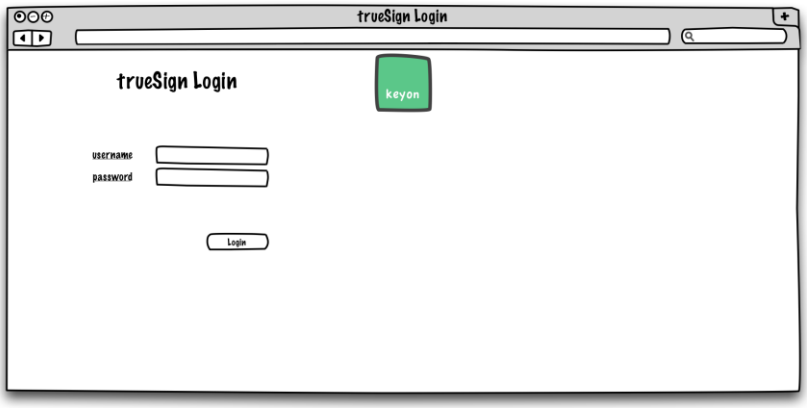
Story #2: Ueli ist froh, dass er sich ab sofort mit keinem Smartcardleser mehr herumschlagen muss. Seine Verträge kann er problemlos per Webportal und mit seinem Smartphone signieren.

8.3. Paper Prototypes

8.3.1. Webportal

Login

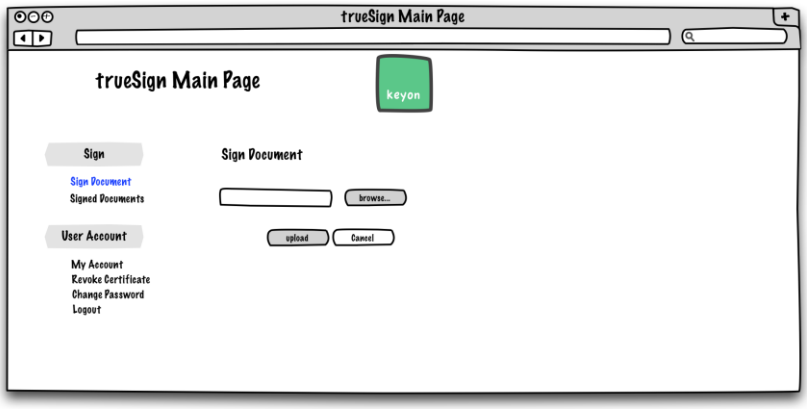
Startseite des Webportals. Der Benutzer gibt hier seine per Post erhaltenen Login-Daten ein.



Main Page

Hauptseite, in welchem auf der linken Seite die Navigation zu allen wichtigen Funktionen führen.

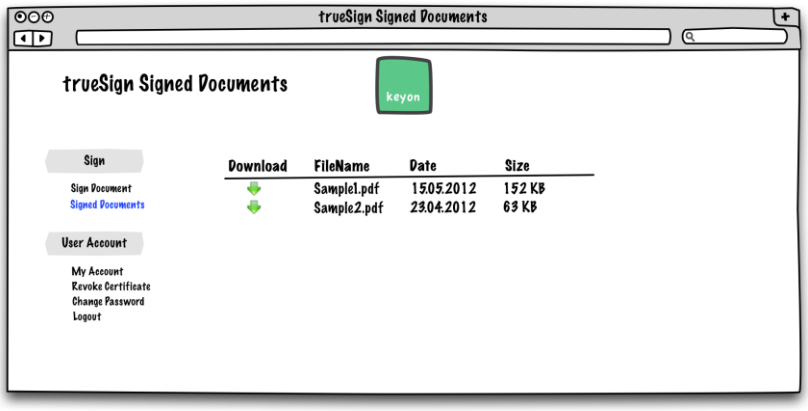
Hier wählt der Benutzer mit dem *Browse*-Button sein zu signierendes Dokument aus. Um es zur Bestätigung an sein Smartphone zu senden, muss er die *Upload*-Schaltfläche betätigen. Ansonsten kann er den Vorgang mit *Cancel* abbrechen.



Signed Documents

Unter dem Menüpunkt Signierte Dokumente, findet er die Dokumente wieder, welche er mit dem Smartphone zum Signieren bestätigt hat. Auf dieser Seite kann er diese Dokumente signiert wieder herunterladen und weiterverwenden.

Um die Übersicht über die bearbeiteten Dokumente zu behalten, werden die Files mit ihren Informationen angegeben.



MyAccount

Unter diesem Abschnitt können Benutzerdaten eingesehen und angepasst werden.

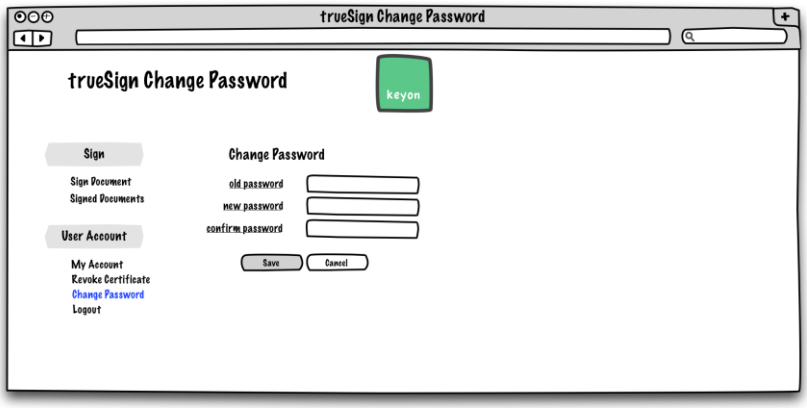
Daten wie Benutzername, Name und E-Mail-Adresse, können jedoch aus Sicherheitsgründen nicht im Portal geändert werden, und müssen von der Registration Authority verifiziert werden.

Revoke Certificate

Der Benutzer kann z.B. bei Verlust des Smartphone sein Zertifikat sperren lassen. Dabei gibt er den Revokations-PIN, welcher er per Post erhalten hat, ein, sobald er auf *revoke* geklickt hat.

Change Password

Nachdem der Benutzer seine Initialisierungsdaten im Webportal und seinem Smartphone eingeben hat, kann er sein Passwort auf dieser Seite ändern.



8.3.2. Mobile-Applikation

8.3.2.1. Erste Anmeldung (nach Installation)

Geräte initialisieren

Nach dem ersten Start der App, wird der Benutzer aufgefordert, seinen Benutzernamen und das per Post erhaltene Initialisierungspasswort einzugeben.

Bitte aktivieren

Um das Zertifikat aktivieren zu können, muss sich der Benutzer zuerst auf dem Webportal anmelden und die Aktivierung starten. Er erhält darauf eine Push-Meldung. Die Einstellungen sind jedoch eingeschränkt.

Aktivierung des Zertifikats

Die erhaltene Push-Meldung wechselt zur Ansicht, welche den Benutzer zur Eingabe seiner per Post erhaltener Initialisierungs-PIN und einer neuen persönlicher PIN auffordert.

8.3.2.2. Signierungsprozess

Hauptansicht

Dies ist die Startseite wenn die App geöffnet wird. Von hier aus ist es möglich die Dokumente anzuzeigen, welche zuvor auf das Webportal hochgeladen wurden.

Dokumentansicht

Nachdem ein Dokument in der Hauptansicht angewählt wurde, kann es hier mit Scroll- und Zoomaktionen betrachtet werden. Es steht eine Signierungsschaltfläche zur Verfügung, um den Signierungsprozess einzuleiten.

Signieren

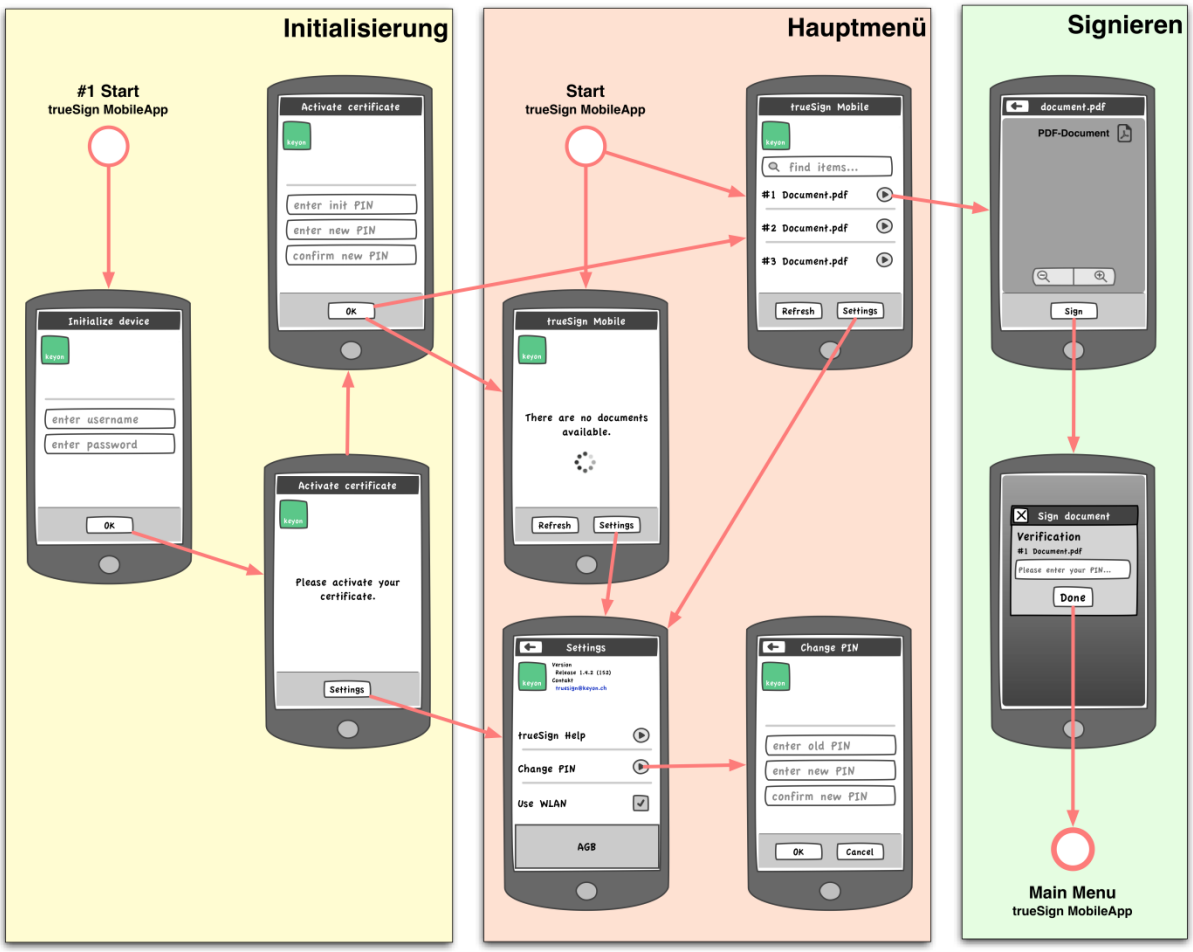
Ein Popup fordert den Benutzer zur Eingabe der PIN seines Private Keys auf und diesen mit Klick auf Done zu bestätigen.

8.3.2.3. Weitere App-Ansichten

Keine Dokumente	**Einstellungen**	**PIN ändern**
Sind keine Dokumente verfügbar, wird eine entsprechende Meldung angezeigt. Ebenfalls ist hier das Fortschrittssymbol illustriert, das beim Laden der Dokumentenliste erscheint, sobald auf den Refresh-Butten gedrückt wird.	Hier können Informationen über die App, wie Version, Kontaktinformationen und die AGB gefunden werden. Es wird auch auf eine kleine Hilfe verwiesen. Hier kann auch der aktuelle PIN geändert werden. Zusätzlich kann eingestellt werden, ob ausschliesslich die WLAN-Verbindung genutzt werden soll.	Der Benutzer kann in dieser Einstellung seine PIN vom Private Key ändern.

8.4. GUI-Map

8.4.1. Mobile-Applikation



9. Schnittstellenbeschreibung

9.1. Schnittstellen Übersicht

In der Applikation gibt es grundsätzlich drei Schnittstellen, welche in folgender Abbildung ersichtlich sind.

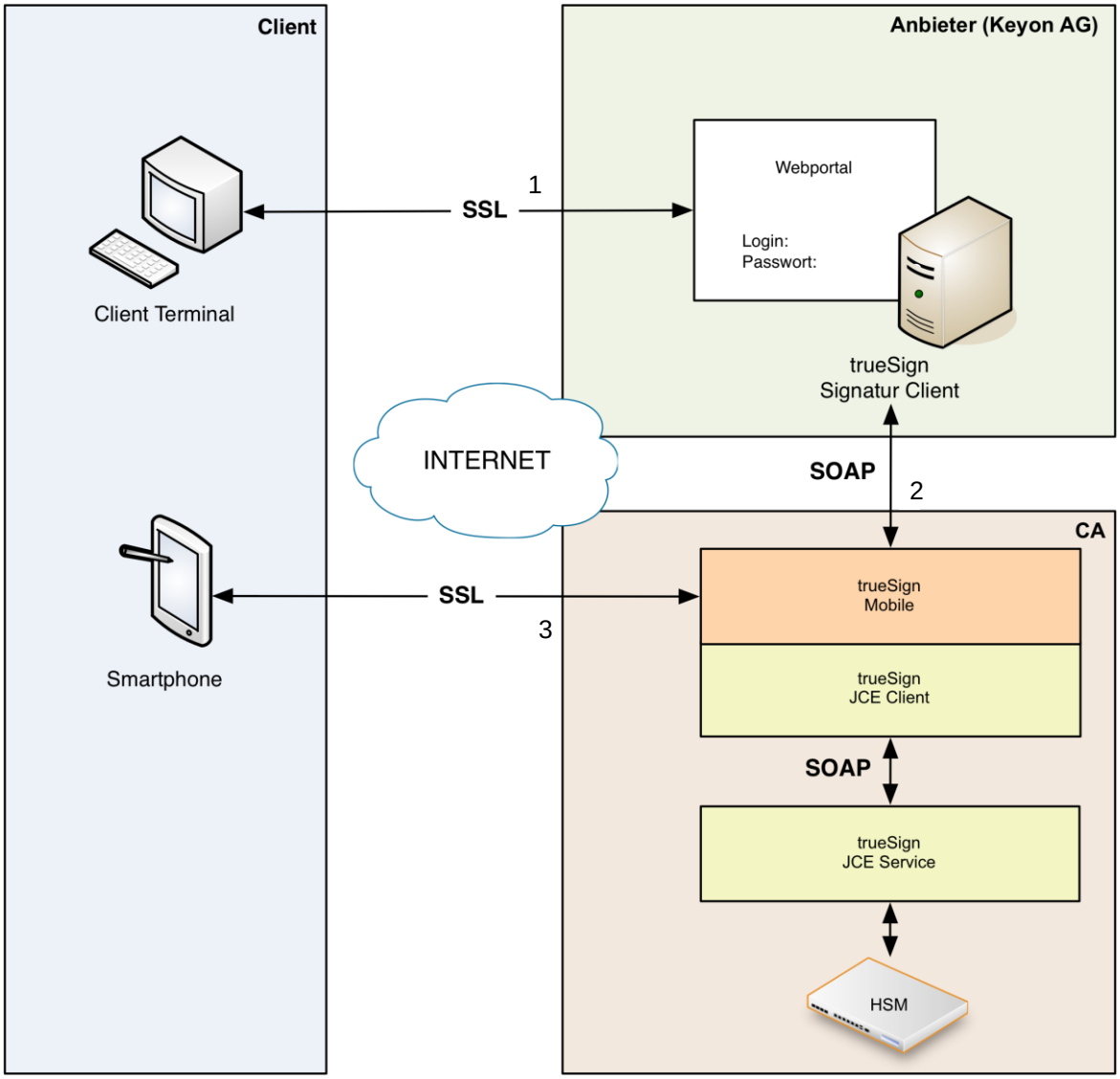


Abbildung 6 Schnittstellen Übersicht

Bei der Schnittstelle 1 (trueSign Webportal) handelt es sich um einen Standard-Zugriff auf eine Webseite mittels eines Browsers. Die Verbindung wird mittels SSL ohne Client-Authentifizierung gesichert. Auf diese Schnittstelle wird hier nicht genauer eingegangen. Allerdings wird noch eine zusätzliche SOAP-Schnittstelle benötigt, welche das Registrieren von mobilen Geräten ermöglicht. Diese soll genauer beschrieben werden.

Die Schnittstellen 2 (trueSignMobile Service Interface) & 3 (Mobile Interface) sind Webservice-Verbindungen zwischen dem trueSign Webportal, dem Signaturservice sowie dem Smartphone des Users. Diese Verbindungen werden in den folgenden Kapiteln genauer betrachtet. Es handelt sich dabei um denselben Webservice, bei manchen Operationen ist jedoch das trueSign Webportal der Client und bei anderen die trueSign Mobile-App.

9.2. trueSign Webportal

Um Verbindungen von mobilen Geräten mit dem trueSign JCE Service möglichst gut beschränken zu können, dürfen nur zuvor registrierte Geräte, welche über eine gültige SessionID verfügen, eine Verbindung aufbauen. Dies bedingt, dass einerseits vor jeder Verbindung welche die Mobile-App zum trueSign Mobile Service aufbauen will, eine Push-Meldung mit einer SessionID an das Smartphone geschickt werden muss und andererseits muss die Registrierung via dem trueSign Webportal erfolgen, was folgend kurz beschrieben wird.

9.2.1. Anmerkungen

Jedes mobile Gerät verfügt über eine IMEI (International Mobile Equipment Identity). Diese soll für die Zuordnung des Gerätes zu einem User genutzt werden. Zudem erhalten die Geräte eine Registration-ID von den jeweiligen Providern der Push-Nachrichten (z.B. Google oder Apple). Diese ID soll fortan als MobileID bezeichnet werden. Sie wird vom Push-Framework (C2DM bei Google, APNS bei Apple) bei der Registrierung zugeteilt, kann aber jederzeit erneuert werden. Das trueSign Webportal muss deshalb auch die Möglichkeit für eine Änderung der MobileID bereitstellen.

9.2.2. Operationen

Nachricht	Sender	Parameter		Zweck
RegisterMobileAppRequest	trueSign Mobile-App	IMEI	String	Ein mobiles Gerät erstmalig beim trueSign Webportal registrieren
		MobileID	String	
		Uid	String	
		InitPIN	String	
RegisterMobileAppResponse	trueSign Webportal	response	Boolean	

Nachricht	Sender	Parameter		Zweck
RenewMobileAppRequest	trueSign Mobile-App	IMEI	String	Ein mobiles Gerät erneuert die MobileID beim Webportal
		MobileID	String	
		Uid	String	
RenewMobileAppResponse	trueSign Webportal	response	Boolean	

9.3. trueSignMobile Service Interface

Hierbei handelt es sich um eine neu entwickelte SOAP-Schnittstelle. In diesem Kapitel werden nur die Operationen, bei welchen das trueSign Webportal der Client ist, beschrieben.

9.3.1. Verbindung

Bei dieser Schnittstelle handelt es sich um eine Duplex-Verbindung. Sie soll mittels SSL gesichert werden.

9.3.2. Operationen

9.3.2.1. SendMobileTanRequest

Nachricht	Sender	Parameter		Zweck
SendMobileTanRequest	trueSign Client	MobileID	String	MobileTAN an Smartphone schicken
SendMobileTanResponse	trueSign Service	TAN	String	

9.3.2.2. SignMobilePDFRequest

Nachricht	Sender	Parameter		Zweck
SignMobilePDFRequest	trueSign Client	UID	String	Hochladen eines Dokumentes welches signiert werden soll
		Alias	String	
		Pdf	Binary	
		DocName	String	
		IMEI	String	
		MobileID	String	
SignMobilePDFResponse	trueSign Service	Signature	Binary	

9.3.2.3. ActivateCertificate

Nachricht	Sender	Parameter		Zweck
ActivateCertificateRequest	trueSign Client	Uid	String	Aktivierung des Zertifikates beim ersten Login auf dem trueSign Webportal
		Alias	String	
		IMEI	String	
		MobileID	String	
ActivateCertificateResponse	trueSign Service	Return	Boolean	

9.3.2.4. ChangePin

Nachricht	Sender	Parameter		Zweck
ChangePinRequest	trueSign	Uid	String	Änderung des PIN für den PrivateKey
		Alias	String	
		IMEI	String	
		MobileID	String	
ChangePinResponse	trueSign Service	Return	Boolean	

9.3.2.5. ProcessFault

Ein ProcessFault wird bei allen Operationen im Fehlerfall von trueSign Service zurückgegeben.

Nachricht	Sender	Parameter		Zweck
Alle	trueSign Service	ProcessFault	String	Fehlermeldung bei Problemen mit der Signatur
		ReasonCode	Int	

9.4. Mobile Interface

Dieses Kapitel behandelt die Verbindung zwischen der trueSign Mobile-App und dem trueSign Mobile Service. Diese Kommunikation läuft teilweise über Push Notification Services, welche von dritten (z.B. Apple oder Google) angeboten werden. Diese Nachrichten sollen folgend ebenfalls beschrieben werden.

9.4.1. Verbindung

Diese Schnittstelle ist ebenfalls eine Duplex-Verbindung. Sie soll mittels SSL verschlüsselt werden, jedoch ohne Client-Authentifizierung, da die mobilen Geräte über keine eigenen Zertifikate verfügen. Es ist aber wichtig, dass das Serverzertifikat genau überprüft wird, um zu verhindern, dass sich die trueSign Mobile-App mit einem falschen Server verbindet.

9.4.2. Operationen

9.4.2.1. sendPushReadyToSign

Anmerkungen: Die SessionID besteht aus einem SHA-256 Hashwert des gesamten zu signierenden Dokumentes. Zusätzlich wird dem Dokument, vor der Berechnung des Hashes noch ein 128 Bit

Random-Seed hinzugefügt, um die Eindeutigkeit der SessionID zu gewährleisten. Diese SessionID wird mit der IMEI des Users verknüpft und ist nur für eine bestimmte Zeit gültig

Nachricht	Sender	Parameter		Zweck
SendPushReadyToSignRequest	JCE Service	MobileID	String	Senden einer Push Notification an die trueSign Mobile-App über ein zum Download bereitstehendes PDF. **Via PushProvider**
		SessionIDs	String	
SendPushReadyToSignResponse	Push Provider	Return	Boolean	

9.4.2.2. GetDocument

Nachricht	Sender	Parameter		Zweck
GetDocumentRequest	trueSign Mobile-App	IMEI	String	Download der zu signierenden PDF-Datei auf das mobile Gerät
		SessionID	String	
GetDocumentResponse	trueSign Mobile	Date	Date	
		Size	String	
		Name	String	
		Document	Binary	

9.4.2.3. SignAcknowledge

Nachricht	Sender	Parameter		Zweck
SignAcknowledgeRequest	trueSign Mobile-App	IMEI	String	Bestätigung, um die Signatur ausführen zu können.
		SessionID	String	
		privateKeyPin	String	
SignAcknowledgeResponse	trueSign Mobile	Return	Boolean	

9.4.2.4. SignDecline

Nachricht	Sender	Parameter		Zweck
SignDeclineRequest	trueSign Mobile-App	IMEI	String	Den Signaturauftrag ablehnen
		SessionID	String	
SignDeclineResponse	trueSign Mobile	Return	Boolean	

9.4.2.5. sendPushActivateCert

Nachricht	Sender	Parameter		Zweck
SendPushActivateCertRequest	trueSign Mobile	MobileID	String	Senden einer Push Notification an die trueSign Mobile-App über ein zu aktivierendes Zertifikat **Via PushProvider**
		ActivateTAN	String	
SendPushActivateCertResponse	Push Provider	Return	Boolean	

9.4.2.6. activateCert

Nachricht	Sender	Parameter		Zweck
activateCertRequest	trueSign Mobile-App	IMEI	String	Zertifikataktivierung nach dem ersten Login beim trueSign Webportal
		ActivateTAN	String	
		initPin	String	
		newPin	String	
activateCertResponse	trueSign Mobile	Return	Boolean	

9.4.2.7. sendPushChangePIN

Nachricht	Sender	Parameter		Zweck
sendPushChangePinRequest	trueSign Mobile	MobileID	String	Senden einer Push Notification an die trueSign Mobile-App mit der Aufforderung die PIN zu ändern **Via PushProvider**
		ChangeTAN	String	
sendPushChangePinResponse	Push Provider	Return	Boolean	

9.4.2.8. changePIN

Nachricht	Sender	Parameter		Zweck
changePINRequest	trueSign Mobile-App	IMEI	String	Änderung des PIN für den PrivateKey
		ChangeTAN	String	
		oldPin	String	
		newPin	String	
changePINResponse	trueSign Mobile	Return	Boolean	

9.4.3. Ablauf der Kommunikation

Folgende Diagramme erläutern den Ablauf der Kommunikation zwischen der trueSign Mobile-App, des trueSign Webportals und dem trueSign Service mit Einbezug der Sicherheitsmechanismen. Dabei wurden einzelne Komponenten zusammengefasst oder weggelassen, um die Übersichtlichkeit zu verbessern.

9.4.4. Kommunikation beim Login

Dieses Diagramm zeigt die Kommunikation während des Logins mit dem Versenden der Mobile-TAN, welche anschliessend beim trueSign Webportal eingegeben werden muss. Um die Verbindung zum Push Provider nur an einem Ort in der Applikation halten zu müssen, wird auch die Mobile-TAN vom trueSign Service und nicht direkt vom trueSign Webportal verschickt.

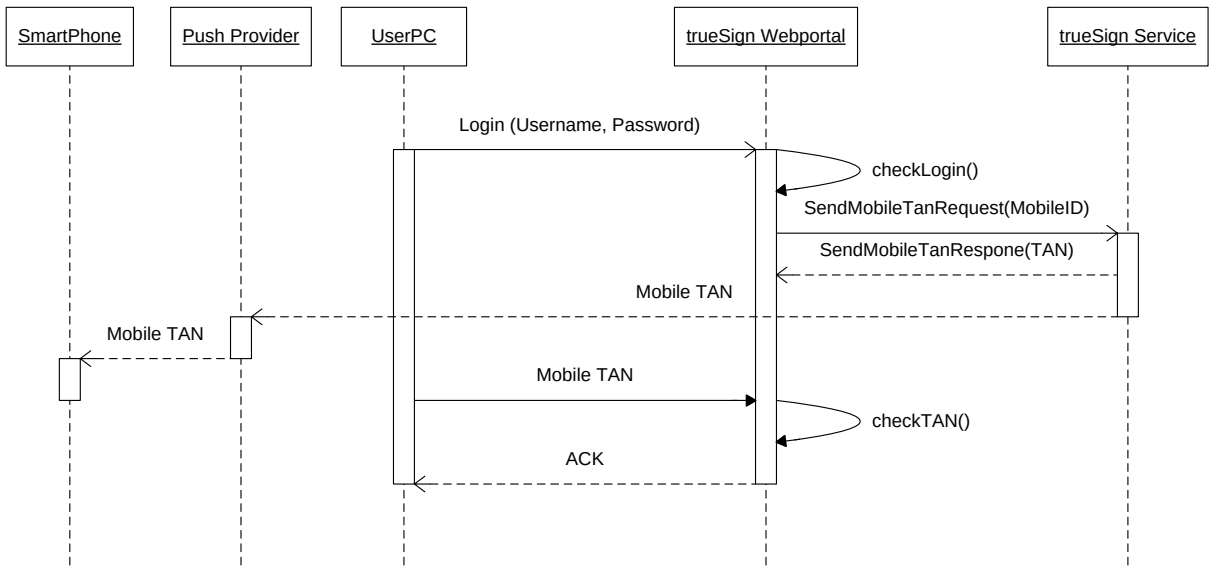


Abbildung 7 Kommunikation Login

9.4.5. Kommunikation beim Signieren

Folgendes Diagramm stellt die Kommunikation bei der Signatur einer Datei dar. Aus Gründen der Übersichtlichkeit wurde der Login-Prozess weggelassen.

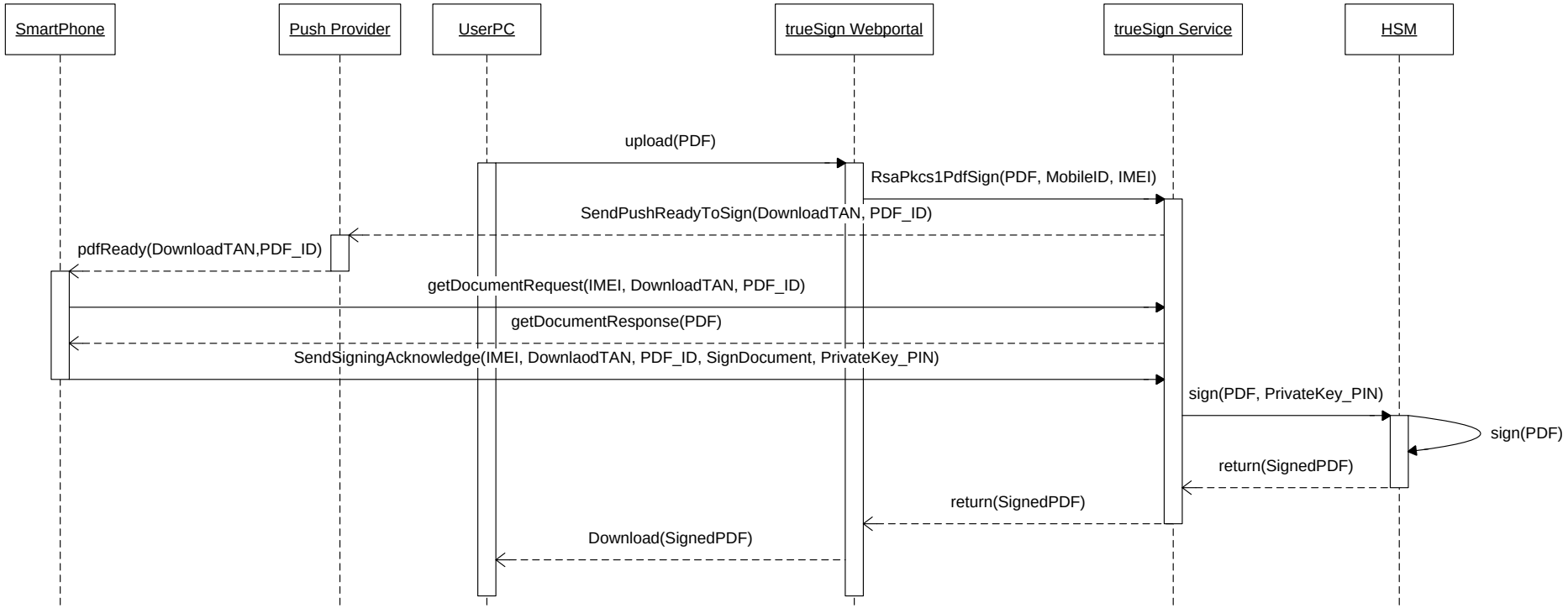


Abbildung 8 Kommunikation Signatur

9.4.6. Kommunikation bei der Aktivierung der trueSign Mobile-App

Dieses Diagramm beschreibt den Kommunikationsablauf bei der ersten Verwendung der trueSign Mobile-App.

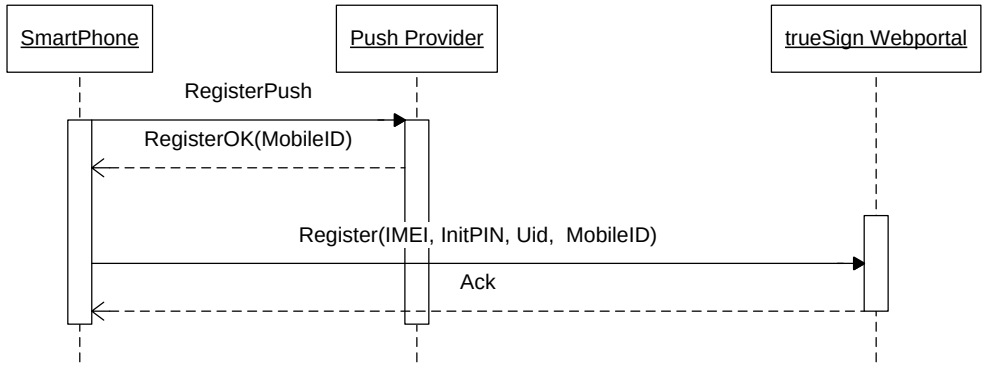


Abbildung 9 Kommunikation Aktivierung Mobile-App

9.4.7. Kommunikation bei der Erneuerung der MobileID

Folgendes Diagramm veranschaulicht den Erneuerungsvorgang der MobileID.

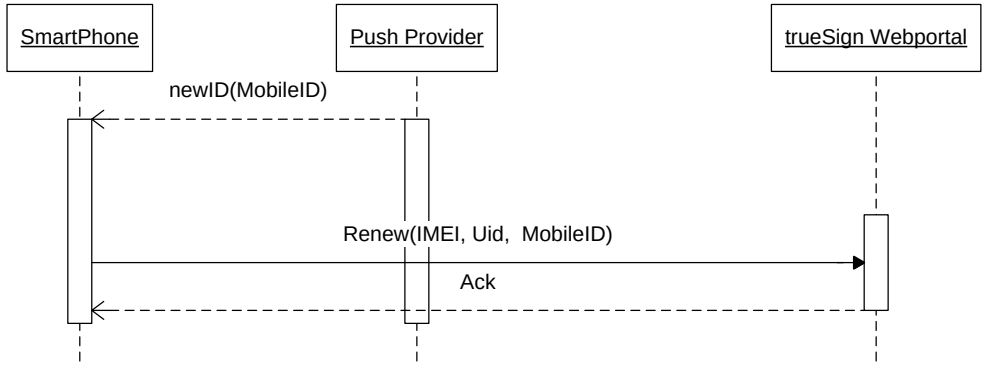


Abbildung 10 Kommunikation Erneuerung MobileID

9.4.8. Kommunikation bei der Aktivierung

In diesem Diagramm wird die Kommunikation während der Aktivierung des Benutzerkontos aufgezeigt.

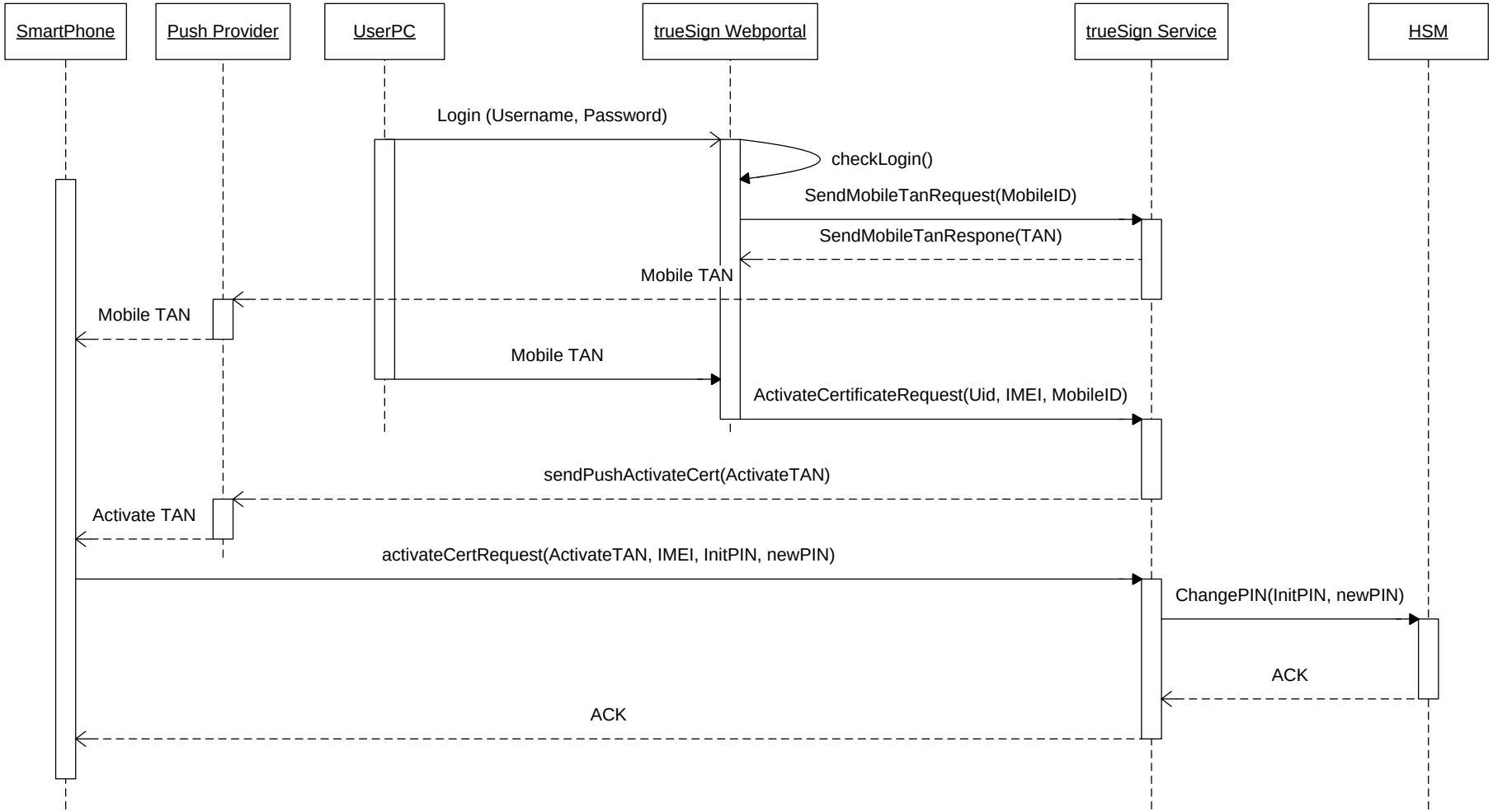


Abbildung 11 Kommunikation Aktivierung

10. Design

10.1. Architektonische Darstellung (Architectural Representation)

Die Applikation besteht grundlegend aus zwei Teilen: aus der trueSign Mobile-App und dem trueSign Webportal. Dieses Kapitel soll für beide Teile gelten. Allfällige Einschränkungen auf nur einen Teil der Applikation werden klar ersichtlich dargestellt.

10.2. Architektonische Ziele & Einschränkungen (Architectural Goals and Constraints)

10.2.1. Ziele

- Die Applikation soll möglichst modular aufgebaut werden, so dass ein späterer Ausbau der Funktionalität einfach möglich ist.
- Die trueSign Mobile-App soll so weit als möglich mit dem PhoneGap-Framework aufgebaut werden. Dieses Framework ermöglicht ein einmaliges Entwickeln einer Mobile-App und ein mehrfaches Deployen auf alle gängigen mobilen Betriebssysteme.

10.3. Physische Sicht

Die Applikation ist ein sogenanntes verteiltes System. Einerseits wird das trueSign Webportal entwickelt, bei welchem sich der User einloggen und die zur Signatur vorgesehene PDF-Datei hochladen kann. Andererseits entsteht die trueSign Mobile-App, welche zur Bestätigung der Signatur auf einem zweiten Kanal verwendet wird. Zudem wird der bestehende trueSign JCE Service leicht erweitert, um den Versand des PDF an das mobile Gerät zu ermöglichen.

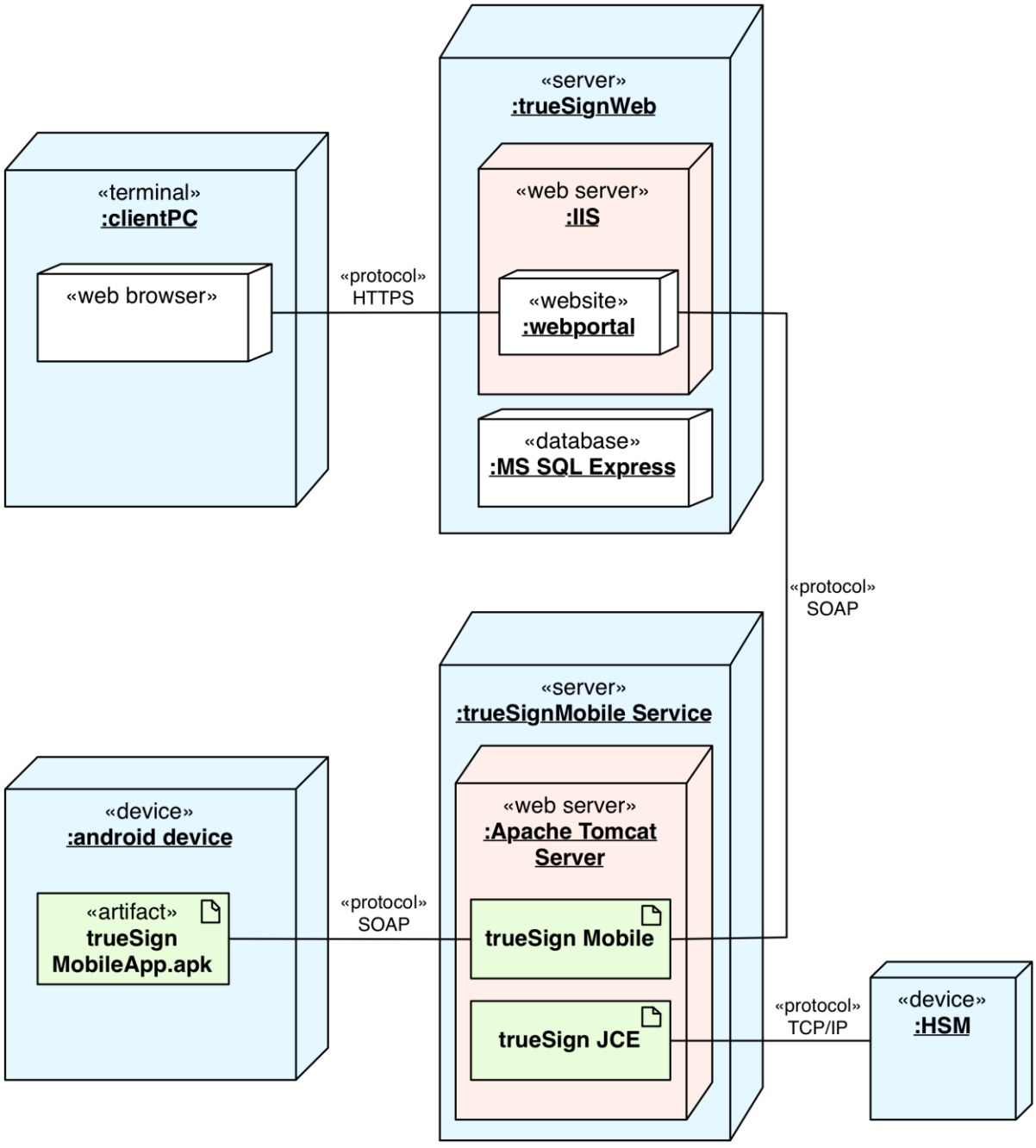


Abbildung 12 Physische Sicht

10.4. Logische Sicht

Die gesamte Java Applikation befindet sich im Package: ch.keyon.security und ist weiter aufgeteilt in die trueSign Mobile-App im Package: ch.keyon.security.mobileClient und ch.keyon.security.mobile für den trueSignMobile Service. In folgender Abbildung sind die Packages dargestellt.

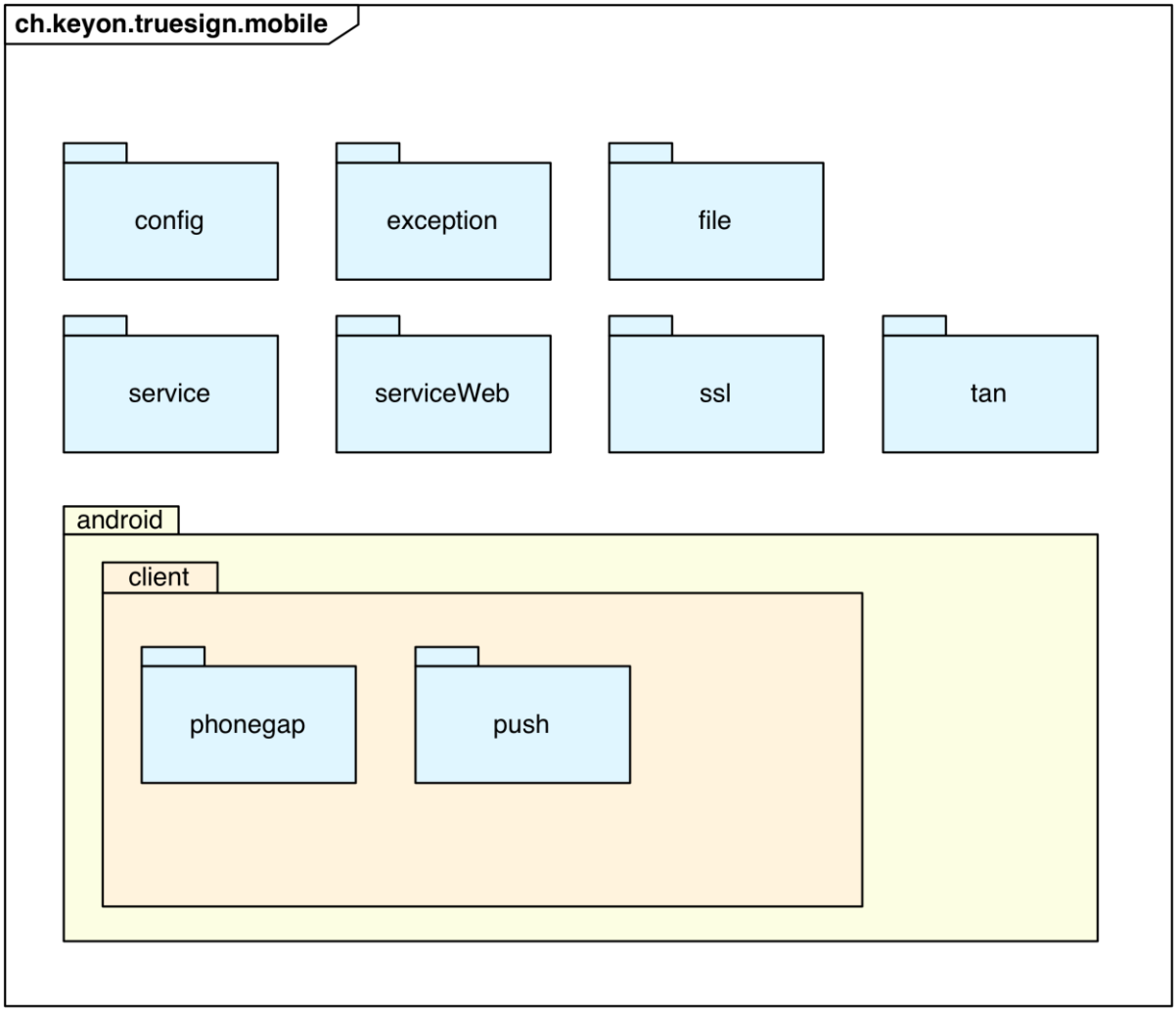


Abbildung 13 Logische Sicht

Bei allen in obiger Abbildung dargestellten Packages, handelt es sich um für die Arbeit neu erstellte Packages. Bestehende Packages (des trueSign JCE Service) werden hier nicht beschrieben.

10.4.1. Designentscheidungen

Da die Arbeit bereits viele verschiedene Technologien beinhaltet, soll sich bei der Kommunikation auf eine Technologie beschränkt werden. Deshalb wird auch für die Kommunikation mit dem Smartphone auf SOAP gesetzt, wenn dies auch einen relativ grossen Overhead bedeutet.

10.4.2. Schnittstellen

Siehe → 9. Schnittstellenbeschreibung

10.4.3. Java Design Packages

In diesem Kapitel werden zuerst die Packages der trueSignMobile Service Server Applikation beschrieben und anschliessend die der trueSign Mobile-App.

10.4.3.1. Package mobile

Beschreibung des Packages

Das Package mobile ist das Top-Package der Applikation. Im Serverteil trueSign Mobile Service befindet sich darin nur die RequestHandler-Klasse. Diese kann als Hauptklasse des trueSign Mobile Service betrachtet werden. Sie nimmt die Request der SOAP-Service-Implementation entgegen, leitet diese an spezifische Klassen weiter oder behandelt sie selbst.

Klassendiagramm

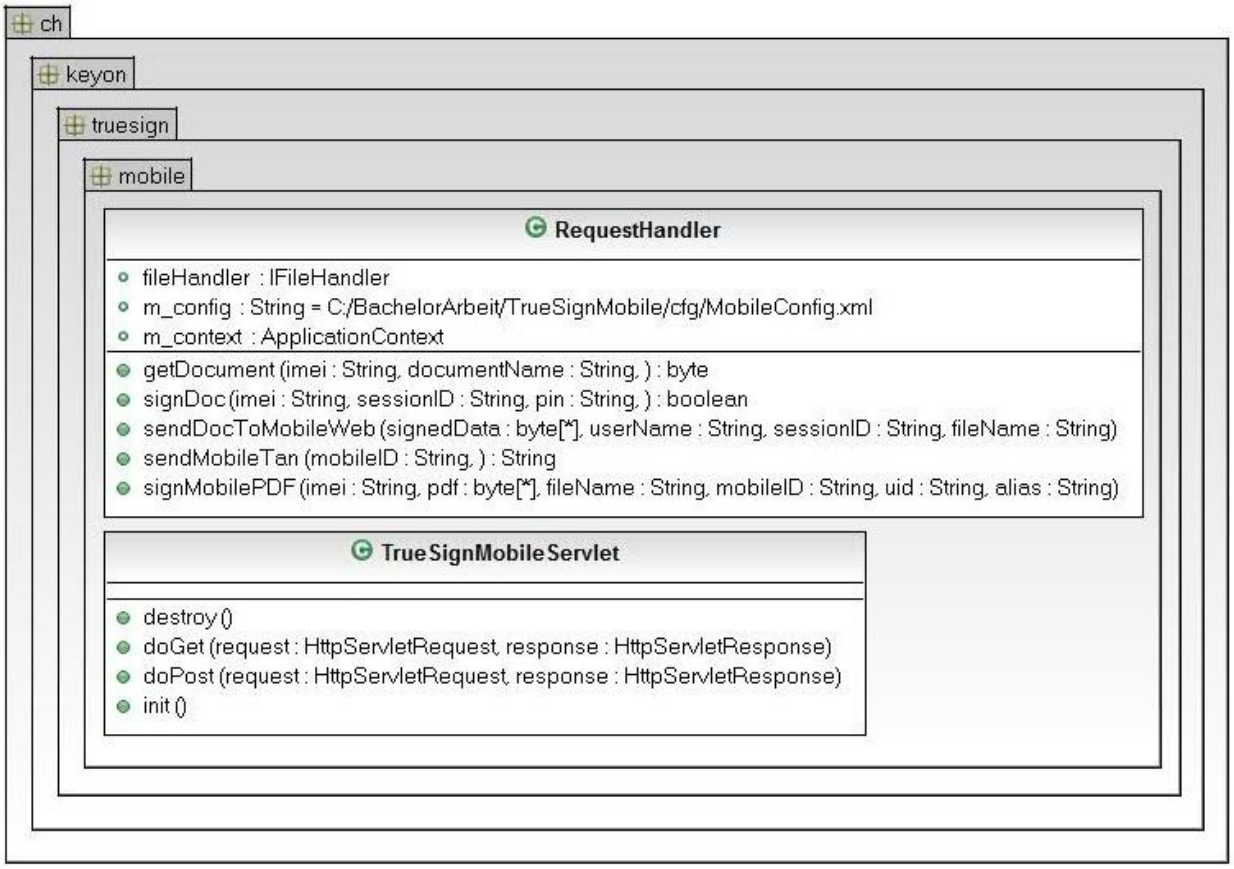


Abbildung 14 Klassendiagramm mobile

Kernoperationen

Hier werden die wichtigsten Operationen des Packages kurz beschrieben

getDocument(String imei, String documentName)
Diese Methode behandelt den GetDocumentRequest des SOAP-Services. Sie prüft, ob die übergebene dem Dokument entsprechende SessionID, der IMEI zugeordnet und noch gültig ist. Falls dies der Fall ist, wird das Dokument ausgelesen und an das mobile Gerät zurückgeschickt. Falls die SessionID nicht mehr gültig, oder nicht der übergebenen IMEI zugeordnet ist, wird eine MobileException geworfen.

signDoc(String imei, String sessionID, String pin)
In dieser Methode wird die SOAP-Action-SignAcknoledge abgearbeitet. Zuerst wird ebenfalls die SessionID sowie die IMEI geprüft. Sind die Angaben gültig, wird ein Aufruf des trueSign JCE Services vorbereitet. Das entsprechende Dokument wird ausgelesen und mitsamt der UserID zu einem DataObject zusammengeführt. Dann wird ein KeyIdentifier Object, bestehend aus der PIN und des Alias des Users, erstellt. Dieser Identifier und das DataObject wird dem trueSign JCE Service via SOAP übergeben, welcher das signierte Dokument retourniert. Dieses wird wiederum über SOAP an das trueSign Webportal geschickt. Falls die Signatur nicht erstellt werden kann, wird eine ProcessException geworfen.

sendMobileTan(String mobileID)
Hier wird der SendMobileTanRequest des SOAP Services behandelt. Mittels SessionManager wird eine neue Mobile-TAN erstellt und dem MessageUtil übergeben, welches die TAN an die gelieferte MobileID schickt. Die erstellte TAN wird ebenfalls an das trueSign Webportal zurückgegeben.

signMobilePDF(String imei, byte[] pdf, String fileName, String mobileID, String Uid, String alias)
Diese Methode behandelt den SendMobilePDFRequest des SOAP Services. Das übergebene Dokument wird mit den nötigen Angaben wie dem FileName und der UserID zwischengespeichert. Daraus wird eine SessionID berechnet, die via MessageUtil an das Smartphone geschickt wird.

10.4.3.2. Package android

Im android Package werden alle androidspezifischen Operationen gehandhabt. Im Serverteil ist ausschliesslich das Versenden von Push-Nachrichten betroffen. Zu diesem Zweck muss der Server zuerst bei Google authentisiert werden, anschliessend können die Nachrichten verschickt werden.

Klassendiagramm

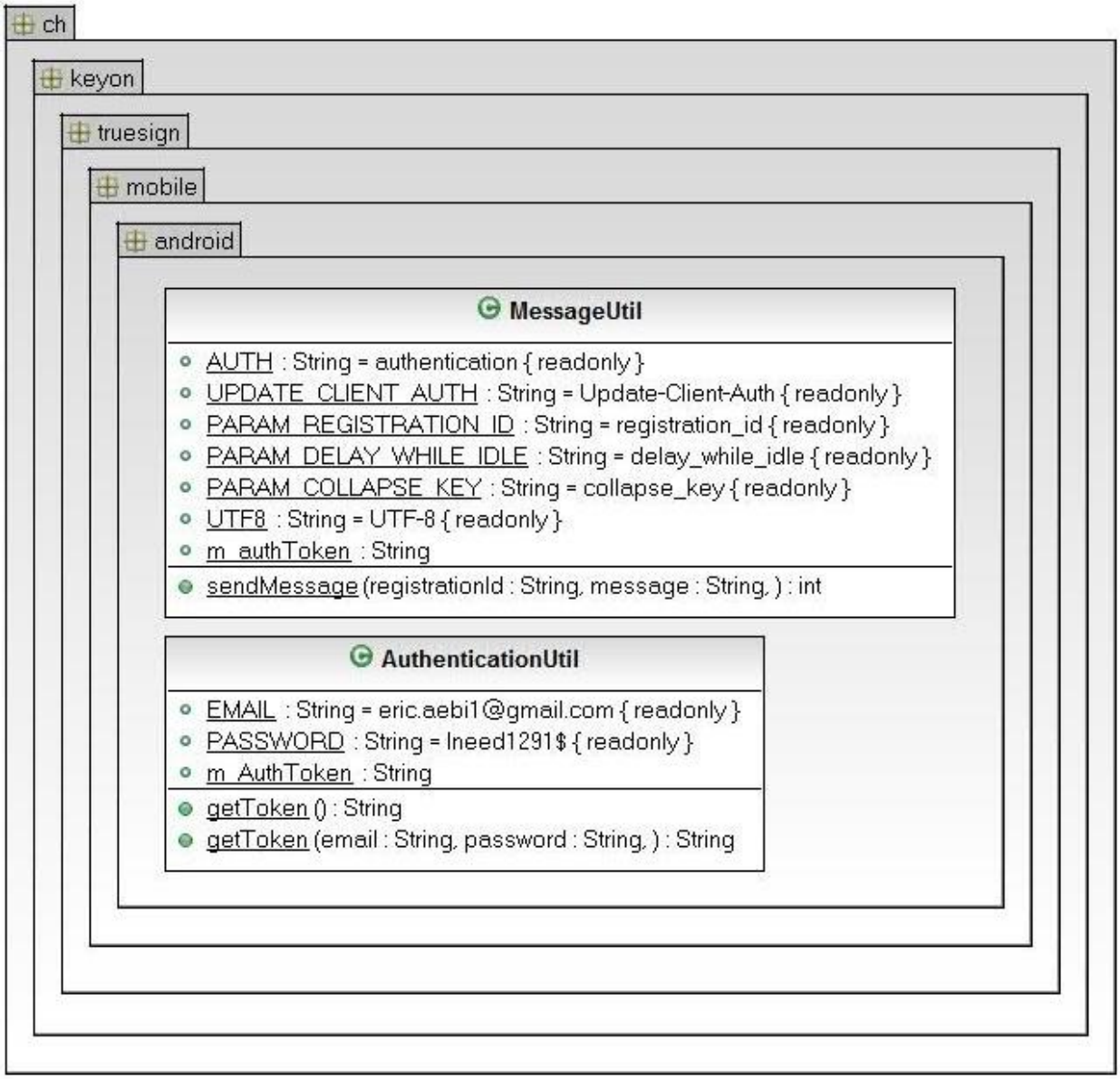


Abbildung 15 Klassendiagramm android

Kernoperationen

Hier werden die wichtigsten Operationen des Packages kurz geschildert.

getToken()
Die Klasse AuthenticationUtil speichert nach einmaliger Authentisierung das AuthenticationToken von Google ab. Ist dies bereits geschehen, gibt diese Methode das Token zurück. Falls noch kein Token angefordert wurde, fordert die Methode ein neues Token an und retourniert dieses.

sendMessage(String registrationId, String message)
Mit dieser Methode werden Push-Nachrichten versendet. Dabei wird die RegistrationId des Smartphone, welches die Nachricht erhalten soll, übergeben. Sie dient als „Telefonnummer“ und identifiziert das Gerät sowie die Mobile-App. Die überlieferte Message wird unverändert an das Smartphone geschickt.

10.4.3.3. Package file

In diesem Package befinden sich alle Klassen, welche sich um die signierten bzw. zu signierenden Dateien kümmern. Die Klasse FileToSign ist ein reines DataObject und wird zur Zwischenspeicherung aller Daten, welche eine Datei betreffen, benutzt. Das Interface IFileHandler dient zu Testzwecken um den Filehandler während Tests durch ein Mock-Object ersetzen zu können.

Klassendiagramm

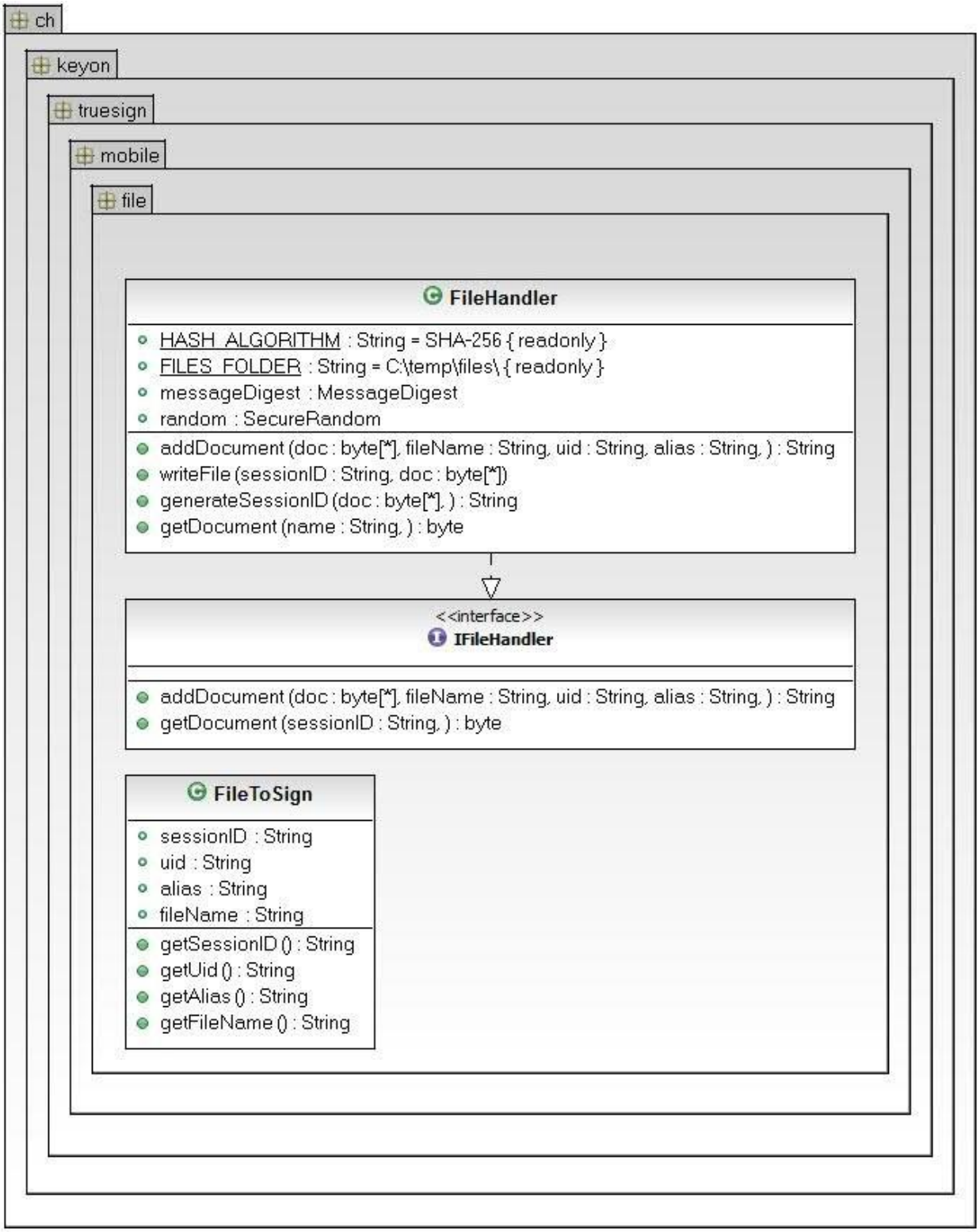


Abbildung 16 Klassendiagramm file

Kernoperationen

addDocument(byte[] doc, String filename, String uid, String alias)
Diese Methode errechnet die SessionID des zu signierenden Dokumentes. Diese ist ein SHA-256 Hash aus dem gesamten Dokument und einem zusätzlichen 128 Byte langen Seed, um die Eindeutigkeit der Session zu gewährleisten. Das File wird auf der Festplatte temporär abgelegt. Die dazugehörigen Informationen, wie die UserId und der Alias, werden in der Applikation im SessionManager zwischengespeichert.

getDocument(String name)
In dieser Methode wird das temporär auf der Festplatte gespeicherte File ausgelesen und retourniert.

10.4.3.4. Package service

Dieses Package beinhaltet alle Klassen des SOAP-Services. Die Implementation befindet sich im Sub Package impl. Die Klassen, welche sich direkt im service Package befinden sind durch JAX-WS generierte SOAP-Klassen. Deshalb wird hier nur die Implementation genauer erläutert.

Klassendiagramm

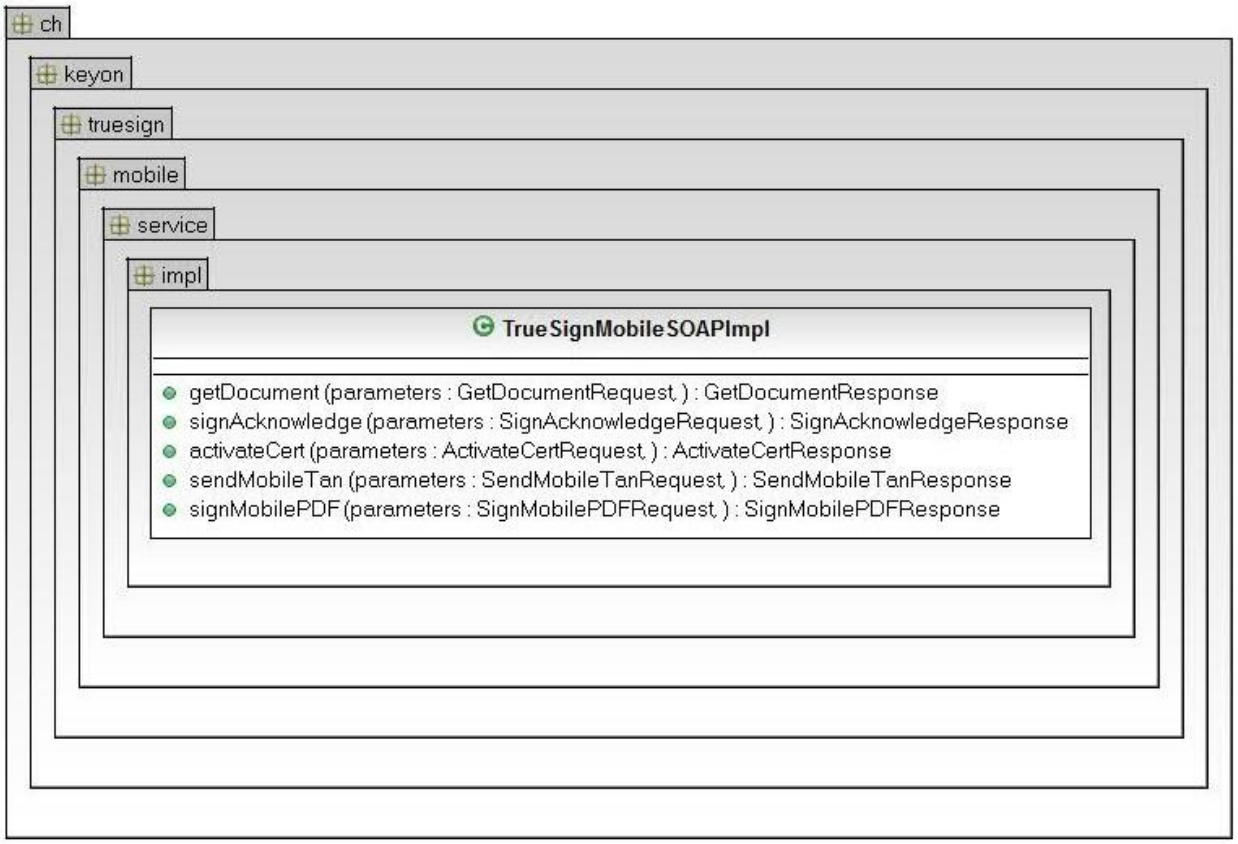


Abbildung 17 Klassendiagramm service.impl

Kernoperationen

Die Operationen der Implementation dienen lediglich als Dispatcher und leiten die SOAP-Requests an die RequestHandler Klasse weiter. Die Methoden erstellen lediglich die SOAP-Request und SOAP-Response-Objekte und übermitteln diese an den Service. Daher wird auf eine genaue Beschreibung der Operationen verzichtet. Diese kann im Kapitel [10.4.3.1 Package mobile] nachgelesen werden.

10.4.3.5. Package serviceWeb

In diesem Package befinden sich die mit JAX-WS generierten Klassen, welche den SOAP-Webservice Richtung trueSign Webportal bedienen.

10.4.3.6. Package tan

Dieses Package behandelt alles, was mit TANs und Sessions im Zusammenhang steht.

Klassendiagramm

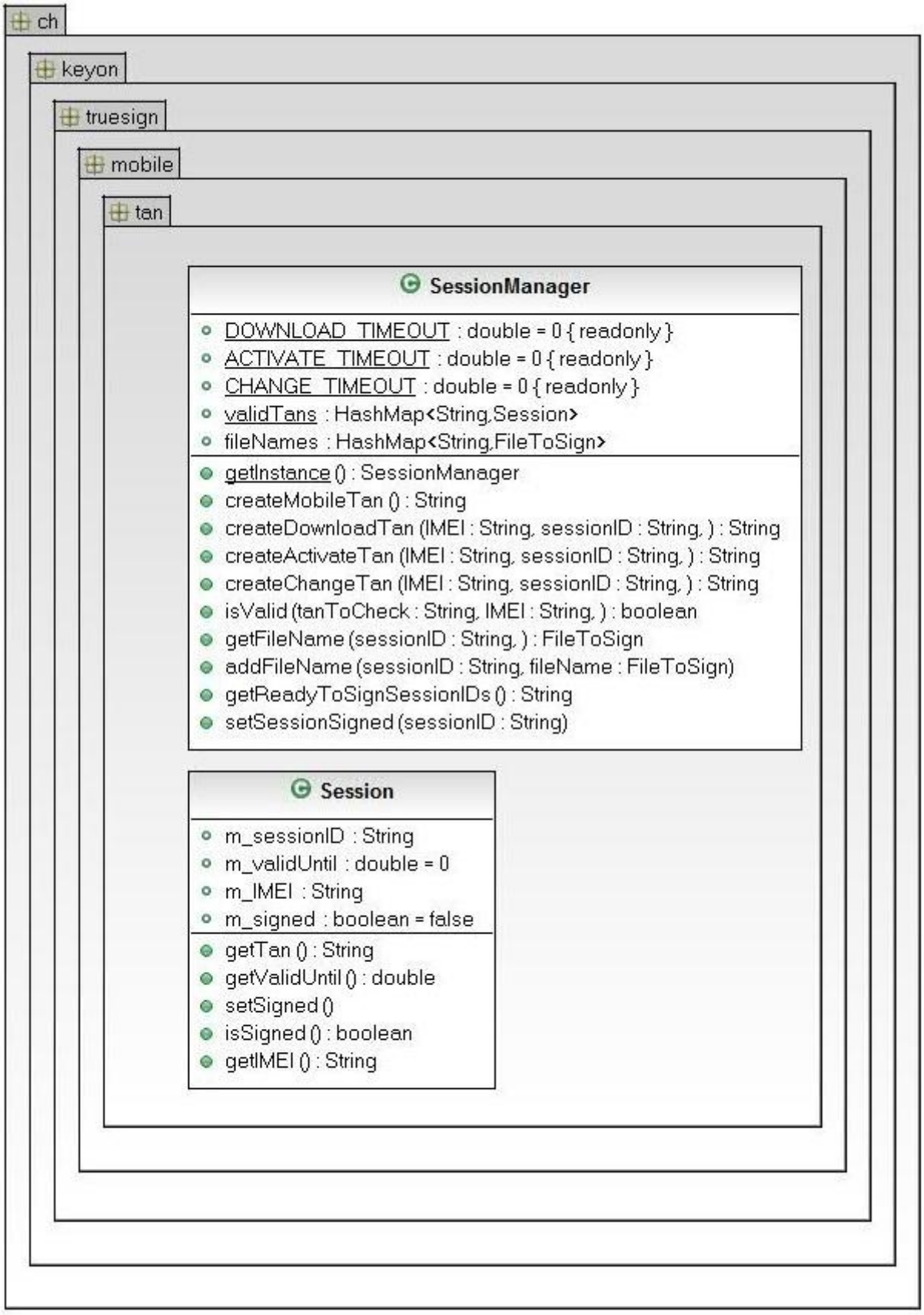


Abbildung 18 Klassendiagramm tan

Kernoperationen

isValid(String tanToCheck, String imei)
Diese Methode prüft ob, die übergebene TAN der IMEI zugeordnet ist und ob ihre Gültigkeit noch nicht überschritten ist. Sie retourniert je nach dem True beziehungsweise False

Create_Type_Tan(String name)
In den FactoryMethoden werden je nach Aufruf die vier verschiedenen TANs generiert. Diese unterscheiden sich lediglich in ihrer Gültigkeitsdauer. Die erstellten TANs werden in dieser Klasse zwischengespeichert, um sie anschliessend auf ihre Gültigkeit prüfen zu können. Generierte Mobile-TANs werden nicht gespeichert, da hier nur das trueSign Webportal die Gültigkeit prüfen muss.

getReadyToSignSessionIDs()
In dieser Methode wird ein String generiert, welcher via Push-Meldung an die trueSign MobileApp geschickt wird. Dieser String besteht aus allen SessionIDs der von diesem User hochgeladenen, noch nicht signierten Dateien.

addFileName(String sessionID, FileToSign fileName)
Alle eine zu signierende Datei betreffende Daten, werden im SessionManager zwischengespeichert. Ein neues File kann mit dieser Methode hinzugefügt werden.

getFileName(String sessionID)
Um wieder auf die zwischengespeicherten Daten zu einem File zugreifen zu können, wird diese Methode benötigt.

10.4.3.7. Package android.client

In diesem Package befindet sich die gesamte Android-Client-Applikation. Das Package selbst beherbergt hauptsächlich die Android-Activity-Klassen, sowie die übergeordneten Klassen Constants.java und SessionIDManager.java.

Klassendiagramm

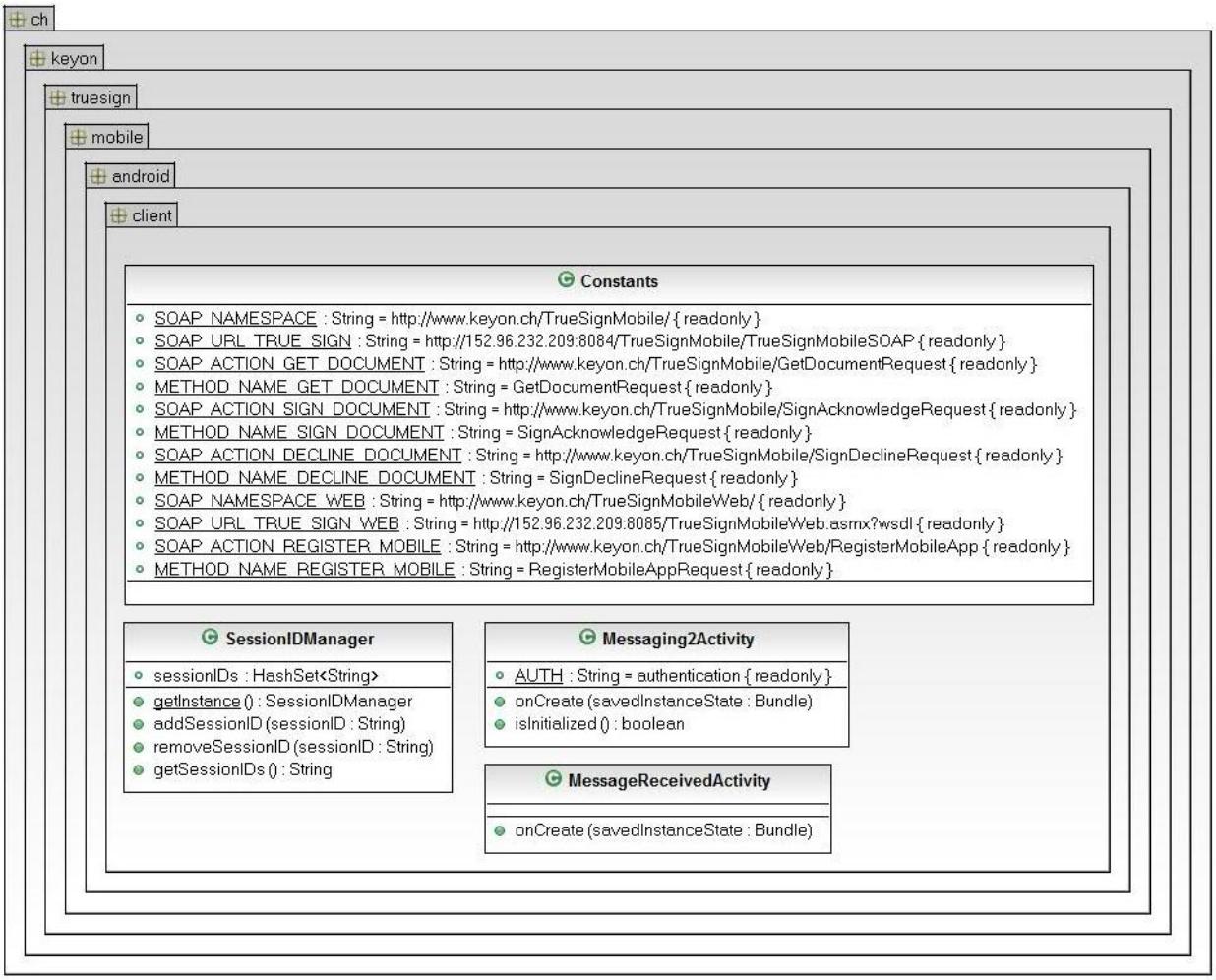


Abbildung 19 Klassendiagramm client

Kernoperationen

onCreate(Bundle savedInstanceState)
Die onCreate Methode der Activity ist der Haupteinstiegspunkt der trueSign Mobile-App. Sie prüft, ob die Applikation bereits initialisiert wurde. Falls ja, öffnet sie den Homescreen, falls nein, wird der Initialisierungs-Screen geöffnet.

addSessionID(String)
Mit dieser Methode kann eine SessionID zum SessionIDManager hinzugefügt werden. Dies geschieht immer dann, wenn eine neue Push-Meldung über ein zur Signatur bereitstehendes File eintrifft.

getSessionIDs()
Mit dieser Methode können alle aktuellen SessionIDs vom SessionIDManager abgeholt werden. Dies wird immer dann ausgeführt, wenn die Applikation gestartet wird oder der Homescreen neu geladen wird. Anschliessend werden alle Files, die diesen SessionIDs zugeordnet sind, vom Server abgeholt und in der Liste auf dem Homescreen zur Anzeige gebracht.

10.4.3.8. Package android.client.phonegap

Das phonegap Package enthält alle Plugins, welche für das Framework entwickelt wurden. Plugins sind native Klassen, im Fall von Android Java-Klassen, welche von der JavaScript-Implementation des PhoneGap-Frameworks aufgerufen werden können. Diese werden benötigt, damit die SOAP-Kommunikation mit dem trueSign Mobile Service abgewickelt und eine PDF-Datei angezeigt werden kann. Für die Kommunikation via SOAP wurde die kSoap-Library benutzt.

Klassendiagramm

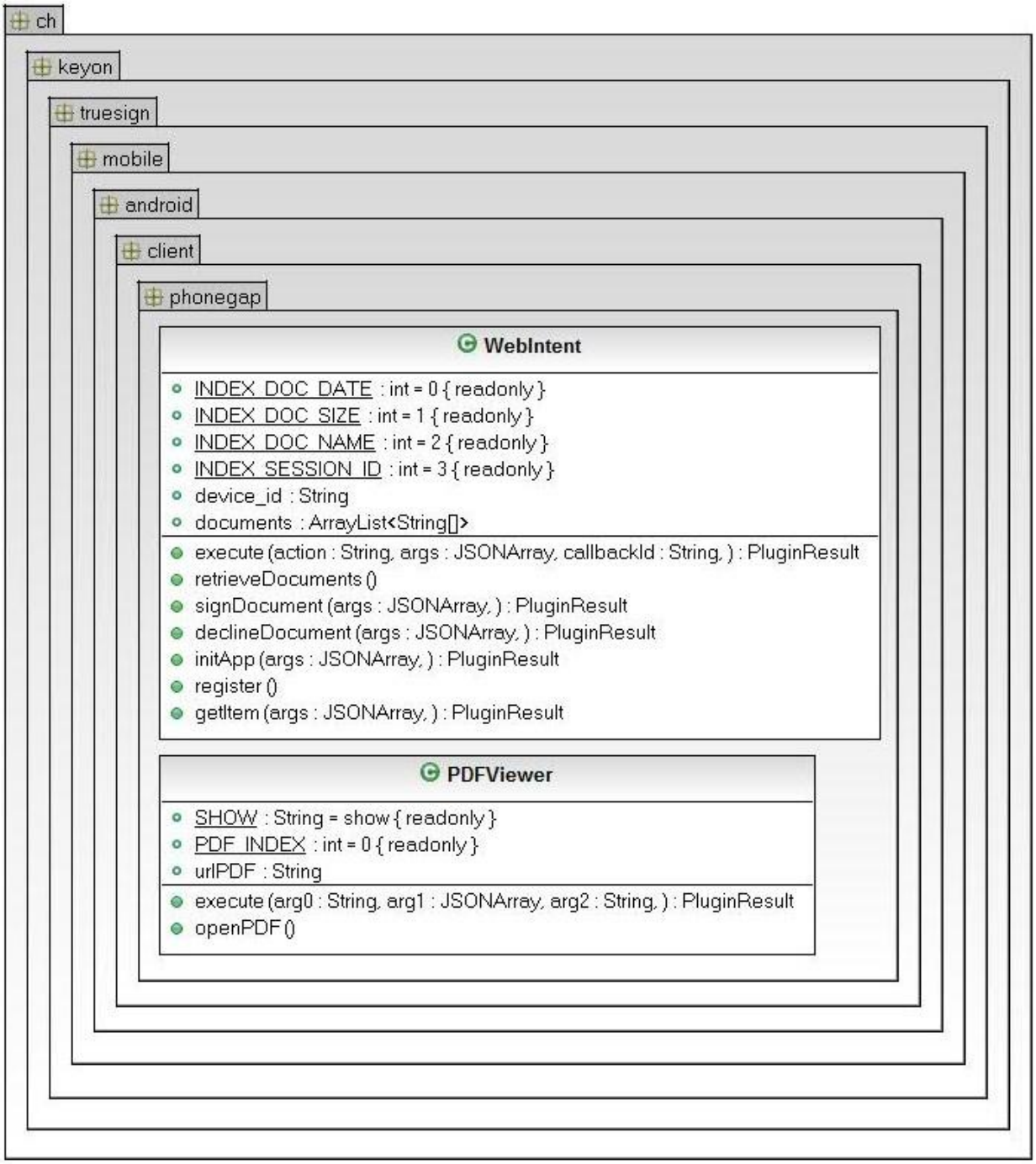


Abbildung 20 Klassendiagramm phonegap

Kernoperationen

execute(String action, JSONArray args, String callbackId)
Diese Methode muss jedes PhoneGap-Plugin implementieren. Sie dient als Entrypoint für die JavaScript-Aufrufe. Die Action wird ausgewertet und die dementsprechende Aktion ausgeführt. Der args-Parameter wird an die ausführende Methode übergeben, da er die nötigen Daten enthält. Mit der

callbackId könnte noch eine Callback-Methode bekannt gegeben werden. Dies wurde in unserer Arbeit allerdings nicht verwendet.

register()

In dieser Methode wird das Smartphone beziehungsweise die Applikation beim Push Notification Provider (Google / C2DM) registriert. Diese wird aufgerufen, falls die Applikation noch keine RegistrierungsID gespeichert hat, die sie bei der Registrierung erhält.

initApp(JSONArray args)

Die initApp Methode führt die Initialisation aus. Dies muss beim ersten Start der Applikation gemacht werden. Dabei werden der Username sowie die Initialisierungs-PIN, die vom Benutzer eingegeben werden, gespeichert.

getItem(JSONArray args)

Diese Methode retourniert ein File, welches zur Signatur bereit ist. Wobei nicht die Datei im eigentlichen Sinne sondern viel mehr die dazugehörigen Meta-Daten zurückgegeben werden. Also der Name der Datei, die Grösse, das Datum und die entsprechende SessionID. Diese Daten werden dann durch das PhoneGap-Framework im GUI der Applikation angezeigt.

retrieveDocuments()

In dieser Methode werden die zu den vom SessionIDManager erhaltenen SessionIDs gehörenden Dateien vom trueSign Mobile Service, sowie deren Meta-Daten abgeholt und lokal zwischen gespeichert.

signDocument(JSONArray)

Die eigentliche Signatur wird beim trueSign Mobile Service in Auftrag gegeben.

declineDocument(JSONArray)

Die Signatur wird verworfen und dies dem trueSign Mobile Service mitgeteilt. Dieser löscht daraufhin die Datei.

10.4.3.9. Package android.client.push

Dieses Package beinhaltet die für das Empfangen von Push-Nachrichten nötigen Klassen.

Klassendiagramm

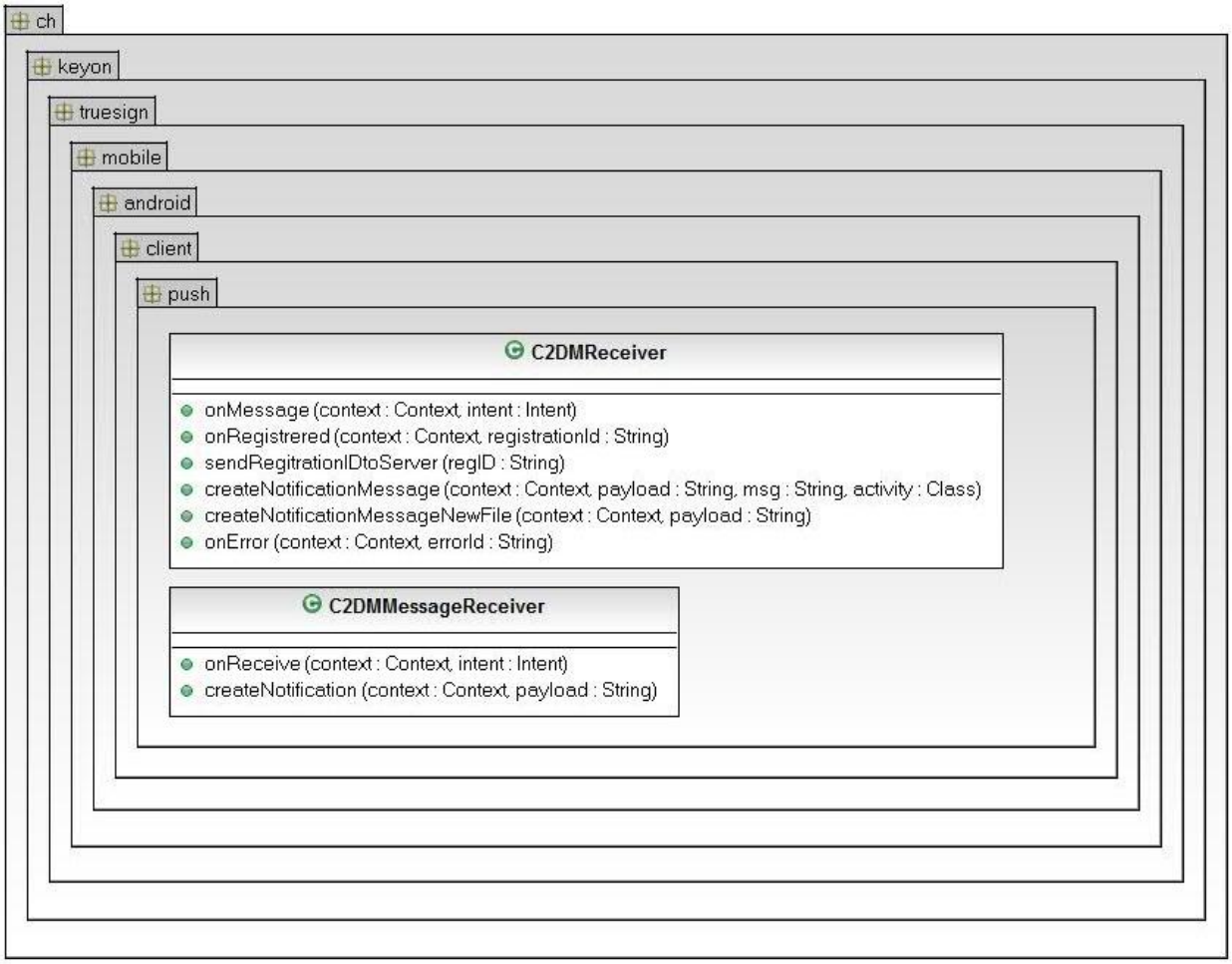


Abbildung 21 Klassendiagramm push

Kernoperationen

onRegistrered(Context context, String registrationId)
Diese Methode wird aufgerufen, sobald vom Push Notification Provider die Registrierung abgeschlossen ist, und der Applikation ihre RegistrierungsID mitgeteilt wird.

sendRegitrationIDtoServer(String regID)
In dieser Methode wird die soeben erhaltene RegistrationID dem trueSign Webportal via SOAP mitgeteilt. Anschliessend kann der trueSign Mobile Service die Applikation mit Push-Nachrichten erreichen, mittels der RegistrationID.

createNotificationMessage(Context context, String payload)
Diese Methode erzeugt eine für den Benutzer sichtbare Mitteilung, welche ihn über die erhaltene Push-Meldung informiert.

10.4.3.10. Package com.google.android.c2dm

In diesem Package befinden sich die von Google zur Verfügung gestellten Basisklassen, welche für das Empfangen von Push-Nachrichten nötig sind. Sie werden hier nicht genauer beschrieben.

Klassendiagramm

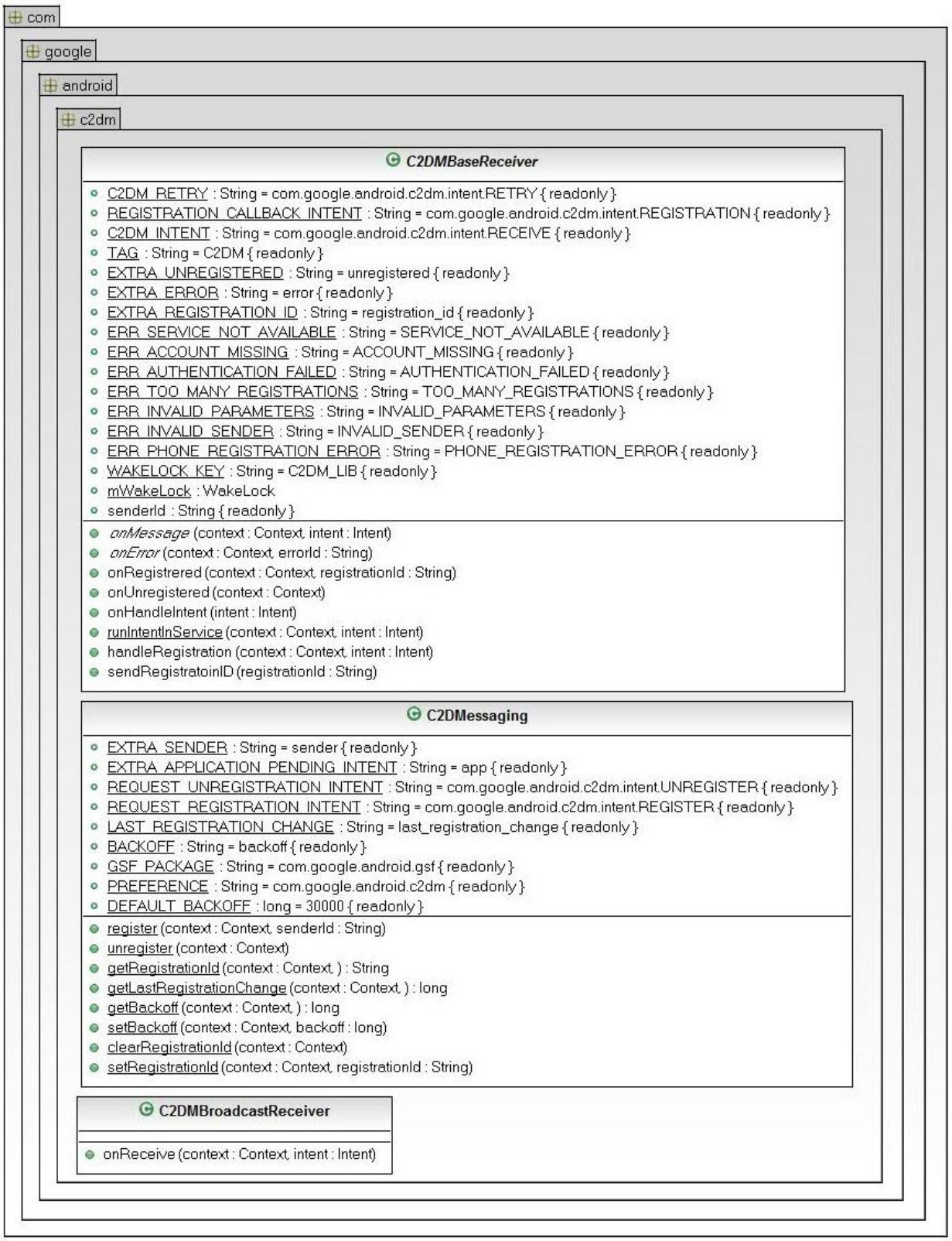


Abbildung 22 Klassendiagramm c2dm

10.4.4. .Net Klassen

In diesem Kapitel sollen die neu entwickelten Klassen des trueSign Webportal beschrieben werden. Diese sind in C-Sharp implementiert, zudem handelt es sich dabei häufig um sogenannten Code-Behind der jeweiligen ASPX Webseiten. Da diese Klassen Hauptsächlich aus Standard ASP Methoden – wie Page_Load und Page_Init – bestehen, wird hier auf Klassendiagramme verzichtet und nur die Intressanten Kernoperationen beschrieben.

10.4.4.1. Datenbankverbindung

Die Datenbankverbindung wird in der ganzen Applikation mit dem .Net Entity-Framework behandelt. Dieses ermöglicht eine einfache Handhabung und nimmt dem Entwickler die sonst langwierige Schreibarbeit bei der Entwicklung von DOA Klassen ab.

10.4.4.2. Login.aspx.cs

Diese Klasse behandelt den Login zur Webapplikation.

Kernoperationen

ButtonLogin_Click(object sender, EventArgs e)
Diese Methode behandelt den Login Prozess, sie Sucht den eingegebenen Username in der Datenbank. Wird ein User gefunden wird Das Passwort überprüft und bei Korrektheit wird der User zur Eingabe der TAN weitergeleitet.

10.4.4.3. EnterTan.aspx.cs

Hier wurde das Versenden der MobileTan implementiert.

LoadPage()
In dieser Methode wird die MobileID des Benutzers aus der Datenbank ausgelesen, anschliessend wird diese via SignaturControlFacade an den trueSign Mobile Service geschickt. Die erhaltene Mobile TAN wird im WebSessionControl abgelegt.

ButtonLogin_Click(object sender, EventArgs e)
Hier wird der eigentliche Login Prozess bearbeitet. Für die überprüfung der MobileTan wird ein CustomValidator verwendet, dies geschieht daher vor dem Aufruf dieser Methode. Es muss deshalb nur noch die Session aufgebaut werden und das AccessToken mit den entsprechenden Berechtigungen erstellt werden. Anschliessend wird der Benutzer zur Hauptseite weitergeleitet.

10.4.4.4. CodeSigning.aspx.cs

Die meisten der Operationen in dieser Klassen fungieren als reine Click-Listener und leiten die Aufgaben nur an die SignatureControlFacade weiter. Deshalb werden diese hier nicht genauer beschrieben.

SignButton_Click(object sender, EventArgs e)
Diese Methode löst den Signaturprozess aus. Sie sucht in der Datenbank nach den entsprechenden Userdaten – wie dem Alias wie auch der IMEI – und leitet diese an die SignatureControlFacade weiter.

10.4.4.5. SignedFiles.aspx.cs

Auf der SignedFiles Page können die signierten Dateien verwaltet werden. Die Hierfür benötigten Operationen werden hier kurz erläutert.

downloadFile(int index)
Dank des übergebenen index kann aus der Tabelle die Zeile ausgelesen werden, in welcher sich das zum Download angeforderte File befindet. Darin enthalten – jedoch dem User nicht angezeigt – ist die SessionID der Datei. Anschliessend kann die Datei gelsen und zum Download bereit gestellt werden.

deleteFile(int index)
Das Löschen einer Datei funktionert sehr ähnlich wie der Download. Mittels index wird ermittelt um welches File es sich handelt, anschliessend können die gespeicherten Metadaten aus der Datenbank entfernt und die Datei gelöscht werden.

10.4.4.6. ChangePassword.aspx.cs

Hier kann das trueSign Webportal Passwort geändert werden. Das Passwort wird in der Datenbank nur als Hashwert gespeichert. Folgend wird die Operation zur Änderung des Passwortes kurz beschrieben.

ButtonSave_Click(int index)
In einem ersten Schritt werden die Userdaten des Benutzers aus der Datenbank gelesen. Anschliessend wird aus dem eingegebenen Passwort und dem in der Datenbank gespeicherten Salt ein Hashwert berechnet. Stimmt dieser mit dem gespeicherten Hash überein, darf das Passwort geändert werden. Somit wird ein neuer Salt erstellt und mit diesem wiederum ein Hashwert des neuen Passwortes berechnet. Zum Schluss wird beides – der Hash sowie der Salt – in der Datenbank abgelegt.

10.5. Datenspeicherung

10.5.1. Persistente Datenhaltung

Die Datenhaltung wird komplett beim trueSign Webportal realisiert. Beim trueSign JCE Service sollen nur temporäre Daten, wie zum Beispiel die benötigten TANs, gespeichert. Dies wird später beschrieben. Die persistenten Daten werden alle zentral beim trueSign Webportal gespeichert. Folgend soll im Datenbankmodel eine Übersicht der Datenhaltung gegeben werden.

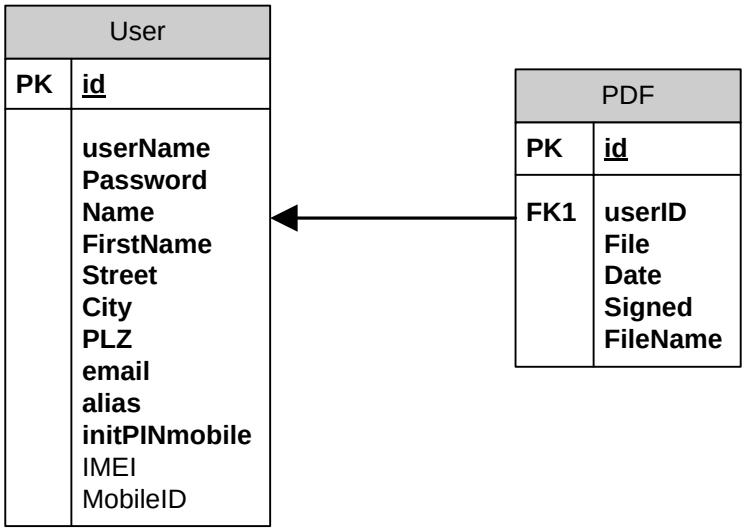


Abbildung 23 Datenbankmodel

10.5.2. Temporäre Datenhaltung

Folgende Tabelle zeigt, wo temporär benötigte Daten abgelegt werden.

Was		Wo
ActivateTAN		trueSignMobile Service
DownloadTAN		trueSignMobile Service
LoginTAN		trueSign Webportal
FileToSign	FileName	trueSignMobile Service
	UserID	trueSignMobile Service
	SessionID	trueSignMobile Service
	alias	trueSignMobile Service

11. Systemtest

11.1. Voraussetzungen

Die Applikation muss vollständig und korrekt aufgesetzt werden, um einen Systemtest durchführen zu können. Hierfür muss das Webportal auf einem IIS-Webserver installiert werden. Der trueSign Mobile Service sowie der trueSign JCE Service müssen auf einem Tomcat Application Server deployed werden. Es können beide Services auf derselben Tomcat-Instanz in Betrieb genommen werden. Der IIS-Webserver beziehungsweise das Webportal muss eine Verbindung zum trueSign Mobile Service aufbauen können. Der trueSign Mobile Service wiederum, muss mit dem trueSign JCE Service kommunizieren können.

Die trueSign Mobile-App muss auf einem Smartphone mit mindestens Android 4.0 installiert werden. Zudem muss die mobile Applikation mit dem trueSign Mobile Service, welcher auf dem Tomcat installiert wurde, Daten austauschen können.

11.2. Vorbereitung

Um für eine saubere Testbasis zu sorgen, muss die Datenbank des trueSign Webportals gesäubert werden. Sie sollte keine Files enthalten, jedoch einen User mit einem Alias. Dieser User muss ebenfalls im trueSign JCE Service konfiguriert sein, damit die Signatur korrekt ausgeführt werden kann.

Die Reihenfolge der Tests ist einzuhalten, da die meisten Testfälle von vorangegangenen Fällen abhängig sind. Zum Beispiel kann in der trueSign Mobile-App keine Signatur bestätigt werden, wenn nicht zuvor ein File im Webportal hochgeladen wurde.

11.3. Test vom 13.06.2012

Device	Testbeschreibung	Erwartetes Resultat	Resultat	Bemerkung
Webportal Login	Login	Weiterleitung zur Eingabe der TAN	OK	
Webportal Login	Login mit falschen Userdaten	Login wird verweigert	OK	
Mobile-App Login	Login beim Webportal	Push-Meldung mit TAN wird angezeigt	OK	
Webportal TAN	Eingabe der TAN	Weiterleitung zur Signatur Webseite	OK	
Webportal TAN	Eingabe einer falschen TAN	Keine Weiterleitung, Fehlermeldung	OK	
Webportal Signed Files	Klick auf Signed Files	Anzeige von leerer Seite	OK	Da noch keine Files signiert wurden.
Webportal My Account	Klick auf My Account	Anzeige der Benutzerdaten	OK	
Webportal Revoke Cert	Klick auf Revoke Certificate	Anzeige der Revoke Cert Seite	OK	
Webportal Change PW	Change Webportal Password	Anzeige der Webportal Seite	OK	
Webportal Change PW	Eingabe eines neuen Passwortes	Passwort wird geändert	OK	
Webportal Change PW	Eingabe eines falschen alten Passwortes	Fehlermeldung Passwort wird nicht geändert	OK	

Device	Testbeschreibung	Erwartetes Resultat	Resultat	Bemerkung
Webportal Change PW	Eingabe zweier unterschiedlicher neuer Passwörter	Fehlermeldung Passwort wird nicht geändert	OK	
Webportal Logout	Klick auf Logout	Weiterleitung auf Login Seite	OK	
Webportal Logout	Nach Logout auf Back drücken	Fehlermeldung	OK	Back geht, da aus Browser Cash geholt wird. Jeder weitere Klick endet jedoch in einem Fehler
Webportal Sign File	Ein File auswählen und auf Sign klicken	Push-Meldung ans Smartphone	OK	
Mobile-App Sign File	File im Webportal hochgeladen	Push-Meldung wird angezeigt	OK	
Mobile-App Sign File	Klick auf Push-Meldung	App wird geöffnet und der Filename wird in der Liste angezeigt	OK	Datum hat falsches Format
Mobile-App Sign File	Klick auf Datei	Die Datei wird angezeigt	OK	
Mobile-App Sign File	Klick auf Sign File	Dialog zur Eingabe der PIN für den Private Key wird angezeigt	OK	
Mobile-App Sign File	Eingabe der PIN für den Private Key und Klick auf Sign	Meldung File wurde signiert	OK	
Mobile-App Sign File	Anzeige des Homescreens	Signiertes File wird in Liste grün markiert	OK	
Mobile-App Sign File	Erneutes Klicken auf Sign File	Meldung, dass Datei bereits signiert ist	OK	
Webportal Signed Files	Klick auf Signed File im Menü	Signierte Datei wird in Liste angezeigt	OK	
Webportal Signed Files	Klick auf Download des signierten Files	Download Dialog des Browsers wird gestartet	OK	
Webportal Signed Files	Download des Files	Das File kann angezeigt werden, Signatur ersichtlich	OK	
Webportal Signed Files	Klick auf Delete des signierten Files	Das File wird gelöscht und nicht mehr in der Liste angezeigt	OK	Evtl. „are you sure“ Nachfrage noch einfügen
Webpotal Sign File	Erneut ein File auswählen und Klick auf Sign	File wird hochgeladen Push-Meldung wird angezeigt	OK	
Mobile-App Sign File	Klick auf Push-Meldung	App wird neu geladen, bereits signiertes File fällt aus Liste, neues wird angezeigt	OK	
Mobile-App Sign File	Klick auf Sign File	Dialog zur Eingabe der PIN für den Private Key wird angezeigt	OK	

Device	Testbeschreibung	Erwartetes Resultat	Resultat	Bemerkung
Mobie-App Sign File	Eingabe eines falschen PIN für den Private Key	Fehlermeldung File wird nicht signiert	OK	Fehlermeldung könnte spezifischer sein, nicht klar, dass PIN falsch
Mobile-App Decline Sig	Klick auf Decline	Dialog zu Bestätigung des Decline wird angezeigt	OK	
Mobile-App Decline Sig	Klick auf Decline	Signatur wird verworfen, Homescreen wird angezeigt, file ist rot eingefärbt	OK	
Mobile-App Decline Sig	Erneutes Klicken auf Sign	Meldung, dass File bereits declined wurde	OK	

12. Installationsanleitung

12.1. trueSign Webportal

12.1.1. Voraussetzungen

Für die Installation des trueSign Webportal wird ein IIS 7.0 Webserver mit einem Korrekten SLL Server Zertifikat benötigt. Zudem muss von diesem IIS aus eine Datenbank-Instanz, wenn möglich MS SQL Express, erreichbar sein.

12.1.2. Vorgehen

1. Erstellen Sie die Datenbank mittels des SQL-Skriptes
2. Erstellen Sie innerhalb des IIS einen neuen Application Pool.
    - Achten Sie dabei speziell auf die verwendete Identity, diese muss Zugriff auf das Root Verzeichnis der Webapplikation sowie auf die Datenbank haben
    - Die .NET Framework Version muss auf v4.0 eingestellt werden.

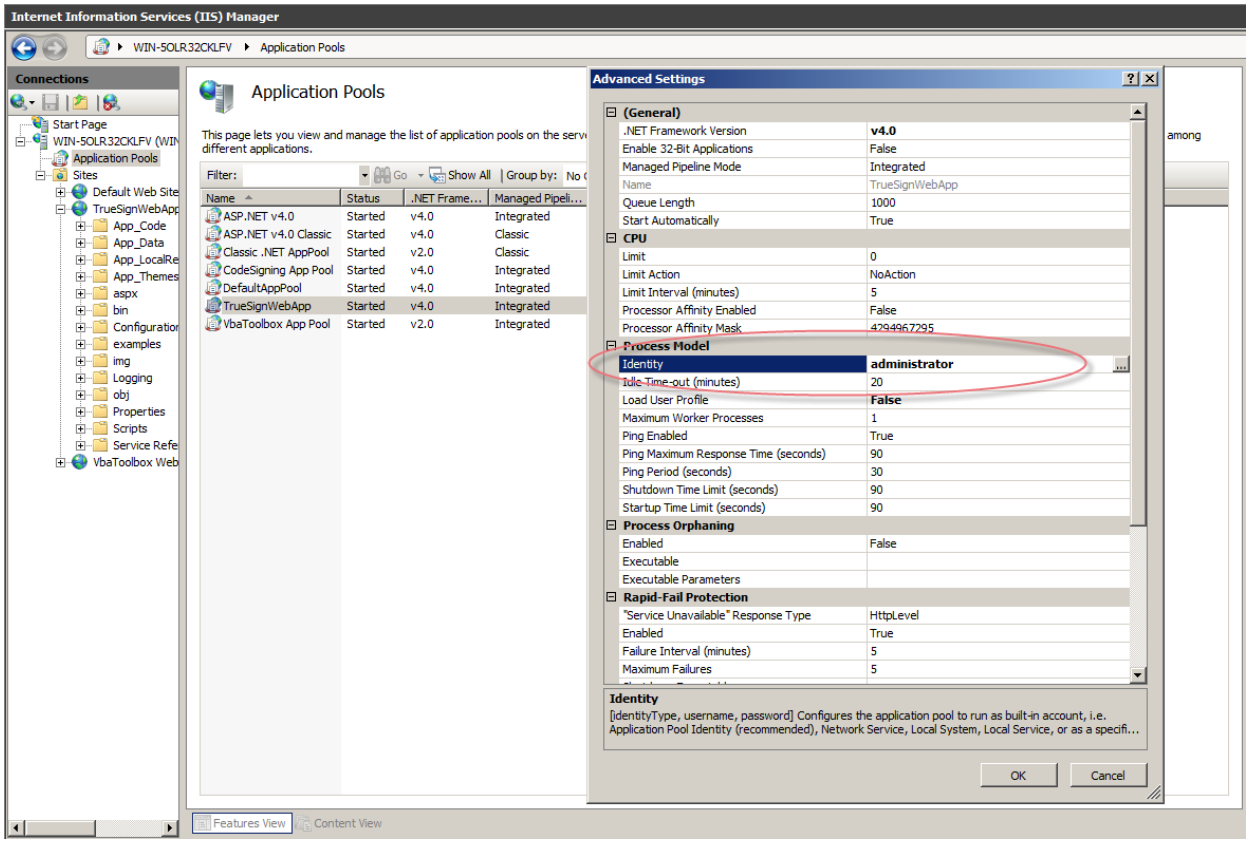


Abbildung 24 Erstellen eines IIS Application Pools

3. Erstellen Sie eine neue Site innerhalb des IIS-Webservers
   - Selektieren Sie dabei den vorher erstellten Application pool
   - Verwenden Sie bei Binding den Type https
   - Wählen sie unter SSL certificate das richtige Server Zertifikat aus.

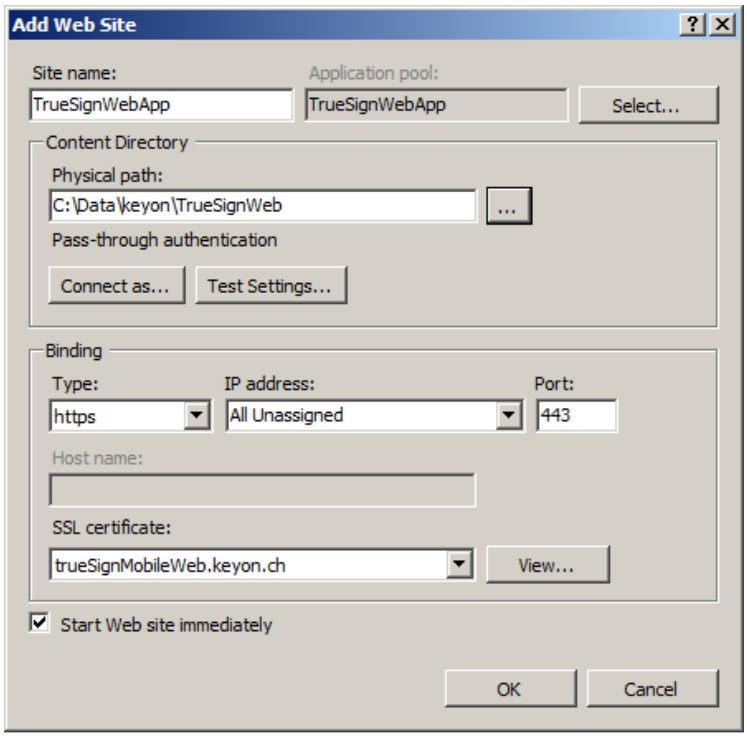


Abbildung 25 Erstellen einer IIS Web Site

12.1.3. Konfiguration

Das trueSign Webportal wird über das sich im Root befindliche web.config File konfiguriert. Folgend werden die wichtigsten Punkte daraus kurz erläutert:

12.1.3.1. Datenbank

Die Datenbank wird mit folgenden Parametern innerhalb des <configuration> Tags konfiguriert

```
<connectionStrings>
  <add name="ApplicationServices" connectionString="data source=.\SQLEXPRESS;Integrated
    Security=SSPI;AttachDBFilename=|DataDirectory|\aspnetdb.mdf;User Instance=true"
    providerName="System.Data.SqlClient" />
  <add name="TrueSignMobileDbEntities"
    connectionString="metadata=res://*/TrueSignMobileModel.csdl|res://*/TrueSignMobileModel.ssdl|re
    s://*/TrueSignMobileModel.msl;provider=System.Data.SqlClient;provider connection
    string=&amp;quot;Data
    Source=.\SQLEXPRESS;AttachDbFilename=|DataDirectory|\TrueSignMobileDb.mdf;Integrated
    Security=True;User Instance=True;MultipleActiveResultSets=True&amp;quot;"
    providerName="System.Data.EntityClient" />
</connectionStrings>
```

Die verbindung zum trueSign Mobile Service kann mit den folgenden Parametern konfiguriert werden:

```
<services>
    <service name="TrueSignMobileWeb">
        <endpoint address="" binding="basicHttpBinding"
        bindingNamespace="http://www.keyon.ch/TrueSignMobileWeb"
        contract="TrueSignMobileWebInterfaces" />
    </service>
```

```
</services>
<bindings>
  <basicHttpBinding>
      <binding name="TrueSignSOAPMobile" closeTimeout="00:01:00" openTimeout="00:01:00"
      receiveTimeout="00:10:00" sendTimeout="00:01:00" allowCookies="false"
      bypassProxyOnLocal="false" hostNameComparisonMode="StrongWildcard"
      maxBufferSize="1572864" maxBufferPoolSize="524288" maxReceivedMessageSize="1572864"
      messageEncoding="Text" textEncoding="utf-8" transferMode="Buffered"
      useDefaultWebProxy="true">
    <readerQuotas maxDepth="32" maxStringContentLength="8192" maxArrayLength="16384"
    maxBytesPerRead="4096" maxNameTableCharCount="16384" />
    <security mode="Transport">
      <transport clientCredentialType="Certificate" />
    </security>
   </binding>
  </basicHttpBinding>
</bindings>
<client>
      <endpoint address="http://localhost:8080/TrueSignMobile/TrueSignMobileSOAP"
      binding="basicHttpBinding" bindingConfiguration="TrueSignSOAPMobile"
      contract="TrueSignReference.TrueSignMobile" name="TrueSignMobileSOAP" />
</client>
```

Wichtig dabei ist, dass die Verbindung zum trueSign Mobile Service Korrekt configuriert wird, sowie auch die SSL Einstellungen korrekt sind

12.2. trueSign Mobile Service

12.2.1. Voraussetzungen

Der trueSign Mobile Service benötigt eine Apache Tomcat Installation, zudem wird eine komplette trueSign JCE Service Installation vorausgesetzt. Da dieser Service bereits von der Firma Keyon AG entwickelt wurde, und nicht Teil dieser Arbeit ist, wird er hier nicht genauer beschrieben.

12.2.1.1. Installation

Der trueSign Mobile Service kann auf einfache Weise im Apache Tomcat deployed werden. Dazu muss jedlich das trueSignMobile.war File, oder der bereits entpacke Ordner der Web Applikation in das Webapps Verzeichnis des Tomcat kopiert werden.

13. Benutzeranleitung

13.1. Übersicht über die Applikation

Die Applikation besteht aus zwei Teilen: einerseits aus dem trueSign Webportal und andererseits aus der trueSign Mobile-App. Das Signieren eines PDF Dokumentes erfolgt über das Webportal. Um die Zwei-Faktor-Authentisierung zu ermöglichen, wird die Mobile-App eingesetzt. Das Smartphone ersetzt somit gewissermassen das, in anderen Signatur-Anwendungen übliche, Hardware-Token.

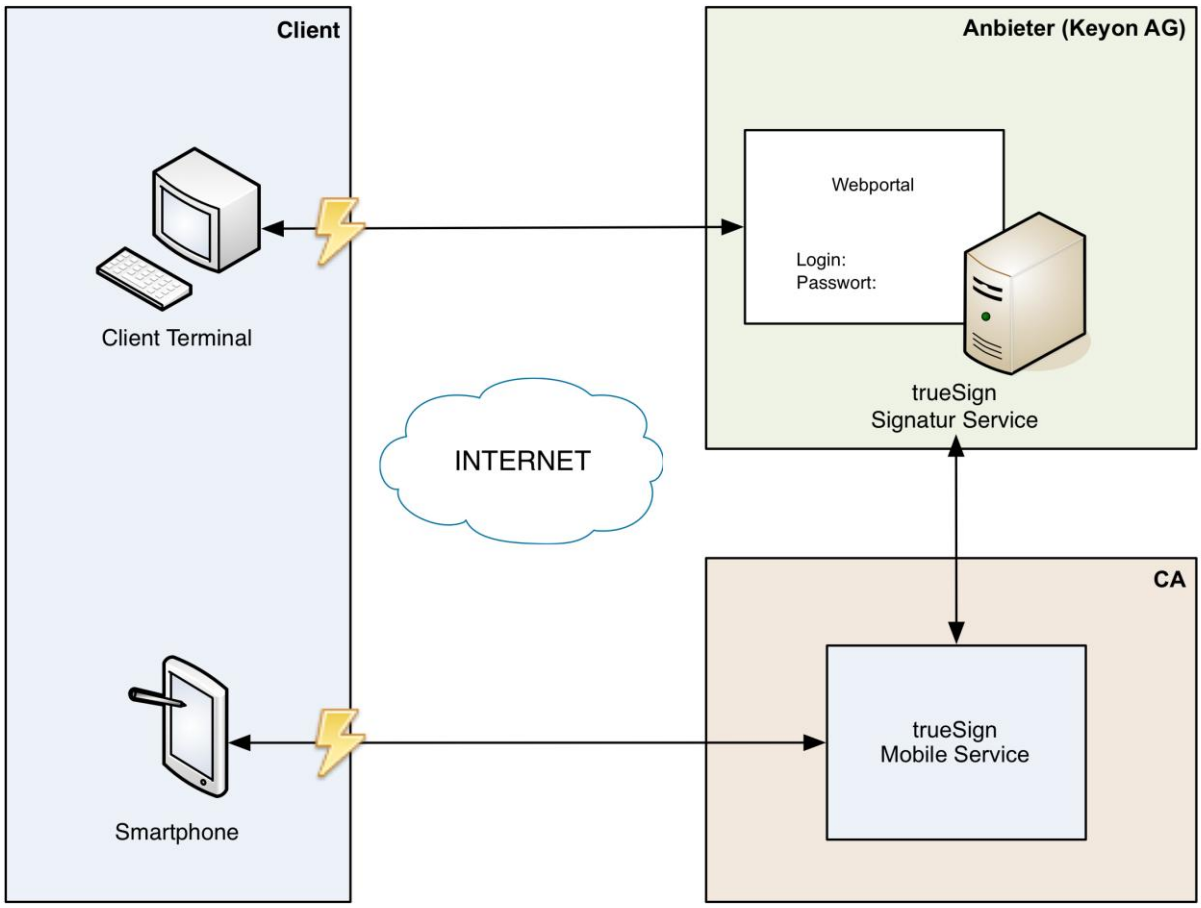


Abbildung 26 Übersicht über die Applikation

13.2. Installation

Die trueSign Mobile-App kann im App Store oder auf der Homepage des Anbieters bezogen werden und muss anschliessend auf dem Smartphone installiert werden.

13.3. Registrierung

Nach dem Sie sich Online Registriert haben und der Anbieter des Signatur-Dienstes ihre Aufschaltgebühr erhalten hat. Werden Ihnen die weiteren Schritte in einem Dokument, welches sie per Post erhalten erläutert. Es ist für zwingend nötig, dass Sie bei einer Registration Authority persönlich Vorsprechen und einen amtlichen Ausweis verlegen. Ist diese Überprüfung der Identität abgeschlossen, werden Sie in einem weiteren Brief die Anweisungen für die Initialisierung der trueSign Mobile-App sowie ihres Zertifikates erhalten. Danach können Sie mit dem trueSign Signatur Service ihre Dokumente signieren.

13.4. Benutzerhandbuch

13.4.1. Initialisierung

Damit die Mobile-App verwendet werden kann, müssen Sie beim ersten Starten der Applikation folgende Schritte durchführen:

- Nach dem Start erscheint ein Fenster, in welchem Sie aufgefordert werden, Ihren Benutzernamen und Ihr Initialisierungspasswort einzugeben. Dieses wurde Ihnen per Post zugestellt.
- Auf Ihrem Smartphone erscheint nun die Meldung, dass Sie sich im Webportal anmelden müssen, um Ihr Zertifikat zu aktivieren.
- Sobald Sie sich im Webportal erfolgreich eingeloggt haben, erhalten Sie auf Ihrem Smartphone eine Nachricht, welche Ihre Initialisierungs-PIN enthält. Drücken Sie auf die Benachrichtigung und ein PIN-Eingabe-Dialog erscheint. Geben Sie die PIN im entsprechenden Feld ein.
- Klicken Sie, nachdem Sie unter «*Neue PIN eingeben*» und «*Neue PIN bestätigen*» eine neue PIN gewählt haben, auf «*OK*» und Sie gelangen ins Hauptmenü der Applikation.

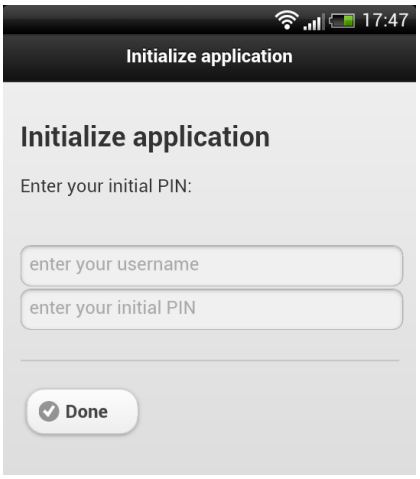


Abbildung 27 trueSign Mobile-App Initialize

13.4.2. Signieren von Dokumenten

Nach erfolgreicher Initialisierung der Mobile-App, steht dem Signieren der Dokumente nichts mehr im Weg. Loggen Sie sich im Webportal mit Ihrem Benutzernamen und Ihrem Passwort ein und geben Sie danach die auf Ihr Smartphone gesendete TAN im entsprechenden Feld ein.

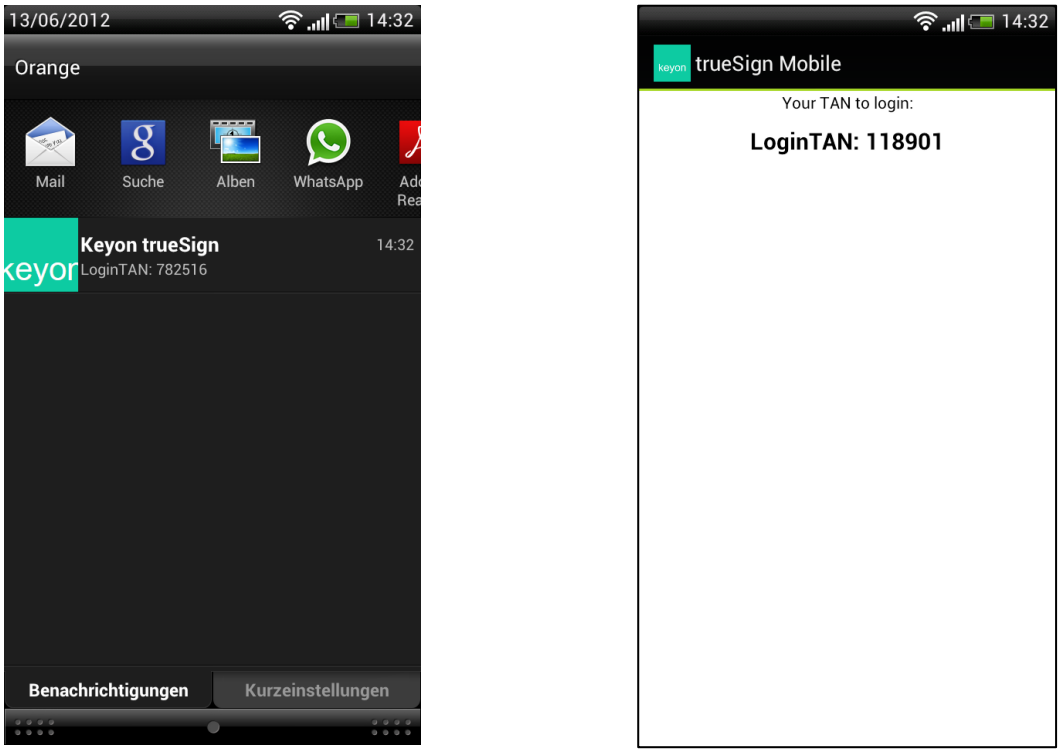


Abbildung 28 trueSign Mobile-App Tan

13.4.2.1. Dokument hochladen

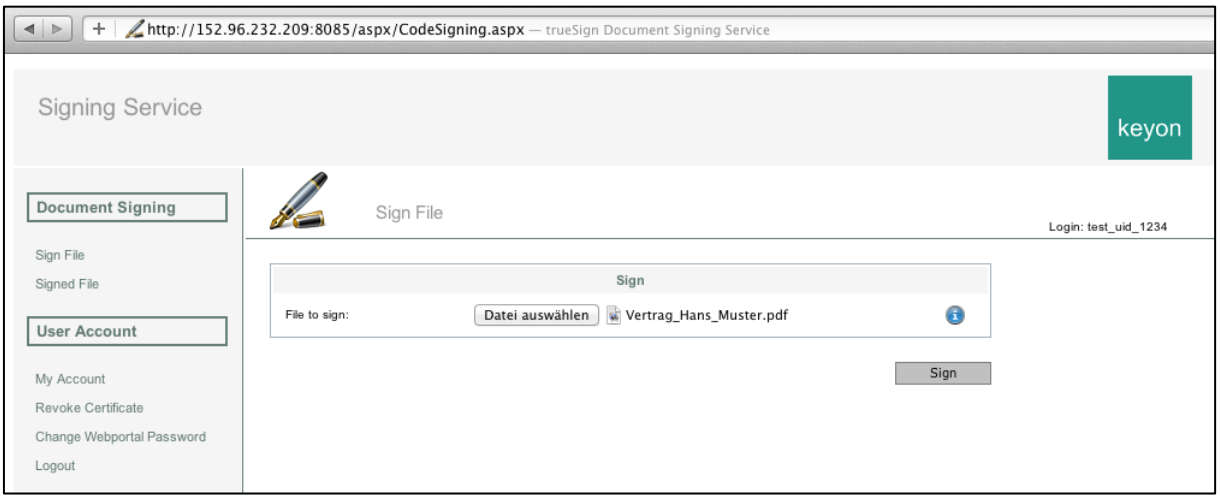


Abbildung 29 trueSign Webportal Sign File

Falls Sie sich nicht auf der Hauptseite befinden, auf die Sie gelangen, nachdem Sie sich eingeloggt haben, klicken Sie in der Navigation auf der linken Seite auf *«Sign File»*.

Drücken Sie die Schaltfläche *«Datei auswählen»* und wählen Sie das PDF-Dokument aus Ihren Daten aus, welches Sie signieren möchten. Nun wird die Datei angezeigt. Klicken Sie auf «*Sign*» und Sie erhalten kurz darauf eine Benachrichtigung auf Ihrem Smartphone, dass eine neue Datei zum Signieren bereitsteht.

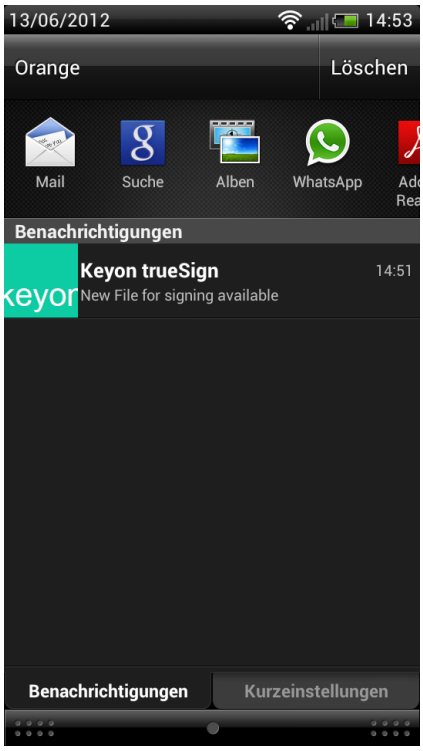


Abbildung 30 trueSign Mobile-App Push New File

13.4.2.2. Signieren mit der Mobile-App

Starten Sie nun die App auf Ihrem Smartphone, indem Sie auf diese Benachrichtigung drücken. Das Dokument (oder die Dokumente, falls Sie mehrere hochgeladen haben) wird nun im Hauptmenü angezeigt.

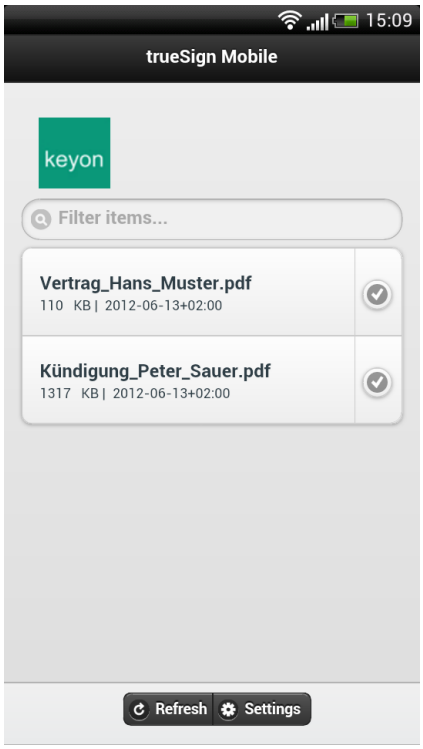


Abbildung 31 trueSign Mobile-App Homescreen

Neben dem Dokumentnamen, sind auch Informationen wie Dateigrösse und das Datum der Dokumente ersichtlich. Klicken Sie nun auf einen Eintrag in der Liste und öffnen Sie so das PDF auf Ihrem Smartphone mit Ihrem Standard-PDF-Viewer, welcher auf Ihrem Gerät installiert ist.

Hinweis: *Befinden sich mehrere PDF-Viewer auf Ihrem Smartphone, erscheint eine Liste der Applikationen, in der Sie Ihren bevorzugten Viewer wählen und diesen auch als Standard-Viewer speichern können.*

Um mit dem Signaturprozess fortzufahren, können Sie mit dem Zurück-Button Ihres Smartphone wieder ins Hauptmenü gelangen und auf der rechten Seite des Dokuments auf die Signatur-Schaltfläche mit dem Checksymbol drücken. Es öffnet sich ein Fenster, in welchem Sie aufgefordert werden, Ihre PIN, welchen Sie bei der Initialisierung geändert haben, einzugeben und mit *«Sign»* zu bestätigen. Das Ablehnen (Decline) der Signatur wird weiter unten in einem eigenen Kapitel beschrieben. Aus Sicherheitsgründen haben Sie vier Versuche Ihre PIN korrekt einzugeben, bevor dieser Prozess gesperrt wird.

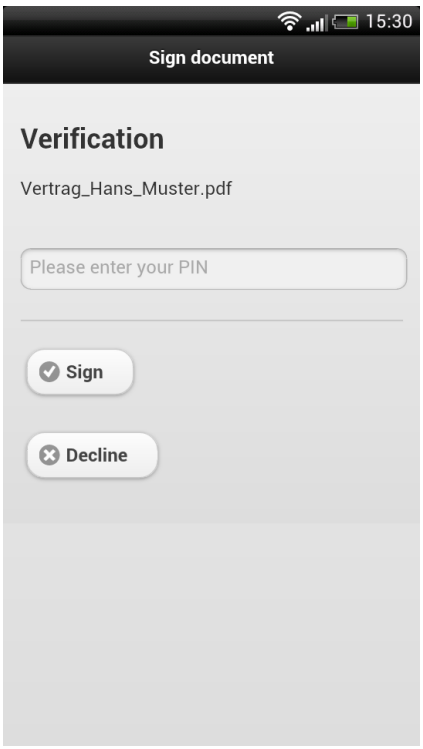


Abbildung 32 trueSign Mobile-App Verification

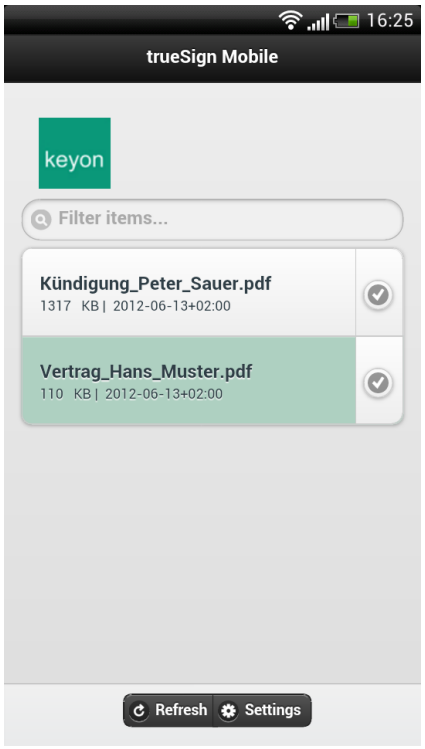


Abbildung 33 trueSign Mobile-App Signed File

Nach erfolgreichem Signieren wird im Hauptmenü der Eintrag mit dem signierten Dokument grün hinterlegt.

13.4.2.3. Downloaden des signierten Dokuments

Wechseln Sie nun wieder ins Webportal. Dort können Sie die signierten Dokumente unter *«Signed File»* herunterladen und weiter verwenden. Werden die Dokumente im Webportal nicht mehr gebraucht, können Sie diese auch mit dem Delete-Button aus dem Portal entfernen.

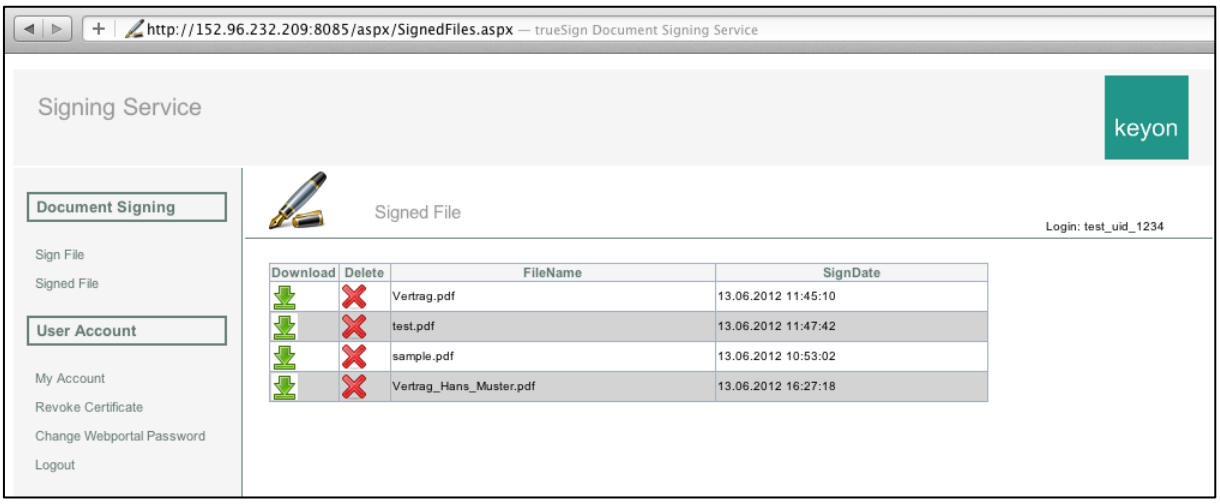


Abbildung 34 trueSign Webportal Signed Files

13.4.2.4. Signieren ablehnen

Muss aus einem Grund der Signierungsprozess abgebrochen werden, kann, statt den PIN zum Signieren einzugeben, auf *«Decline»* geklickt werden. Danach muss dies bestätigt werden. Nun kann für dieses Dokument keine Signatur mehr erstellt werden.

Im Hauptmenü wird der Eintrag mit dem abgelehnten Dokument rot eingefärbt.

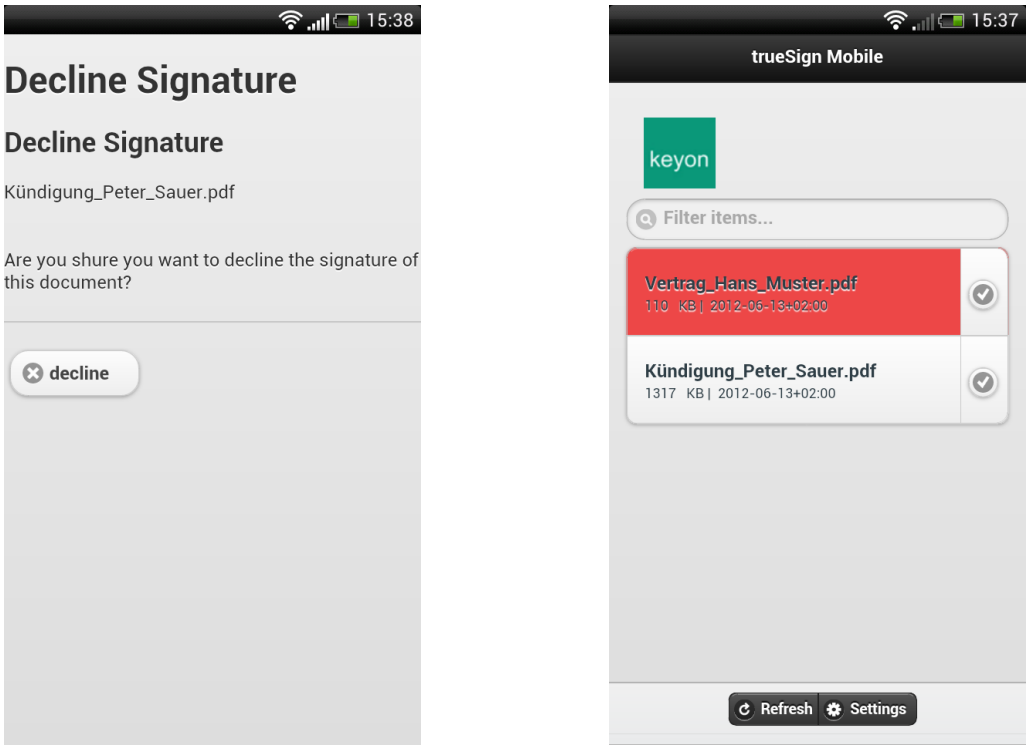


Abbildung 35 trueSign Mobile-App Decline Signature

13.4.3. Einstellungen

13.4.3.1. Benutzerkonto

Um Ihre Benutzerdaten einzusehen, können Sie im Webportal auf *«My Account»* klicken.

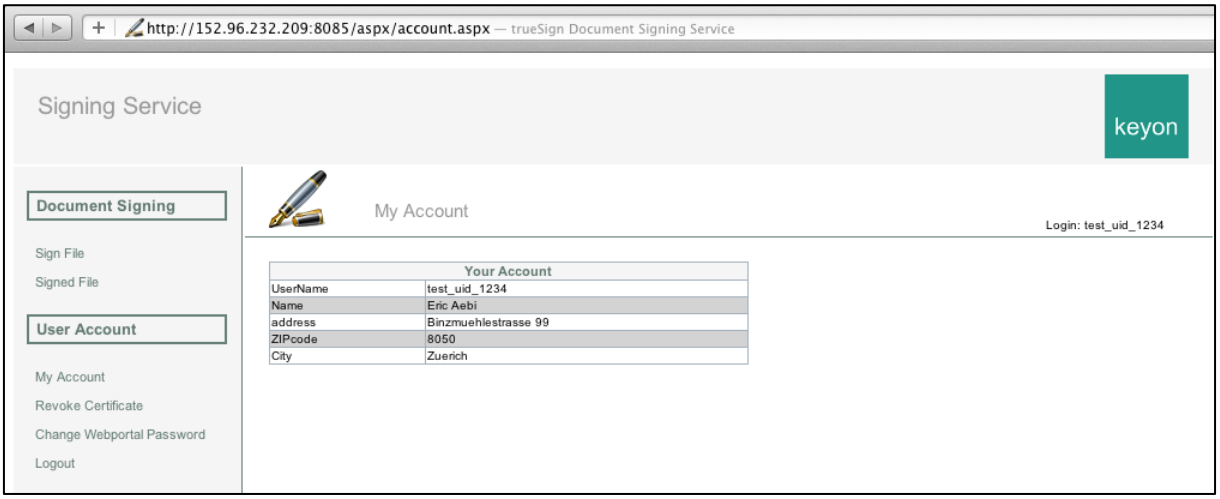


Abbildung 36 trueSign Webportal My Account

13.4.3.2. Zertifikat revozieren

Muss aus einem sicherheitstechnischen Grund, Ihr Zertifikat gesperrt werden, kann dies unter dem Menüpunkt *Zertifikat revozieren* vorgenommen werden.

***Vorsicht:** Danach kann Ihr Zertifikat nicht mehr verwendet und auch nicht wieder aktiviert werden. Es ist zwingend erforderlich, bei der Certification Authority ein neues Zertifikat anzufordern.*

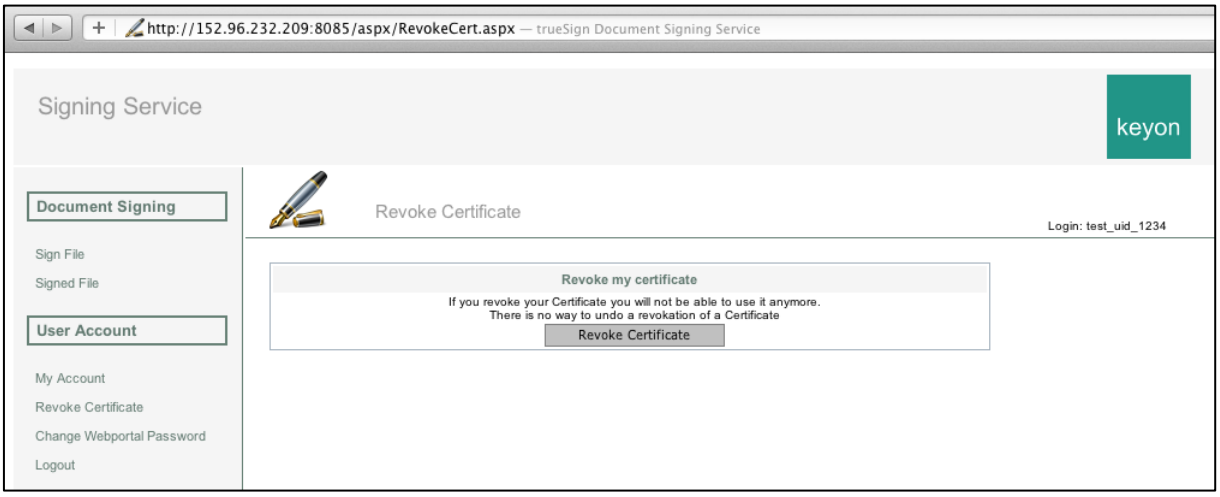


Abbildung 37 trueSign Webportal Revoke Certificate

13.4.3.3. Passwort für Webportal ändern

Sie können jederzeit Ihr Passwort für die Anmeldung im Webportal ändern. Klicken Sie im Menü auf *«Change Webportal Password».* Geben Sie zuerst Ihr bisheriges Passwort ein, danach das neue. Dieses bestätigen Sie, indem Sie es im folgenden Feld wiederholen. Klicken Sie auf *«Save»*.

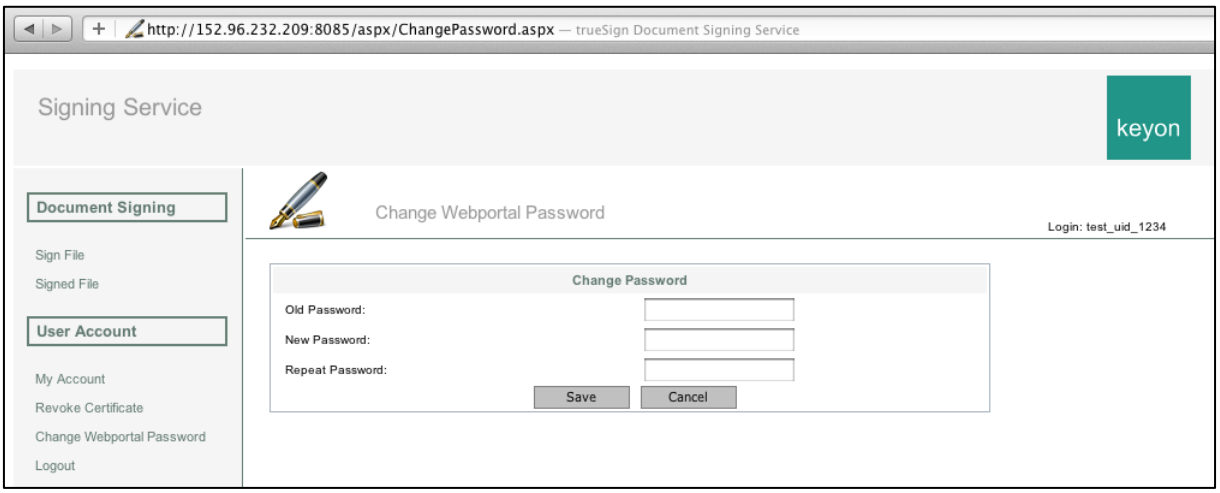


Abbildung 38 trueSign Webportal Change Password

14. Projektplan

14.1. Projektübersicht

Das rechtsgültige digitale Signieren von PDF-Dokumenten nach ZertES ist heute bereits möglich. Dazu benötigt der Benutzer eine SuisseID, ein Lesegerät sowie die entsprechende Software zum Erstellen und Prüfen der digitalen Signaturen. Den meisten Anwendern ist dies jedoch zu kompliziert oder sie scheitern bereits an der Installation. In Organisationen ergibt sich ein hoher Supportaufwand bezüglich der Installation von Treibern sowie der Unterstützung der Benutzer.

14.1.1. Zweck und Ziel

Hauptziel dieses Projektes ist, die Vereinfachung vom digitalen Signieren von Dateien. Das Projekt sieht einen zentralen Signaturdienst vor, bei welchem die kryptografischen Schlüssel der Anwender auf einem Hardware Security Module (HSM) gespeichert werden. Somit benötigt der EndUser keine Smartcard. Die gesetzlich vorgeschriebene Zwei-Faktor-Authentisierung wird erreicht, indem sich der User zuerst bei diesem Signaturdienst einloggt. Anschliessend kann er das zu signierende File hochladen. Dieses wird ihm dann auf dem Smartphone zur Kontrolle angezeigt. Bestätigt der User dies, wird das File mit seinem Signaturschlüssel signiert.

Für das Signieren der Dokumente kann auf eine bestehende Lösung der Firma Keyon AG zurückgegriffen werden. Diese muss um das Webportal, sowie um den Versand des zu signierenden Dokumentes an ein Smartphone erweitert werden. Dabei gilt es noch einige konzeptuelle Probleme zu klären, beispielsweise wie die Registrierung des Benutzers resp. des Smartphone bei diesem Service gelöst werden kann. Die technischen und organisatorischen Prozesse sollen zudem sicherheitstechnisch bewertet werden.

Ein ferneres Ziel ist es, die Applikation soweit auszubauen, dass die Anzeige der Dokumente auf allen gängigen Smartphone-Betriebssystemen funktioniert. Eine Möglichkeit wäre das PhoneGap-Framework, welches die Entwicklung, einer App für mehrere Betriebssysteme zulässt.

14.1.2. Annahmen und Einschränkungen

Obwohl dieses Projekt aus einer Studienarbeit und einer Bachelorarbeit besteht, wird es als ein gemeinsames Projekt betrachtet. Die Arbeit wird so unterteilt, dass die Bachelorarbeit den Serverteil umfasst, in welchem die bestehende Lösung der Keyon AG um das Webportal erweitert wird. Die Studienarbeit besteht darin, eine Lösung für das Empfangen und Anzeigen des Dokumentes auf einem Smartphone zu erstellen.

Die konzeptuellen Aspekte werden zusammen erarbeitet.

14.2. Projektorganisation

Das Projekt wird von einem Team bestehend aus zwei Studenten der HSR Hochschule für Technik in Rapperswil erarbeitet. Alle Teammitglieder sind hinsichtlich der Bewertung der Arbeit gleichgestellt. Der Aufwand für die Bachelorarbeit ist aber um 4 ECTS-Punkte höher und dauert entsprechend auch zwei Wochen länger als die Studienarbeit.

14.2.1. Organisationsstruktur

Teammitglied	Zuständigkeiten und Verantwortung
Eric Aebi	Webportal (Server)
Stefan Rohner	Mobile-Applikation (Client)

14.2.2. Externe Schnittstellen

- Betreuer ist Herr Prof. Dr. A. Steffen
- Industriepartner ist keyon AG
    - Ansprechpartner ist René Eberhard (CEO)
    - Technischer Kontakt ist Markus Isler (CFO)

14.3. Managementabläufe

14.3.1. Projekt Kostenvoranschlag

Das Projekt startet am 20. Februar 2012 und soll am 15. Juni 2012 abgeschlossen werden. In dieser Zeitspanne stehen für den Projektmitarbeiter der Studienarbeit ca. 18h pro Woche und dem Mitarbeiter der Bachelorarbeit ca. 20h pro Woche, gefolgt von einem zwei-wöchigen Block mit einem Arbeitsaufwand von 45 Stunden/Woche zur Verfügung. Das Gesamtbudget von 610 Stunden setzt sich zusammen aus 8 ECTS = 240 Stunden (Studienarbeit) und 12 ECTS = 370 Stunden (Bachelorarbeit).

14.3.2. Projektplan

14.3.2.1. Zeitplan

➔ Siehe Anhang A, Zeitplanung

14.3.2.2. Iterationsplanung / Meilensteine

Meilenstein	Titel	Resultat	Datum
MS1	Projektplan	Projektplan erstellt und alle Iterationen inkl. Daten geplant	SW04
MS2	Anforderungen und Analyse	Abschluss der Anforderungsspezifikation der Domainanalyse	SW05
MS3	Ende Elaboration	Architekturprototyp festgelegt, Release 0.1	SW08
MS4	Architektur / Design	Release 0.1b	SW12
MS5	Abgabe	Release 1.0	SW15

14.3.2.3. Besprechungen

Es sollen wöchentliche Meetings mit allen Projektmitglieder abgehalten werden. Diese finden jeweils mittwochs von 09:00 – 10:00 Uhr statt. An diesen Meetings sollen der Projektfortschritt und allfällige Probleme bei den einzelnen Arbeitsaufträgen besprochen und Massnahmen beschlossen werden.

14.3.2.4. Releases

Release	Typ	Requirements
0.1	Prototyp	
0.1b	Beta	
1.0	Final	Fertige Applikation

14.4. Risikomanagment

Zetieinheit: Projektstunden

Gewichteter Schaden 33.7

Risiko-Nr	Risikotitel	Risikobeschreibung	max. Schaden [h]	Eintrittswahr-scheinlichkeit	gewichteter Schaden [h]	Massnahmen zur Vermeidung/Verminderung	Vorgehen bei Eintreffen
Projektspezifische Risiken							
R1	Fehlein-schätzung des Aufwandes	Zeitplan wird nicht eingehalten, da der Aufwand von einzelnen Arbeitspaketen falsch eingeschätzt wurde	60	20.00 %	12.00	kritische Arbeitspakete (Domainmodell, Backupplanung, Konfigurationsverwaltung) frühzeitig (Elaboration 1&2) bearbeiten	nicht alle geplanten Funktionen und Features umsetzen, Mehrarbeit
R2	Sicherheits-aspekte halten rechtlich nicht	Trennung der Kanälen, weitere Notwendigkeiten z.B. eines Schlüssels erforderlich, reicht ev. ein PIN Code auf dem Smartphone, etc.	40	20.00 %	8.00	Genaue Analyse der Gesetzestexte Nach erarbeiten des Sicherheitskonzept Review mit Betreuer und der Keyon AG	Sicherheitskonzept überdenken und verbessern
R3	Probleme mit Technologien beim Webportal	Die zum Teil für die Mitarbeiter neuen Technologien benötigen mehr Zeit für die Einarbeitung	60	5.00 %	3.00	Einarbeitung in Internet-Technologien und Kennenlernen der verschiedenen Smartphone Technologien	Mehraufwand
R4	Probleme mit dem PhoneGap Framework	Phonegap unterstützt die benötigten Funktionen nicht	60	5.00 %	3.00	Proof of Technology mit Phonegap frühzeitig erarbeiten	Mobile Applikation für erste Version nur für Android Programmieren
R5	Push-Nachrichten	Push-Nachrichten können nicht gesendet werden und/oder App startet nicht direkt gestartet werden	20	5.00 %	1.00	Mit Smartphone frühzeitig Prototyp erstellen	Implementation ohne Push-Nachrichten

R6	Probleme mit Kommunikation Webportal <-> Smartphone	Kommunikation und Verschlüsselung kann nicht korrekt ausgeführt werden.	80	2.00 %	1.60	Proof of Technology mit Smartphone frühzeitig erarbeiten	Mehraufwand
Allgemeine Risiken							
R7	Ausfall der Infrastuktur	HW eines Projektmitgliedes fällt aus Git-Server fällt aus Netzwerk-Infrastruktur der HSR fällt aus	40	0.50 %	0.20	1. Alternative HW organisieren 2. Kopien auf Arbeitsrechner aktuell halten 3. mehrere Möglichkeiten für Datenaustausch innerhalb des Teams bereithalten	an Arbeitsplatzrechner in HSR-SA / BA-Zimmer arbeiten Neuer Git Server aufsetzen Daten über alternatives Netzwerk/Medien austauschen
R8	Datenverlust	erarbeitete Projekt-Artefakte werden lokal unwiderruflich geändert oder gelöscht	610	0.50 %	3.05	Artefakte konsequent unter Versionsverwaltung stellen, lokale Kopien, Hudson CI Server mit Autobackup auf einen externen Datenserver mit Hardware-RAID	Dateien aus letztem Backup vom letzten Build wiederherstellen und ev. auf den neuesten Stand bringen
R9	Ausfall eines Projektmitgliedes	Ausfall eines Projektmitgliedes aufgrund Krankheit, Unfall, Studiumabbruch	370	0.50 %	1.85	Verantwortlichkeiten in Team klar definieren	Arbeit innerhalb des bestehenden Teams aufteilen, Funktionsumfang kürzen
Summe			**1340**	**58.50 %**	**33.7**		

14.5. Arbeitspakete

Nr	Name	Inhalt	Iteration	Prio
0x	**Projektmanagement**			
01	Aufgabenstellung	Aufgabenstellung für AVT wird erstellt und grobe Aufwandschätzung definiert	Inception	1
02	Projektplan	Projektplan wird erstellt	Elaboration 1	1
03	Review & Anpassung Projektplan	Review des Projektplan	Elaboration 1	2
04	Zeitplanung Projektplan	Jedes Teammitglied trägt seine Stunden ein	Elaboration 1	1
05	Risiko-Management und Massnahmen	Erstellen des Riskmanagement	Elaboration 1	3
06	Q-Massnahmen	Erstellen der Q-Massnahmen	Elaboration 1	1
07	Infrastruktur	Einrichten der Infrastruktur, Versionierung	Elaboration 1	2
2x	**Requirements**			
20	Use Case brief	Erarbeiten aller Use Cases im briefformat	Elaboration 1	2
21	Use Case fully dressed	Erstellen der wichtigsten Use Cases in Fully dressed	Elaboration 1	2
22	Supplementary Spec.	Erarbeiten der Supplementary Specifications	Elaboration 1	2
4x	**Analyse**			
40	Domain Model	Domainmodell anhand der Usecases ausarbeiten	Elaboration 1	1
41	System Sequenzdiagramme	Darstellen der wichtigsten internen Abläufe	Elaboration 1	2
42	Operation Contracts	Verfassen der Contracts	Elaboration 1	2
43	Activity Diagramm	Darstellen des allgemeinen Ablaufs der Activity	Elaboration 1	2
44	Externes Design UI	Erstellen der Paperprototypes	Elaboration 1	1
45	Analyse Registrierung der Benutzer	Registrierung der Benutzer auf dem Webportal analysieren	Elaboration 1	2
46	Analyse Sicherheitsmerkmale	Sicherheitsmerkmale des Signaturdienstes analysieren	Elaboration 1	2
47	Analyse Mobile Kommunikation	Funktionen der Kommunikation auf Android analysieren	Elaboration 1	2
48	Review & Korrektur Anforderungen	Review	Elaboration 1	3
49	Review & Korrektur Externes Design UI	Review	Elaboration 1	3

6x	**Design**			
60	Design Model	Erstellen des Design Models	Elaboration 2	2
61	Logische Architektur	Festlegen der logischen Architektur	Elaboration 2	1
62	Review & Korrektur Design	Review	Elaboration 2	2
63	Internes Design	Internes Design festlegen	Elaboration 2	2
8x	**Implementation**			
	Prototyp		Elaboration 2	2
80	Prototyp: GUI	Prototyp des GUIs erstellen	Elaboration 2	2
81	Prototyp: Webportal	Prototyp des Webportals erstellen	Elaboration 2	2
82	Prototyp: Zugriff mit PC – Smartphone	Prototyp der Zugriffe PC – Smartphone erstellen	Elaboration 2	2
83	Prototypen zusammenführen	Alle Prototypen zusammenführen	Elaboration 2	2
	Ausbau Prototyp		Construction 1	2
84	Projekt Automation	ANT script für automatischen Build erstellen u.s.w.	Construction 1	2
85	Ausbau GUI	GUI auf Endzustand ausbauen	Construction 1	2
86	Ausbau Webportal	Webportal auf Endzustand ausbauen	Construction 1	2
87	Ausbau Zugriff mit PC – Smartphone	Zugriff mit PC – Smartphone auf Endzustand ausbauen	Construction 1	2
88	Ausbau Portierung	Portierung auf verschiedene Smartphones inkl. iPad	Construction 1	2
89	Refactoring	Ganzer Code überarbeiten	Construction 1	2
10x	**Test / Bugfixing**			
100	Test Units entwickeln	Unit Test für zwingende Funktionen erarbeiten	Elaboration 3	2
101	Test Units durchführen	Unit Test durchführen	Construction 2	2
102	Bugfixing	Gefundene Fehler beheben	Construction 2	2
103	Usability Tests	Usability Tests durchführen	Construction 2	2
104	Systemtest	Systemtest durchführen	Construction 2	2
12x	**Dokumentation**			
120	Übersicht Q-Massnahmen	Dokumentation über Q-Massnahmen prüfen	Transition 1	2
121	Dokumentation	Dokumentation erstellen	Transition 1	2
122	BA-Schlusspräsentation	Schlusspräsentation vorbereiten und halten	Transition 1	2

14.6. Infrastruktur

Um das Projekt zu bewältigen, arbeitet jedes Teammitglied mit seinem persönlichen Laptop, sowie dem von der HSR zur Verfügung gestellten Desktop Rechner. Für die Entwicklung sowie die Tests auf der Serverseite soll die Umgebung der Firma Keyon AG benutzt werden.

Auf den Rechnern wird zur Entwicklung der Mobile-Applikation (Clientseite) IDE Eclipse Indigo installiert. Zusätzlich werden die Android Developer Tools eingebunden. Diese enthalten einen Android Emulator, um die Applikation zu testen. Zur Portierung der in Java geschriebenen Software in andere Technologien, wie z.B. iOS, soll PhoneGap 1.6 verwendet werden.

Für die Sicherstellung der Versionskontrolle sowie die Dokumentenverwaltung wird zudem ein externer Git-Server benutzt.

Die Zeiterfassung wird in einer Liste in MS Excel von jedem Mitarbeiter selbständig mitgeführt.

14.7. Qualitätsmassnahmen

14.7.1. Massnahmen

14.7.1.1. Dokumentation

Jedes erstellte Dokument muss vom anderen Teammitglied korrigiert und überarbeitet werden.

Zudem sind Dokument-Reviews im Projektplan ersichtlich, bei welchen die Dokumentation zusammen mit dem Betreuer besprochen wird.

14.7.1.2. Quellcode

Um die Code Qualität hoch zu halten werden im Folgenden die Coderichtlinien beschrieben, an welche sich alle Teammitglieder zu halten haben.

Ausserdem wurden Code-Reviews geplant und im Projektplan eingetragen, an welchen der Code mit dem Betreuer besprochen wird. Weitere Reviews wurden im Projektplan mit dem Industriepartner Keyon AG festgelegt.

Formatierung

Da das ganze Team mit Eclipse arbeitet, soll auch gleich die in Eclipse vorhandene automatisierte Formatierung mit der Java-Standard-Einstellung, gebraucht werden. Somit werden keine zusätzlichen Regeln über Einzug, Klammersetzung usw. benötigt.

Namenskonventionen

Alle Namen von Klassen, Variablen usw. sollen in der bekannten CamelCase-Schreibweise und in Englisch geschrieben werden. Zudem sind bei der Namensgebung folgende Richtlinien zu beachten.

Element	Schreibweise	Beispiel
Klasse, Interface	• Nomen • Beginnt mit Grossbuchstaben • Interfaces beginnen mit gross „I“ • Enthält Name des verwendeten Patterns	ObjectFactory
Methode	• Verb • Beginnt mit Kleinbuchstaben	calculateThings()
Konstante	• Nur Grossbuchstaben, Wörter Trennung mit _	NEVER_CHANGE
Attribut	• Beginnt mit m (Member)	mColor
Variable	• Beginnt mit Kleinbuchstaben	counter

Kommentar

Kommentare werden in Deutsch verfasst. Jede Klasse muss zwingend einen einleitenden Kommentar enthalten. Dieser muss mit JavaDoc kompatibel sein. Komplizierte bzw. lange Methoden sollten wenn mögliche vermieden werden. (Oder allenfalls mit einem JavaDoc konformen Kommentar versehen werden.)

14.7.1.3. Unit Testing

Damit die Funktionalität des Programms überprüft werden kann, werden während dem Entwickeln der Software vorzu Test-Units erstellt. Diese Tests werden jeweils vor der Implementierung einer weiteren Programmfunktion erstellt und anschliessend durchgeführt.

Systemtests

Systemtests mit dem Webportal enthalten folgende Ausführungen:

- Stimmt der Loginprozess?
- Kann der Benutzer die hochgeladenen Daten einsehen?

Für die Systemtests mit Smartphone wird der Android-Emulator verwendet.

Als Test wird jeweils Folgendes überprüft:

- Erhält der Benutzer die Benachrichtigung des Webservers?
- Kann sich der Benutzer einloggen?
- Wir das Dokument korrekt angezeigt?
- Findet der Upload zum Webserver korrekt statt?

Um die Systemtests durchzuführen, wird die Software gemäss den Use-Cases verwendet.

14.7.1.4. Versionsmanagement

Für das Projekt wird ein Git-Repository erstellt, in welchen alle Dokumente abgelegt und versioniert werden.

14.8. Projektmonitoring

14.8.1. Zeiterfassung

Jedes Teammitglied erstellt eine eigene Zeiterfassung. Hierfür wird Excel verwendet. Darin soll ersichtlich sein, wie viele Stunden, an welchen Tagen, für welche Arbeiten aufgewendet wurden.

14.8.2. Sitzungsprotokolle

Es soll um die Beschlüsse nachvollziehbar zu machen, jedes Meeting protokoliert werden. In diesen Protokollen soll folgendes aufgezeigt werden:

- wann hat das Meeting stattgefunden,
- wer hat am Meeting teilgenommen,
- was waren die Traktanden,
- was wurde zu den Traktanden beschlossen,
    - Wer ist dafür verantwortlich,
- wann findet das nächste Meeting statt.

15. Technischer Bericht

15.1. Übersicht

Die Arbeit hatte einen grossen konzeptuellen Teil. Die rechtsgültige digitale Signatur ist kein neues Prinzip. Jedoch ist die digitale Signatur bis jetzt nur mittels einem Hardwaretoken, wie zum Beispiel einer Smartcard, möglich. Das ZertES wurde im August 2011 soweit angepasst, dass eine rechtsgültige Signatur auch ohne Hardwaretoken möglich ist. Dabei müssen aber einige Sicherheitsmerkmale erfüllt werden. Diese galt es zu analysieren und ein Konzept für eine Applikation zu erstellen, welches all diese Forderungen erfüllt. Oberstes Ziel einer solchen Applikation ist es zu verhindern, dass ein Angreifer eine Signatur mit dem Zertifikat einer Drittperson ausführen kann.

15.2. Ergebnisse des Konzeptes

In der konzeptuellen Phase der Arbeit haben wir mehrere Varianten analysiert. Für die Umsetzung haben wir uns für eine Variante entschieden. Die wichtigsten Varianten sollen hier kurz beschrieben werden.

15.2.1. Variante: Dokument als Hash

Die zu signierenden Dokumente stehen im Mittelpunkt dieser Applikation. In einer realen Anwendung werden dies häufig Dokumente sein, bei welchen eine gewisse Geheimhaltung bzw. Privatsphäre sehr wichtig ist. Mit diesem Punkt kam die ursprüngliche Idee, nach dem Hochladen in das Webportal einen Hashwert der Dokumente zu erstellen. Dann könnte dieser an den trueSign JCE Service weitergegeben werden und durch diesen signiert werden. Die Gefahr, dass ein Angreifer die Dokumente abfangen und einsehen kann, wäre somit wesentlich geringer. Zudem würde es vermutlich auch der Philosophie einer CA entgegenkommen, da eine CA häufig nicht direkt mit Userdaten in Kontakt kommt und dies auch gar nicht will.

Diese Variante kollidierte leider mit dem Vorhaben, das zu signierende File an ein Smartphone zur Bestätigung zu schicken. Diese Bestätigung sollte von einem Modul möglichst nahe beim eigentlichen Signieren verlangt werden, um ein Verändern der Daten nach der Bestätigung so gut als möglich auszuschliessen. Dies bedingt aber, dass das komplette Dokument in diesem Modul vorhanden ist. Somit ist es leider nicht möglich, nur einen Hashwert der Datei zur Weiterverarbeitung zu verwenden.

15.2.2. Variante: Webportal bei CA

Da, wie im Kapitel 15.2.1 beschrieben, das komplette Dokument an die CA für die Signatur sowie die Bestätigung geschickt werden muss, wurden Überlegungen geweckt, die ganze Applikation zu vereinfachen und alles, inklusive Webportal, als ein Modul zu implementieren. Diese Applikation hätte dann von einer CA betrieben werden müssen, da die rechtsgültige Signatur nur von einer anerkannten CA ausgeführt werden kann. Diese Variante wäre von der Implementation sicherlich die einfachste gewesen, da die jetzt erstelle SOAP-Schnittstelle zwischen dem Webportal und dem trueSign Mobile Server weggefallen wäre und alles in einer Webapplikation hätte behandelt werden können.

Allerdings widerspricht diese Variante der Philosophie einer CA komplett. Eine CA muss ihre Rechner und Daten extrem gut schützen und will deshalb möglichst wenig Kommunikation mit der Aussenwelt zulassen. Natürlich ist die Kommunikation nicht ganz auszuschliessen, jedoch wird diese nur mit bekannten Partnern geführt. Ein Webportal, auf welchem jeder, der ein Dokument signieren möchte Zugriff haben muss, wird vermutlich keine CA betreiben wollen.

15.2.3. Variante: Webportal getrennt, Signatur bei CA (umgesetzt)

Die von uns umgesetzte Variante vereint die Überlegungen aus den beiden anderen Varianten. Die Bestätigungsanfrage wird vom trueSign Mobile Service an das Smartphone geschickt. Dieser Service leitet das Dokument anschliessend direkt an den trueSign JCE Service weiter, welcher das Dokument signiert. Die Gefahr, dass ein Angreifer das Dokument zwischen dem trueSign Mobile Service und dem trueSign JCE Service abfangen und verändern kann, ist sehr gering. Denn beide Applikationen befinden sich innerhalb der CA. Wenn auch, in einer realen Anwendung, wahrscheinlich auf verschiedenen Sicherheitsstufen.

Das Webportal wurde als eigenständiges Modul entwickelt und kommuniziert mit dem trueSign Mobile Service via SOAP-Schnittstelle. Dadurch kann das Webportal unabhängig und auch ausserhalb der CA betrieben werden. Es ist somit auch möglich, dass ein Dritter als Anbieter des Signaturdienstes auftritt und die CA im Hintergrund nur die Signaturen erstellt.

15.2.3.1. Kritische Punkte

Web-Entry Server

Einige Angelegenheiten müssten bei einer realen Anwendung noch mit dem Betreiber der Applikation erörtert werden. Das Webportal wurde als eigenständiges Modul umgesetzt, um die Kommunikation mit den für die CA unbekannten Benutzern, von der CA abzuschirmen. Jedoch benötigt die Bestätigung via Smartphone ebenfalls eine Kommunikation von dem Benutzer in Richtung der CA. Dies kann nicht umgangen werden, da mit jeder Schicht mit der sich der Empfänger der Bestätigung von dem die Signatur ausführenden Modul entfernt, die Sicherheit und somit die Essenz der Bestätigung abnimmt.

Der trueSign Mobile Service würde jedoch in einer CA sicherlich nicht direkt ans Internet angebunden werden. Viel mehr würde vorher noch ein Web-Entry Server zum Einsatz kommen. Dies müsste natürlich in der trueSign Mobile-App berücksichtigt werden. Denkbar wäre eine Variante, bei welcher sich die Mobile-App mittels Username und einer Hashfunktion des Passwortes des Webportals oder des Initial-PIN beim Web-Entry Server mittels Basic Authentification anmeldet. Das würde aber Anpassungen an der trueSign Mobile-App bedeuten. Allerdings lässt sich dies kaum vermeiden, um die Applikation in einer realen Umgebung einsetzen zu können.

Registrierung der Benutzer

Für die Registrierung der Benutzer müsste so etwas wie ein Superuser eingefügt werden. Dieser müsste nachdem sich ein User für den Signatur Service angemeldet hat, und den ganzen Registrierungsprozess durchlaufen hat, die Möglichkeit haben, einen Benutzer für das Webportal aufzuschalten. Hierfür müsste auch der Initial-PIN noch generiert und an den Benutzer verschickt werden.

15.3. Ergebnisse Umsetzung

Hinsichtlich der Features, die wir in unserer Applikation geplant haben, konnten wir alle Schlüsselfunktionen realisieren. Dies beinhaltet seitens der trueSign Mobile-App das Herunterladen und Anzeigen der zu signierenden Dokumente, das Verschicken der Bestätigung beziehungsweise der Verwerfung der Signatur und das Empfangen von Push-Meldungen, welche zur Anzeige der Mobile-TAN für das Login beim trueSign Webportal und für die Information an den Benutzer, dass ein neues File zur Signatur bereit steht, genutzt werden. Zudem konnte die grafische Benutzeroberfläche mit dem PhoneGap- und JQuery-Mobile-Framework so gestaltet werden, wie sie im Paper Prototype skizziert wurde.

Deutlich unterschätzt haben wir die Anzeige eines PDF-Files mit einem eigenen Viewer, der ein Overlay mit spezifischen Schaltflächen enthält. Leider würde eine Implementierung eines eigenen Viewer den Rahmen dieser Arbeit sprengen. Darum haben wir uns entschieden, den standardmässigen PDF-Viewer des Smartphone zu nutzen und dann die Möglichkeit zu bieten mit dem Back-Button des Geräts in die App zurückzukehren.

Leider mussten wir für die Zeit welche wir durch den Einsatz von JQuery-Mobile und PhoneGap beim Erstellen des GUI der Mobile-App gespart hatten, bei der anschliessenden Implementation der Logik doppelt bezahlen. Viele der Schlüsselfunktionen der Applikation wie die Datenübertragung via SOAP und das Empfangen von Push-Nachrichten, sind mit PhoneGap nicht oder nur sehr schwer möglich. Deshalb mussten diese Funktionen nativ in Android programmiert werden. Dies musste anschliessend auf eher umständliche Weise wieder mit dem PhoneGap-Framework verknüpft werden. Hier wären wir um einiges schneller gewesen, hätten wir auf PhoneGap verzichtet und alles nativ in Android entwickelt. Deshalb musste leider auf die Implementation der Möglichkeit die PIN für den Private Key in der trueSign Mobile-App zu ändern, verzichtet werden.

15.3.1. Erfüllung des Gesetzes

In diesem Kapitel sollen kurz die wichtigsten Anforderungen des ZertES an eine qualifizierte elektronische Signatur beziehungsweise deren Erstellung beschrieben werden. Auch wird die Einhaltung innerhalb der Applikation geprüft, die referenzierten Artikel können im Anhang B nachgelesen werden.

15.3.1.1. ZertES Artikel 2 Absatz b. (Anforderungen an eine qualifizierte Signatur)

Laut Gesetz muss die Signatur ausschliesslich der Inhaberin oder dem Inhaber zugeordnet werden können. Durch den in der Arbeit beschriebenen Registrierungsprozess – im Speziellen das für den Antragssteller notwendige Erscheinen beim Aussteller der Zertifikate und das Vorweisen eines amtlichen Ausweises – ist das eindeutige Zuordnen der Signatur gewährleistet.

Da die PIN für den Private Key nur dem Benutzer zugestellt wird, und dieser auch nie zwischengespeichert oder abgelegt wird, hat der Inhaber des Zertifikates die alleinige Kontrolle über die Mittel, mit welchen die Signatur erstellt werden.

Der trueSign JCE Service erstellt eine in der Datei eingebettete Signatur, durch eine Veränderung der Datei würde die Signatur ungültig und die nachträgliche Veränderung kann erkannt werden.

Somit sind alle Anforderungen des Gesetzes an eine qualifizierte Signatur erfüllt. Einzig die in Artikel 2 Absatz c.) beschriebene Signaturerstellungseinheit, bei welcher es sich eigentlich um ein FIPS 140-2 Level 3 zertifiziertes HSM handelt stand für unsere Testinstallation leider nicht zur Verfügung. Eine solche Einheit könnte aber ohne weiteren Entwicklungsaufwand an das System angeschlossen werden.

15.3.1.2. TAV über Zertifizierungsdienste im Bereich der elektronischen Signatur

Die technischen und administrativen Vorschriften (TAV) stellen weitere Anforderungen an die Signaturerstellungseinheit. Diese wurden in unserer Testinstallation nicht eingehalten, können jedoch in einer echten Anwendung von einem HSM, das in der Umgebung von einer CA in Betrieb genommen wird, ohne Probleme erfüllt werden. Somit können auch die Anforderungen der TAV als Erfüllt betrachtet werden.

Kapitel 3.3.3 b

Dieses Kapitel beschreibt die Anforderungen an eine Signaturerstellungseinheit bei welcher sich der Zertifikatsinhaber und die Einheit selbst nicht am selben Ort befinden. Auch hier gibt es Anforderung welche unser Testsystem nicht erfüllt, aber in einer produktiven Umgebung problemlos umzusetzen wären. Wie Die FIPS 140-2 Level 3 Zertifizierung der Einheit sowie der Betrieb der Komponenten in einer gesicherten Umgebung, wie es in einer CA der Fall wäre.

Die Anforderung an einen Logisch getrennten, gesicherten Kanal zwischen der Signaturapplikation, den Komponenten zur Signaturerstellen und dem Zertifikatsinhaber hingegen erfüllt die Applikation vollständig, da die Bestätigung der Signatur nicht nur Logisch getrennt von den Restlichen Kommunikationskanälen ist, sondern auch noch auf einem anderem Medium basiert. Dieser Kanal muss zudem auch die Identifizierung der Endpunkte und die Vertraulichkeit und Integrität der übermittelten Daten gewährleisten können. Da dies mittels SSL problemlos bewerkstelligt werden kann, wird auch diese Anforderung von der Entwickelten Applikation erfüllt.

Weiter muss im Vorfeld der Signatur ein Zwei-Faktor Identifizierungs- und Authentifizierungs-mechanismus bereitgestellt werden. Das trueSign Webportal in Kombination mit der trueSign Mobile-App bietet genau dies an und gewährleistet so dem Zertifikatsinhaber die alleinige Kontrolle über erstellte Signaturen.

Da die trueSign Mobile-App dem Zertifikatsinhaber die Möglichkeit bietet, die Datei kurz bevor die Signatur erstellt wird, nochmals einzusehen und zu prüfen, kann auch sichergestellt werden, dass der Benutzer tatsächlich die Absicht hat, die gelieferten Daten zu signieren.

15.4. Schlussfolgerung

Die entwickelte Applikation erlaubt Dokumente rechtsgültig zu signieren und zwar über ein Intuitives UI via Webportal und Mobile-App. Viele Sicherheitstechnische Probleme konnten in der konzeptuellen Phase der Arbeit erörtert werden und Lösungen aufgezeigt werden. Ein Grossteil davon wurde auch in der Applikation Umgesetzt. Sie stellt somit für die Firma Keyon AG eine sehr gute Basis dar, um die Vision der simplen digitalen Signatur für Jedermann weiter zu verfolgen und auszubauen.

15.5. Ausblick

Für die Weiterführung der Mobile-App, wäre es erforderlich, eine Cross-Plattform-Applikation zu entwickeln, welche sowohl mit Android-Smartphones sondern auch mit iOS-Geräten wie einem iPhone oder iPad, Windows7-Geräten oder Blackberries bedienbar ist. Notwendigerweise müssten, um weiterhin PhoneGap einzusetzen, noch weitere Plugins geschrieben werden, welche eine Schnittstelle zur nativen Programmierumgebung der jeweiligen Geräte anbieten. Des Weiteren müsste die Mobile wie in Kapitel 15.2.3.1 beschrieben für die Kommunikation mit einem Webentry-Server erweitert werden.

16. Persönlicher Bericht Stefan Rohner

In den letzten zwölf Wochen bearbeitete ich mit meinem Kollegen Eric Aebi eine sehr spannende aber konzeptuell auch anspruchsvolle Arbeit. Wir waren fast immer einer Meinung und sehr motiviert, ein funktionsfähiges Produkt zu entwickeln. Sehr angenehm war auch die Zusammenarbeit mit unserem Betreuer, Prof. Dr. Andreas Steffen und dem Industriepartner Keyon AG in Jona, welche wichtige konzeptuelle Inputs während den Meetings (z.B. in den Reviews der Meilensteine) geben konnten.

Zu den für mich interessantesten Punkten gehörten unter anderem:

- Besuch bei der QuoVadis AG in St. Gallen, ein Certification Service Provider
- Sicherheitsanalyse
- GUI-Analyse
- Kennenlernen des PhoneGap- und JQuery-Frameworks

Ich stellte fest, dass wir mit den Analyse- und Designdokumenten – vor allem mit schematischen Visualisierungen – sehr präzise über die konzeptuelle Sicht betreffend Sicherheit diskutieren und somit die Schwachstellen erkennen konnten. Bei der weiteren Ausarbeitung der Dokumente oder bei der Implementierung, änderte sich dann aber plötzlich das weitere Vorgehen und ein Teil des Konzepts wurde auf den Kopf gestellt. Dies war nicht ganz einfach, da wir vor allem die Programmierung zeitlich nach hinten verschieben mussten. Aber so wussten wir, dass wir auf dem richtigen Weg waren.

Erfahren durfte ich auch, dass Analysen und Nachforschungen in der Machbarkeit im technischen Bereich sehr zeitaufwendig sein können, obwohl diese in der Projektplanung grosszügig einkalkuliert wurden. Beispielsweise konnte das PhoneGap-Framework, mit welchem wir ursprünglich gerne die ganze App entwickeln wollten, letztendlich nur für wenige Teile, z.B. für die Menüführung verwendet werden, da Anforderungen an die Mobile-App nicht mit diesem Framework umgesetzt sind.

Abschliessend kann ich aber sagen, dass ich sehr viel dazugelernt habe. Vor allem im Bereich der digitalen Signatur und natürlich während der Programmierarbeit.

17. Persönlicher Bericht Eric Aebi

Die Arbeit war für mich sehr spannend und lehrreich. Ich konnte dank dem grossen konzeptuellen Teil und den Reviews mit der Firma Keyon AG wie auch mit Herrn Steffen viel im Sicherheitsbereich lernen und mir auch ein gewisses „sicherheitstechnisches-Denken“ aneignen. Anfangs tauchten bei jeder Lösung eines Problems neue Fragen auf, die es zu beantworten galt. Mit fortschreitender Projektdauer wurde unsere Sicht klarer und durch die bereits gesammelte Projekterfahrung, waren die Herausforderungen einfacher zu lösen.

Die Zusammenarbeit mit Herrn Steffen war sehr angenehmen und unsere wöchentlichen Meetings waren immer produktiv. Da Ich bereits seit eineinhalb Jahren bei der Firma Keyon AG arbeite, fiel mir die Zusammenarbeit natürlich besonders leicht.

Das Teamwork mit Stefan Rohner verlief ohne Probleme, was sicher auch darauf zurückzuführen ist, dass wir uns schon sehr lange kennen und auch schon öfters miteinander gearbeitet haben. Einzig das Projektmanagement hat vermutlich darunter etwas gelitten. Dank der guten Kommunikation wussten wir immer, wer was als nächstes zu tun hat. Wir haben uns auch häufig bei Arbeiten gegenseitig geholfen oder diese abgetauscht, um die eigenen Kompetenzen möglichst effizient zu nutzen. Leider haben wir dies jeweils aber nur schlecht oder nicht dokumentiert. Zu Beginn des Projektes hatten wir zwar eine grobe Zeitplanung erstellt, diese verloren wir aber während dem Arbeiten etwas aus den Augen.

Obwohl der konzeptuelle Teil sehr spannend war, denke ich, dass es rückblickend besser gewesen wäre. Mit der Implementation etwas früher zu beginnen, da wir hier gegen Ende der Arbeit etwas in Zeitnot geraten sind. Vor allem die Risikobereiche wie das PhoneGap-Framework, hätten wir in der Startphase intensiver untersuchen sollen. Möglicherweise wäre uns auf diese Weise das Zeitmanagement bei einzelnen Aufgaben etwas leichter gefallen.

18. Angang A Zeitplanung

19. Anhang B Gesetzestexte

In diesem Anhang sind die innerhalb des Berichts referenzierten Gesetzestexte gelistet.

19.1. Bundesgesetz über Zertifizierungsdienste im Bereich der elektronischen Signatur

(Bundesgesetz über die elektronische Signatur, ZertES)

Artikel 2.

Absatz b

1. Sie ist ausschliesslich der Inhaberin oder dem Inhaber zugeordnet.
2. Sie ermöglicht die Identifizierung der Inhaberin oder des Inhabers.
3. Sie wird mit Mitteln erzeugt, welche die Inhaberin oder der Inhaber unter ihrer oder seiner alleinigen Kontrolle halten kann.
4. Sie ist mit den Daten, auf die sie sich bezieht, so verknüpft, dass eine nachträgliche Veränderung der Daten erkannt werden kann;

Absatz c

qualifizierte elektronische Signatur: eine fortgeschrittene elektronische Signatur,

die auf einer sicheren Signaturerstellungseinheit nach Artikel 6 Absätze

1 und 2 und auf einem qualifizierten und zum Zeitpunkt der Erzeugung

gültigen Zertifikat beruht;"

Artikel 10

Absatz 1

Die anerkannten Anbieterinnen von Zertifizierungsdiensten erklären ein qualifiziertes

Zertifikat unverzüglich für ungültig, wenn:

a) die Inhaberin oder der Inhaber oder die Person, die sie oder ihn vertritt, einen entsprechenden Antrag stellt;
b) sich herausstellt, dass dieses unrechtmässig erlangt worden ist;
c) es keine Gewähr mehr bietet für die Zuordnung eines Signaturprüfschlüssels zu einer bestimmten Person.

Absatz 2

Bei der Ungültigerklärung nach Absatz 1 Buchstabe a müssen sie sich vergewissern,

dass die Person, welche die Ungültigerklärung beantragt, dazu berechtigt ist.

19.2. Technische und administrative Vorschriften über Zertifizierungsdienste im Bereich der lektronichen Signatur

Kapitel 3.3.3a

Die CSP muss den Antragstellerinnen und Antragstellern eines Zertifikats sichere Signaturerstellungseinheiten liefern, die den Mindestanforderungen von Artikel 6 Absatz 2 ZertES [1] entsprechen, oder sicherstellen, dass diese solche verwenden. Mit dem Dokument CWA 14169 [15] wird die Konformität mit den Anforderungen von Artikel 6 Absatz 2 ZertES [1] sichergestellt. Die sicheren Signaturerstellungseinheiten müssen zudem folgende zusätzliche Anforderungen erfüllen:

- sie dürfen weder den zu signierenden Inhalt ändern, noch die signierende Person daran hindern, diesen Inhalt vor dem Signieren genau zur Kenntnis zu nehmen;
- das qualifizierte Zertifikat (oder der eindeutige Verweis auf dieses Zertifikat) muss im System vorhanden sein;

- der dem qualifizierten Zertifikat entsprechende Signaturschlüssel darf nicht verwendet werden, bevor er durch Aktivierungsdaten aktiviert worden ist;
- inkorrekte und aufeinander folgende Aktivierungsversuche müssen festgestellt werden können;
- wenn eine im Voraus festgelegte Anzahl aufeinander folgender und inkorrekter Aktivierungsversuche erreicht wurde, muss der Gebrauch der Signaturschlüssel gesperrt werden. Diese Anzahl darf nicht grösser als 4 Versuche für eine PIN-Länge von 6 Zeichen sein. Bei einer längeren PIN kann sie grösser als 4 sein, sofern die vom Entwickler der zertifizierten sicheren Signaturerstellungseinheiten bereitgestelle Dokumentation das vorsieht;
- die CSP kann einen gesperrten Signaturschlüssel erst freigeben, nachdem sie verifiziert hat, dass der Antrag auf Freigabe vom Schlüsselinhaber stammt.

Kaptitel 3.3.3b

Die Zertifizierung der sicheren Signaturerstellungseinheiten muss für alle oben stehenden Anforderungen erhältlich sein und

- entweder die Prüfstufe EAL 4 der Norm ISO/IEC 15408:2005 [11] umfassen, erhöht um die Versicherungselemente AVA_MSU.3 (vulnerability assessment, analysis and testing of insecure states) und AVA_VLA.4 (vulnerability assessment, highly resistant) oder die entsprechende Versicherungselemente der Norm ISO/IEC 15408:2009,
- oder die Prüfstufe E3 hoch des Dokuments ITSEC [14] umfassen.

Andere Komponenten gelten als sichere Signaturerstellungseinheit, um qualifizierte elektronische Signaturen gemäss ZertES zu erstellen, sofern folgende Anforderungen erfüllt sind:

- die Signatur wird in einer nach FIPS 140-2 Level 3 (oder höher) zertifizierten Einheit oder gemäss einem für die Erstellung von qualifizierten elektronischen Signaturen ausgearbeiteten Schutzprofil (Protection Profile) erstellt;
- die für die Speicherung des Signaturschlüssels und die Signaturerstellung verwendeten Komponenten müssen in einer gesicherten Umgebung verwaltet und betrieben werden, und zwar gemäss denselben Verwaltungs- und Betriebsgrundsätzen, wie sie für die Infrastruktur der CSP gelten und in der Spezifikation ETSI TS 101 456 [6], Kapitel 7.4 CA management and operation beschrieben sind;
- es muss ein gesicherter Kanal zwischen der Zertifikatsinhaberin oder dem Zertifikatsinhaber, der Signierapplikation und den Komponenten zur Signaturerstellung und Speicherung des Signaturschlüssels eingesetzt werden, wenn die Zertifikatsinhaberin oder der Zertifikatsinhaber und diese Komponenten sich nicht am selben Ort befinden. Dieser gesicherte Kanal muss logisch von den anderen Kommunikationskanälen getrennt sein sowie die Identifizierung seiner Endpunkte und die Vertraulichkeit und Integrität der übermittelten Daten gewährleisten;
- verwendet die Zertifikatsinhaberin oder der Zertifikatsinhaber den Signaturschlüssel aus der Distanz, ist im Vorfeld ein geeigneter 2-Faktor Identifizierungs- und Authentifizierungsmechanismus bereitzustellen, um die alleinige Kontrolle durch die Zertifikatsinhaberin oder den Zertifikatsinhaber zu gewährleisten;
- vor der Erstellung einer elektronischen Signatur ist ein Verfahren umzusetzen um sicherzustellen, dass die Zertifikatsinhaberin oder der Zertifikatsinhaber tatsächlich die Absicht hat, die der Signaturerstellungseinheit gelieferten Daten zu signieren.

20. Literaturverzeichnis

Bilder in Personas:

1. Papa Moll: http://www.img.biz/en/impuls-productions/produced-movies/papa-moll/
    a. Letzter Zugriff 15.06.2012
2. Ueli Untermaurer: http://oliraths.ch/humor/viktor/ueli.jpg
    a. Letzter Zugriff 15.06.2012

20.1. Links

1. http://ksoap2.sourceforge.net/
2. https://developers.google.com/android/c2dm/
3. http://jquerymobile.com/test/
4. http://docs.phonegap.com/en/1.8.1/index.html
5. https://github.com/phonegap/phonegap-plugins

21. Abkürzungsverzeichnis

Abkürzung	Bedeutung
C2DM	Cloud to device messaging
APK	Android application package
APNS	Apple Push Notification Service
CA	Certification Authority
HSM	Hardware security modul
IIS	Internet Information Services
IMEI	International Mobile Equipment Identity
JAX-WS	Java API for XML Web Services
JCE	Java Cryptography Extension
PIN	Persönliche Identifikationsnummer
RA	Registration Authority
SDK	Software Development Kit
SOAP	Simple Object Access Protocol
SSL	Secure Sockets Layer
TAN	Transaktionsnummer
ZertES	Bundesgesetz über Zertifizierungsdienste im Bereich der elektronischen Signatur
ECTS	European Credit Transfer System
IDE	Integrated development environment

22. Abildungsverzeichnis