Diplomarbeit

Authentisierung im mobilen Web

Zur Usability eID basierter Authentisierung auf einem NFC Handy

Dominik Oepen

Humboldt-Universität zu Berlin
Mathematisch-Naturwissenschaftliche Fakultät II
Institut für Informatik
Lehrstuhl für Systemarchitektur

Betreuer: Dr. rer. nat. Wolf Müller
Erstgutachter: Prof. Dr. rer. nat. Jens-Peter Redlich
Zweitgutachter: Prof. Dr. sc. nat. Hartmut Wandke

Berlin, den 1. September 2010

Selbstständigkeitserklärung:

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe.

Berlin, den ....
Datum Unterschrift

Einverständniserklärung:

Ich erkläre hiermit mein Einverständnis, dass die vorliegende Arbeit in der Bibliothek des Institutes für Informatik der Humboldt-Universität zu Berlin ausgestellt werden darf.

Berlin, den ....
Datum Unterschrift

Inhaltsverzeichnis

1. Einleitung

1.1. Motivation

Am 1. November 2010 wird in Deutschland der elektronischer Personalausweis (ePA) eingeführt werden. Dieser Ausweis wird einen RFID-Funkchip nach dem Standard ISO 14443 enthalten. Im Vergleich zum bisherigen Ausweisdokument wird er über zusätzliche Funktionen verfügen: Neben der hoheitlichen Identitätskontrolle soll auch eine Authentisierung des Ausweisinhabers über das Internet sowie das Leisten von qualifizierten elektronischen Signaturen mit dem elektronischen Personalausweis möglich sein.

Um die elektronischen Funktionen des ePA zu nutzen, wird ein passendes Lesegerät benötigt. Typischerweise ist dies ein RFID-Lesegerät, welches zum Zugriff auf den elektronischen Personalausweis die Anforderungen der technischen Richtlinie 03119 des Bundesamt für Sicherheit in der Informationstechnik (BSI) erfüllen muss. Eine weitere Möglichkeit ein Lesegerät zu realisieren, besteht in der Nutzung der Near Field Communication (NFC) Technologie. Diese Funktechnik ist kompatibel zu ISO 14443 und soll in Zukunft verstärkt Einzug in neue Mobiltelefone halten. So kündigte beispielsweise der finnische Handyhersteller Nokia an, all seine Smartphones ab dem Jahr 2011 mit NFC auszurüsten[1].

Der Grad der Nutzung des Internets per Mobiltelefon nimmt stetig zu. In Deutschland nutzten laut einer Studie von TNS Infratest[2] im Jahr 2008 bereits 16 Prozent der Bevölkerung das Internet von ihrem Handy aus. Das US-Marktforschungsinstitut Forrester Research prognostizierte 2008, dass der Anteil der Handybesitzer, welche von ihrem Mobiltelefon aus auch das Internet nutzen, in Europa bis zum Jahr 2013 auf 38 Prozent steigen werde[3]. Vor diesem Hintergrund erscheint die Nutzung der Authentisierungsfunktion des ePAs mit einem NFC-fähigen Handy als zukunftsträchtiger Anwendungsfall.

Als weiteres Indiz für diese Einschätzung kann man den Beitritt der Bundesdru-

[1] Near Field Communications World: *All new Nokia smartphones to come with NFC from 2011.* 17. Juni 2010. Verfügbar unter: http://www.nearfieldcommunicationsworld.com/2010/06/17/33966/all-new-nokia-smartphones-to-come-with-nfc-from-2011/. Letzter Zugriff: 15.08.2010

[2] TNS Infratest: *Mobiles Internet und seine Applikationen nutzen 16 Prozent der Deutschen.* 11.09.2008. Verfügbar unter: http://www.tns-infratest.com/presse/presseinformation.asp?prID=641. Letzter Zugriff: 15.08.2010

[3] Tecchanel.de: *Prognose: Bis 2013 hat jeder Vierte ein HSDPA-Handy.* 20.03.2008. Verfügbar unter: http://www.tecchannel.de/kommunikation/news/1751472/prognose_bis_2013_hat_jeder_vierte_ein_hsdpa_handy/ Letzter Zugriff: 15.08.2010

ckerei zum NFC Forum, dem Standardisierungsgremium für die NFC-Technologie, werten. Ulrich Hamann, Vorsitzender der Geschäftsführung der Bundesdruckerei, kommentierte den Beitritt folgendermaßen: „Wenn es uns gelingt, Mobiltelefone, die jeder selbstverständlich nutzt, und den elektronischen Personalausweis, den jeder permanent bei sich tragen wird, sinnvoll miteinander zu verknüpfen, wird das Schlagwort eID-Management eine ganz neue Bedeutung erhalten. [...] Unsere Idee, mobile eIdentity-Anwendungen zu ermöglichen, ist keine Zukunftsmusik.“[4]

In der Vergangenheit lief die Authentisierung eines Nutzers im Internet meist unter Zuhilfenahme von Passwörtern ab. Die Unzulänglichkeiten Passwort-basierter Authentisierungsmechanismen sind seit Langem bekannt: Menschen können sich nur eine begrenzte Zahl an Passwörtern merken und neigen vor allem dazu, leicht zu erratende Passwörter zu wählen. Dabei steigen die Anforderungen an die Länge und Komplexität von Passwörtern stetig. Passwörter erfüllen also weder die Kriterien guter Usability noch liefern sie ein hohes Maß an Sicherheit.

Wenn der elektronische Personalausweis klassische Passwort-basierte Verfahren ersetzen soll, so muss er diese beiden Anforderungen erfüllen - also bei gleichzeitiger, guter Usability ein hohes Maß an Sicherheit schaffen. „Die Menschen werden ihren elektronischen Ausweis nur dann auch online einsetzen, wenn dies einfach, komfortabel und sicher geschehen kann.“ heißt es dazu in einer Pressemeldung der Bundesdruckerei[5].

1.2. Ziel der Arbeit

In dieser Arbeit wird untersucht, wie die eID-Authentisierung des elektronischen Personalausweises auf einem Smartphone verwendet werden kann. Dazu wird sowohl die notwendige Hardware in Form eines NFC-fähigen Smartphones als auch die Software zur Nutzung der Funktionalität des elektronischen Personalausweises entwickelt.

Ziel der Arbeit ist es, die neue Authentisierungsmethode so gebrauchstauglich wie möglich umzusetzen. Zu diesem Zweck werden über den kompletten Entwicklungszyklus hinweg unterschiedliche Methoden des Usability-Engineerings eingesetzt. Eine abschließende Evaluation zeigt auf, inwiefern das Ziel einer gebrauchstauglichen Umsetzung erreicht wurde und welche Maßnahmen in Zukunft noch zur Verbesserung der Gebrauchstauglichkeit getroffen werden können.

[4] eGovernment Computing: *Der elektronische Personalausweis wird mobil.* 26. August 2009. Verfügbar unter: http://www.egovernment-computing.de/specials/it-infrastruktur/articles/226990/index3.html. Letzter Zugriff: 12.08.2010

[5] Bundesdruckerei GmbH: *Bundesdruckerei forciert komfortable Online-Nutzung von elektronischen Ausweisen.*06 Mai 2009. Verfügbar unter: http://www.bundesdruckerei.de/de/presse/presse_archiv/2009/pm_2009_05_06.html. Letzter Zugriff: 12.08.2010

1.3. Verwandte Arbeiten

Zum Zeitpunkt des Verfassens dieser Diplomarbeit beschäftigten sich bereits einige weitere Projekte mit der Verwendung des neuen Personalausweises zusammen mit NFC-fähigen Mobiltelefonen:

- In seiner Bachelorarbeit entwickelte Moritz Horsch eine J2ME-basierte Software für Nokias NFC-Telefone, um das PACE-Protokoll mit dem ePA durchzuführen [Hor09].
- In seiner Diplomarbeit beschrieb Ingo Kampe die Umsetzung von PACE und einigen Funktionen zum Zugriff auf den ePA für das Nokia 6212 [Kam10].
- Auf der Cebit 2010 stellte das Fraunhofer-Institut für Sicherheit in der Informationstechnik eine Konzeptstudie zur Nutzung des neuen Personalausweises mit Mobilfunkgeräten vor[6]. Zielplattform waren wiederum die NFC-Telefone von Nokia.

Weiterhin existieren auch Projekte, die sich mit Software für den elektronischen Personalausweis beschäftigen, ohne ein Handy als Zielplattform anzustreben. So lief beispielsweise an der Universität Koblenz-Landau das Projekt „rosecat“[7], dessen Ziel es war, eine Open-Source Software zur Nutzung der eID-Funktion des ePA zu entwickeln.

Die vorliegende Diplomarbeit unterscheidet sich von den anderen Projekten durch ihren Fokus auf die Untersuchung der Usability der eID-Authentisierung und die Verwendung von Methoden des Usability-Engineerings.

1.4. Aufbau der Arbeit

In Kapitel 2 werden die für das Verständnis dieser Arbeit benötigten Grundlagen beschrieben. Es werden sowohl grundlegende Begriffe aus dem Bereich der Usability als auch aus der IT-Sicherheit erläutert. Kapitel 3 widmet sich dem neuen Personalausweis. Es werden sowohl die neuen elektronischen Funktionen als auch das Sicherheitskonzept beschrieben. Darüber hinaus wird die eID-Authentisierung aus der Perspektive des Benutzers analysiert.

Kapitel 4 beschreibt eine erste Untersuchung zur gleichzeitigen Handhabung von Ausweis und Mobiltelefon. Die durch den Versuch gesammelten quantitativen und

[6] Fraunhofer-Institut SIT: *CeBIT 2010: neuer Personalausweis fürs Handy*. 25. Februar 2010. Verfügbar unter: http://www.sit.fraunhofer.de/pressedownloads/pressemitteilungen/20100225mobilenPA.jsp. Letzter Zugriff: 15.08.2010

[7] Universität Koblenz-Landau – Institut für Wirtschafts- und Verwaltungsinformatik: *Rosecat - rosecat open source ePA/eID client attains transparency*. Verfügbar unter: http://www.uni-koblenz-landau.de/koblenz/fb4/institute/iwvi/aggrimm/projekte/epa/rosecat. Letzter Zugriff: 15.08.2010

qualitativen Daten werden analysiert und diskutiert. In Kapitel 5 wird die Implementation der eID-Authentisierung auf einem Handy beschrieben. Teil dieser Implementation ist zum einen der Einbau eines RFID-Lesegeräts in das Handy, um die Datenübertragung mit dem Personalausweis zu realisieren. Zum anderen wird der Aufbau der Software-Architektur, welche im Rahmen dieser Diplomarbeit entwickelt wurde, beschrieben. Besonderes Augenmerk liegt dabei auf der grafischen Nutzeroberfläche, welche nach einem partizipativen Entwicklungsansatz umgesetzt wurde. Anschließend wird das mobile System in Kapitel 6 zusammen mit einer Vorabversion des Bürgerclients, der offiziellen Software für die Nutzung des elektronischen Personalausweises, einer heuristischen Evaluation unterzogen.

Kapitel 7 fasst abschließend die Ergebnisse dieser Arbeit zusammen und nennt mögliche Ansatzpunkte für weiterführende Arbeiten.

2. Grundlagen

2.1. Usability

Für den Begriff der Usability –zu Deutsch Gebrauchstauglichkeit – existiert eine Vielzahl von Definitionen. So vergleichen etwa Folmer, van Gurp und Jan Bosch in [FB04] acht verschiedene Definitionen miteinander und fassen ihre Ergebnisse in [FvGB03] in einer Tabelle zusammen (vergleiche Tabelle 2.1).

Overview of authors							
Constantine	**Hix**	**ISO9126**	**Nielsen**	**Preece**	**Shackel**	**Shneiderman**	**Wixon**
Learnability	Learnability	Learnability	Learnability	Learnability	Learnability	Time to learn	Learnability
Efficiency in use	Long-term performance	Operability	Efficiency of use	Throughput	Effectiveness	Speed of performance	Efficiency
Rememberability	Retainability	-	Memorability	-	Learnability	Retention over time	Memorability
Reliability in use		Operability	Errors	Throughput	Effectiveness	Rate of errors by users	Error rates
User satisfaction	Long-term user satisfaction	Attractiveness	Satisfaction	Attitude	Attitude	Subjective Satisfaction	Satisfaction

Tabelle 2.1.: Vergleich verschiedener Usability Definitionen. Quelle: [FvGB03]

Im Rahmen dieser Arbeit wird der Begriff Gebrauchstauglichkeit gemäß der Definition aus der ISO-Norm 9241 [ISO98] verwendet. Dort ist Gebrauchstauglichkeit wie folgt definiert:

> Gebrauchstauglichkeit: gibt an, inwieweit Benutzer mit einem Produkt ihre Ziele in einem definierten Nutzungskontext effektiv, effizient und zufriedenstellend erreichen können.

Die drei in dieser Definition verwendeten Komponenten Effektivität, Effizienz und Zufriedenheit sind dabei wiederum folgendermaßen definiert:

1. Effektivität: die Genauigkeit und Vollständigkeit, mit der ein Benutzer ein bestimmtes Ziel erreicht

2. Effizienz: der im Verhältnis zur Genauigkeit und Vollständigkeit eingesetzte Aufwand, mit dem Benutzer ein bestimmtes Ziel erreichen

3. Zufriedenheit: Freiheit von Beeinträchtigungen und positive Einstellung gegenüber der Nutzung des Produkts

Aus der Definition der Gebrauchstauglichkeit wird außerdem der Kontext ersichtlich, in welchem die Usability stattfindet: Ein Benutzer verwendet ein Produkt (etwa einen Computer), um bestimmte Ziele zu erreichen. Das Zusammenspiel der drei Parteien Aufgabe, Benutzer und Computer lässt sich durch das sogenannte ABC-Modell beschreiben.

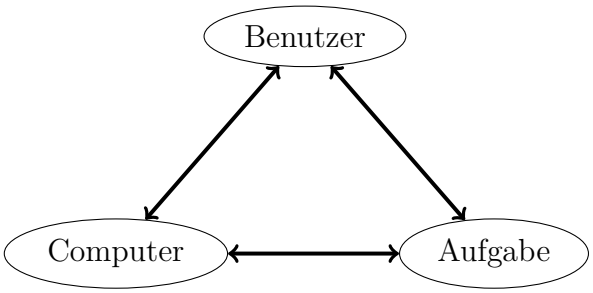

Abbildung 2.1.: Das ABC-Modell

Das Modell zeigt, dass nicht nur der Nutzer die Aufgabe mithilfe des Computers löst, sondern alle drei Parteien miteinander in Wechselwirkung stehen. So verändert beispielsweise der Einsatz eines Computers die Aufgabe selbst. Weiterhin verändert der Benutzer die Aufgabe durch seine Interpretation. Die Interaktion von Nutzer, Computer und Aufgabe findet dabei vor dem Hintergrund der Arbeitsumgebung statt. Deren Gestaltung ist Gegenstand der Arbeitspsychologie bzw. Arbeitswissenschaften und wird in dieser Arbeit nicht weiter betrachtet.

2.1.1. Benutzungsschnittstelle

Um ein gebrauchstaugliches System zu entwerfen, muss eine geeignete Schnittstelle entworfen werden, mittels welcher der Benutzer mit dem System interagieren kann. Diese Schnittstelle wird als Benutzungsschnittstelle bzw. User Interface bezeichnet. Die grafische Nutzeroberfläche ist dabei zwar der Teil der Schnittstelle, der für den Nutzer sichtbar ist, sie macht aber nur einen geringen Teil des User Interfaces aus und wird daher häufig als Spitze des Eisberges bezeichnet[1].

Thomas Moran stellte in [Mor81] ein Schichtenmodell der Benutzungsschnittstelle vor, welches aus drei Komponenten besteht: der konzeptuellen Komponente, der Kommunikationskomponente und der physikalischen Komponente. Jede dieser drei Komponenten ist wiederum in zwei Unterebenen aufgeteilt. Die konzeptuelle Komponente besteht aus der Aufgabenebene, welche die Aufgabe, die mit dem System gelöst werden soll, repräsentiert, und der semantischen Ebene, welche die dazu verwendeten

[1] Vergleiche: Dick Berry: *The iceberg analogy of usability.* 01. Oktober 2000. Verfügbar unter: http://www-106.ibm.com/developerworks/library/w-berry/. Letzter zugriff: 13.06.2010

Konzepte beinhaltet. Die Kommunikationsebene besteht aus der syntaktischen Ebene und der Interaktionsebene. Die syntaktische Ebene beschreibt dabei die genaue Form und die Interaktionsebene die Struktur der Dialoge, durch welche das System bedient wird. Die physikalische Komponente lässt sich schließlich in die räumliche Ebene und die Hardwareebene einteilen, wobei die Hardwareebene die tatsächlichen Ein- und Ausgabegeräte und die räumliche Ebene die Anordnung derselbigen beschreibt.

Konzeptuelle Komponente:	Aufgabenebene Semantische Ebene
Kommunikationskomponente:	Syntaktische Ebene Interaktionsebene
Physikalische Komponente:	Räumliche Ebene Hardwareebene

Tabelle 2.2.: Morans Schichtenmodell der Benutzungsschnittstelle

In der Praxis ist diese saubere Trennung der einzelnen Ebenen, insbesondere die Trennung der Benutzungsschnittstelle von der eigentlichen Anwendung, für den Benutzer meist nicht nachvollziehbar. Daher beschreibt beispielsweise van der Veer die Benutzungsschnittstelle im Sinne der kognitiven Ergonomie als „the complete and correct relevant knowledge of the combination of the user interface in the restricted sense and the application interface “ (siehe [vdV90] S.9). In dieser Definition wird die Benutzungsschnittstelle also als das benötigte Wissen des Benutzers aufgefasst, wobei sowohl das Wissen um die korrekte Bedienung (das sogenannte Interaktionswissen) als auch das für die korrekte Aufgabenlösung benötigte Anwendungswissen gemeint ist.

Die gebrauchstaugliche Gestaltung eines Systems hat nun die Erfüllung zweier Ziele zur Aufgabe: Zum einen gilt es, das zur Erfüllung einer Aufgabe benötigte Benutzerwissen zu minimieren, zum anderen das benötigte Wissen effektiv zu vermitteln.

2.1.2. Usability-Engineering

Um die Gebrauchstauglichkeit eines Systems sicherzustellen, ist die Berücksichtigung von Usability-Aspekten in allen Phasen der Entwicklung notwendig. Der Prozess des *Usability Engineerings* läuft daher eng verzahnt mit der klassischen Software-Entwicklung ab. Man kann grob vier Phasen der Entwicklung unterscheiden:

1. Analysephase: In dieser Phase gilt es, sowohl die Zielgruppe als auch die eigentliche Aufgabe eines Systems genau zu analysieren und daraus Anforderungen an das System abzuleiten.

2. Konzeptphase: In dieser Phase werden die Erkenntnisse aus der Analysephase zur Konzeption des zu entwickelnden Systems verwendet. Hierbei gilt es, sowohl

die benötigte Funktionalität als auch die entsprechende Gestaltung der Arbeitsabläufe zu definieren. Bereits in dieser frühen Phase können die Konzepte mit den späteren Nutzern erprobt werden, beispielsweise durch die Verwendung von Mock-Ups aus Papier.

3. Entwicklungsphase: Nach der Konzeption des Systems folgt die eigentliche Implementierung. Im Usability-Engineering wird hierbei zumeist ein iterativer Ansatz verfolgt. Das bedeutet, dass zunächst Prototypen des Systems entwickelt werden, welche anschließend getestet und analysiert werden, um danach als Ausgangspunkt für die nächste Iteration zu dienen.

4. Einführungsphase: Ist das System fertiggestellt, so muss es noch in der Praxis eingeführt werden. Eventuell ist dafür eine Schulung der Anwender und die schrittweise Einführung von Änderungen der Arbeitsabläufe notwendig. Ist das System in Betrieb genommen, so sollte eine abschließende Evaluation vorgenommen werden, um den Erfolg des neuen Systems zu kontrollieren und Ideen für mögliche Weiterentwicklungen zu sammeln.

Für jede dieser Phasen existieren zahlreiche Methoden des Usability-Engineerings. Die Methoden lassen sich dabei in *formative* und *summative* Methoden unterteilen. Formative Methoden zielen dabei auf konkrete Optimierungen während des Entwicklungsprozesses ab. Die Ergebnisse formativer Methoden fließen direkt wieder in die Entwicklung zurück. Summative Methoden werden dagegen meist auf abgeschlossene Produkte oder Protoypen angewandt. Sie dienen zum einen der abschließenden Bewertung zum anderen der Kontrolle des eigentlichen Entwicklungsprozesses.

Eine weitere Möglichkeit, die Methoden des Usability-Engineerings zu kategorisieren, ist die Einteilung in expertenorientierte und benutzerorientierte Methoden. Benutzerorientierten Methoden liegt dabei ein eher empirisches Vorgehen zugrunde. Hierbei werden die Endanwender eines Produktes befragt oder bei der Verwendung des Produktes beobachtet. Expertenorientierte Methoden sind eher analytischer Natur. Sie basieren auf der Untersuchung eines Produktes durch professionelle Usability-Evaluatoren auf der Basis etablierter Richtlinien oder der eigenen Erfahrung. Beide Kategorien haben ihre eigenen Vor- und Nachteile und werden daher meist in Kombination eingesetzt.

2.2. Identifizierung und Authentisierung

Eine grundlegende Aufgabe vieler IT-Systeme liegt darin, legitimen Benutzern Zugriff auf Ressourcen und Dienste des Systems zu gewähren (*permission problem*) und allen anderen Personen eben diesen Zugriff zu verweigern (*prevention problem*). Im Folgenden wird der Vorgang der *Identifizierung*, *Authentisierung* und *Autorisierung* einer Person beschrieben. Es sei jedoch darauf hingewiesen, dass derselbe Vorgang prinzipiell nicht nur für natürliche Personen, sondern auch allgemein für beliebige Entitäten – beispielsweise Computer – verwendet werden kann.

In der Identifizierungsphase macht die zu authentifizierende Person eine Behauptung darüber, wer sie ist, indem sie einen Identifikator präsentiert. Dabei handelt es sich um ein nicht-geheimes Merkmal einer Person, wie zum Beispiel einen Benutzernamen oder eine E-Mail-Adresse. Es sei darauf hingewiesen, dass eine allgemein anerkannte Definition des Begriffes Identität bisher nicht existiert. So wird beispielsweise in [BSSW10, S. 14 ff] eine ganze Reihe von Begriffsdefinitionen aus unterschiedlichen Perspektiven (sozial, rechtlich, technisch) vorgestellt.

In der anschließenden Authentisierungsphase muss die Person beweisen, dass sie tatsächlich der rechtmäßige Inhaber des Identifikators ist. Ein solcher Nachweis kann wiederum auf drei verschiedene Arten erfolgen: über etwas, was eine Person weiß, zum Beispiel ein Passwort, über etwas, das eine Person besitzt, wie etwa eine Chipkarte, oder über etwas, das eine Person ist. Der letzte Punkt bezieht sich auf Biometrie-gestützte Authentisierungsverfahren, beispielsweise anhand von Fingerabdrücken.

Um ein höheres Schutzniveau zu erreichen, werden oftmals mehrere dieser Verfahren miteinander kombiniert. Eine weit verbreitete Variante ist z. B. die Kombination von Wissens- und Besitz-basierten Mechanismen in der Form einer Smartcard, welche wiederum über eine PIN gesichert ist. Man spricht in diesem Fall von *Mehrfaktorauthentisierung*.

In der letzten Phase, der Autorisierung, werden einer Person Berechtigungen für den Zugriff auf Daten oder Funktionen innerhalb des Systems gewährt. Idealerweise verfügt ein Benutzer lediglich über diejenigen Berechtigungen, die für ihre aktuelle Aufgabe notwendig sind. In der Praxis ist die Rechtevergabe meist jedoch sehr viel grobgranularer.

2.2.1. Authentisierungsmechanismen

Die oben beschriebenen drei Klassen von Authentisierungsmechanismen – wissensbasiert, besitzbasiert und biometrisch – haben unterschiedliche Eigenschaften bezüglich Sicherheit und Gebrauchstauglichkeit.

Am weitesten verbreitet sind nach wie vor wissensbasierte Mechanismen, allen voran die Authentisierung mittels Passwörtern. Ursache dafür ist die einfache Umsetzung dieser Mechanismen. Aufgrund ihrer weiten Verbreitung ist die Usability von Passwörtern sehr gut untersucht. So zeigt etwa der Artikel *Users are not the enemy — Why users compromise computer security and how to take remedial measures* [AS99] von Anne Adams und Martina Angela Sasse eine Vielzahl von Problemen in der Gebrauchstauglichkeit passwortbasierter Authentisierungsmechanismen auf. Als Ursache für die meisten Probleme werden in diesem Artikel ungeeignete Sicherheitspraktiken genannt. Dazu zählen etwa die Notwendigkeit, Passwörter regelmäßig zu ändern oder aber die Vielzahl von Passwörtern, über die jeder einzelne Benutzer verfügen muss. Durch diese Maßnahmen wird den Nutzern eine hohe Gedächtnislast (*cognitive load*) auferlegt, welche die Unterwanderung der Sicherheitsanforderungen durch die Wahl schwacher Passwörter oder das Aufschreiben der Passwörter zur Folge hat.

Die Gedächtnisleistung, welche dem Benutzer durch Passwörter aufgebürdet wird,

ist zum Teil beträchtlich. In dem Artikel *Transforming the 'weakest link' — a human/computer interaction approach to usable and effective security* [SBW01] berichten Sasse et al., dass in einem untersuchten Unternehmen jeder Nutzer im Durchschnitt über nicht weniger als 16 Passwörter verfügte. Gleichzeitig steigen die Anforderungen an die Komplexität von Passwörtern stetig. Der Grund dafür ist, dass mit der stetig wachsenden Rechenleistung moderner Computer auch die Fähigkeit steigt, Passwörter durch systematisches Ausprobieren von Kombinationen zu erraten (*Dictionary Angriff*, bzw. *Brute Force Angriff*). Ross Anderson fasst die Problematik der Auswahl eines sicheren Passwortes in [And08, S. 32] folgendermaßen zusammen: „Chosse a password you can't remember, and don't write it down".

Eine Alternative zur Erhöhung der geforderten Passwortkomplexität besteht darin, die Anzahl möglicher Versuche, ein Passwort zu erraten, zu begrenzen. Bei Chipkarten ist es beispielsweise üblich, die Karte nach drei fehlerhaften PIN-Eingaben zu sperren. Aus diesem Grund können PINs wesentlich weniger komplex sein (üblicherweise nur wenige Dezimalziffern) als andere Passwörter. Brostoff und Sasse schlagen schlagen allerdings auch in diesem Bereich vor, die Anzahl erlaubter Fehlversuche auf zehn zu erhöhen, um die Erfolgsquote der Nutzer zu steigern und die Kosten für Support zu senken (vergleiche [BS03]). Viega und McGraw hingegen schlagen vor, einen zusätzlichen Fehlversuchszähler zu etablieren, um die gesamte Anzahl fehlerhafter Authentisierungsversuche zu protokollieren (zusätzlich zu der Anzahl der Versuche pro Sitzung) ([VM02]).

Besitz-basierte Authentisierungsmechanismen bürden dem Benutzer keine so hohe Gedächtnislast auf. Er muss allerdings daran denken, den notwendigen Identifikator mit sich zu tragen. Dies könnte bei einer stärkeren Verbreitung Besitz-basierter Authentisierungsmechanismen zum Problem werden, nämlich dann, wenn der Nutzer einen eigenen Identifikator pro Dienst benötigt. Schon heute ist die Anzahl der im Umlauf befindlichen Chipkarten beträchtlich. Mögliche Angriffe auf Identifikatoren sind zum einen Diebstahl, zum anderen das Klonen eines Identifikators. Letzteres kann entweder durch Extraktion der Schlüssel (vergleiche beispielsweise [LLG+05]) oder durch das Ausnutzen eventuell vorhandener Schwächen in den verwendeten Algorithmen und Protokollen (siehe z.B. [Plö08]) erfolgen.

Biometrische Authentisierungsmerkmale können vom Benutzer nicht vergessen werden und auch der Verlust (beispielsweise in Form des Verlusts eines Fingers) ist relativ selten. Der Nachteil ist, dass diese Merkmale nicht versteckt werden können und sie durchaus kopierbar sind. So können etwa Fingerabdrücke, welche auf glatten Flächen – wie etwa Gläsern oder Flaschen – hinterlassen werden[2], einfach vervielfältigt werden. Problematisch daran ist vor allem, dass sich biometrische Merkmale nicht verändern lassen, auch wenn ein Missbrauch bekannt werden sollte. Des Weiteren wenden Kritiker biometrischer Authentisierungsverfahren ein, dass die Überprüfung von Fingerabdrücken diskriminierend sei, da ein nicht unerheblicher Teil der Bevölkerung – vor allem ältere Personen – nicht über genügend ausgeprägte Fingerabdrücke

[2] Eine Anleitung zum Kopieren eines Fingerabdrucks findet sich etwa unter `http://dasalte.ccc.de/biometrie/fingerabdruck_kopieren` [25.04.2010]

für eine Verifikation verfügt. Ganz allgemein erzielen biometrische Authentisierungsverfahren niemals absolut korrekte Ergebnisse, so dass bei der Bewertung eines bestimmten System immer auch Fehlerraten, wie etwa die False Acceptance Rate (FAR) oder die False Rejection Rate (FRR), berücksichtigt werden müssen.

2.3. RFID und NFC

Radio Frequency Identification (RFID) Techniken werden, wie der Name bereits vermuten lässt, zur Identifikation von Objekten via Funk verwendet. RFID-Systeme bestehen grundsätzlich aus zwei Parteien: einem *Transponder* und einem Lesegerät (bzw. *reader*). Der Transponder wird in der Praxis häufig auch als *Tag* bezeichnet.

Bei RFID handelt es sich allerdings nicht um ein einzelnes Funksystem, sondern vielmehr um einen Oberbegriff für eine Reihe von Techniken. Die verschiedenen Systeme unterscheiden sich dabei zum einen in den Parametern des verwendeten Funkkanals – wie etwa Frequenz, Sendestärke (und damit Reichweite) und Kodierung – zum anderen in der Art und Weise der Identifizierung. Tabelle 2.3 zeigt eine Klassifizierung von RFID-Systemen des Bundesamt für Sicherheit in der Informationstechnik (BSI). Als Hauptunterscheidungsmerkmal dient dabei die verwendete Frequenz. Auf die Aufnahme von Systemen, die im Mikrowellenbereich operieren, wurde in unten stehender Tabelle aus Platzgründen verzichtet.

Parameter	Niedrigfrequenz	Hochfrequenz	Ultrahochfrequenz
Frequenz	125 - 134 KHz	13,56 MHz	868 bzw. 915 MHz
Leseabstand	bis 1,2 m	bis 1,2 m	bis 4 m
Lesegeschwindigkeit	langsam	mittel bis schnell	schnell
Heutige ISO-Standards	11784, 11785 und 14223	14443, 15693 und 18000	14443, 15693 und 18000

Tabelle 2.3.: Eigenschaften verschiedener RFID-Techniken. Frei nach [BSI04, S. 29]

Grundsätzlich unterscheidet man zwischen aktiven und passiven RFID-Tags. Aktive Tags verfügen über eine eigene Stromversorgung, beispielsweise in Form einer Batterie. Sie erreichen meist eine höhere Sendereichweite, sind aber teurer als passive RFID-Tags. Letztere beziehen ihren Strom mittels elektromagnetischer Kopplung vom Lesegerät.

Ein weiteres Unterscheidungsmerkmal der verschiedenen RFID-Klassen ist die Komplexität der Datenverarbeitung, zu welcher die Transponder imstande sind. Das Spektrum reicht hierbei von extrem einfachen Tags, welche lediglich in der Lage sind, eine eindeutige Seriennummer – der Unique Identifier (UID) – auszusenden, bis hin zu den sogenannten *kontaktlosen Smartcards*. Letztere haben dabei meist die

Form klassischer kontaktbehafteter Mikrochipkarten; das in ISO 7816-2 definierte ID-1 Format.

Viele der kontaktlosen Smartcards basieren auf der Standardfamilie ISO 14443. Zur Datenübertragung nimmt die Karte (Proximity Integrated Circuit Card (PICC)) eine Lastmodulation des vom Lesegerät (Proximity Coupling Device (PCD)) erzeugten elektromagnetischen Feldes vor. Die Frequenz des Feldes beträgt dabei 13,56 MHz. Der Standard definiert zwei verschiedene Typen der Funkübertragung, welche sich unter anderem in Modulation und Kodierung unterscheiden. Auf die technischen Details soll hier nicht weiter eingegangen werden. Weitergehende Informationen dazu finden sich beispielsweise in [Plö08] und [Fin08]. Das Anwendungsprotokoll, welches in ISO 14443 Part 4 (siehe [ISO00]) beschrieben ist, entspricht zu weiten Teilen dem in ISO 7816 Part 4 (siehe [ISO05]) definierten Anwendungsprotokoll kontaktbehafteter Smartcards. Hierbei kommunizieren Lesegerät und Chipkarte in einem blockorientierten request-response Protokoll in Form von Application Protocol Data Units (APDUs) genannten Nachrichten.

Die Near Field Communication (NFC)-Technologie setzt auf ISO 14443 auf und ist daher vom Funkkanal her kompatibel. Ein NFC-Gerät kann in drei unterschiedlichen Modi betrieben werden:

1. Reader / Writer Mode: In diesem Modus agiert das Gerät als PCD und kann somit als Lesegerät für RFID-Tags nach ISO 14443 verwendet werden.

2. Card Emulator Mode: In diesem Modus agiert das Gerät als PICC.

3. Peer-to-peer Mode: In diesem Modus stellen beide an der Kommunikation beteiligten NFC-Geräte ihr eigenes Feld zur Verfügung. Sie werden also beide im aktiven Modus betrieben.

NFC wurde hauptsächlich für den Einsatz in Mobiltelefonen entwickelt. Als Anwendungsszenarien sind vor allem Micropayment und mobile Ticketing Systeme anvisiert. NFC kommt bisher in verschiedenen Pilotprojekten zum Einsatz, wie etwa dem Touch-and-Travel Projekt der Deutschen Bahn[3]. Der peer-to-peer Modus kann als Alternative zu Bluetooth oder ähnlichen Funktechnologien verwendet werden. Ein möglicher Anwendungsfall ist hier der einfache Austausch von Kontaktinformationen.

[3] Deutsche Bahn AG: *Touch & Travel – einfach einsteigen und losfahren.* Verfügbar unter: `http://www.touchandtravel.de`. Letzter zugriff: 03.06.2010

3. Der elektronische Personalausweis

Im Folgenden werden die Begriffe neuer Personalausweis (nPA), elektronischer Personalausweis (ePA) und Ausweis synonym verwendet. Wenn der frühere Ausweis ohne Funkchip gemeint ist, so wird dieser als „alter Personalausweis“ bezeichnet.

Nach [Sch09] ist ein Ausweisdokument ein „privates oder amtliches Dokument in einem kleinen Format, das die Identität des Inhabers belegt und diesem (optional) bestimmte Rechte bescheinigt“. Der neue Personalausweis ist ein Ausweisdokument in diesem Sinne. Abweichend vom bisherigen Personalausweis wird der neue Ausweis das Format td-1 gemäß [ICA08] haben. Dies entspricht dem ID-1 Format herkömmlicher Chipkarten.

Abbildung 3.1.: Muster des neuen Personalausweises. Quelle: Personalausweisverordnung (PAuswV)[1]

Welche Daten auf dem neuen Ausweis aufgedruckt sind, ist in § 5 Absatz 2 Personalausweisgesetz (PAuswG)[2] festgelegt:

1. Familienname und Geburtsname
2. Vorname
3. Doktorgrad
4. Tag und Ort der Geburt

[1] Verordnung über Personalausweise und den elektronischen Identitätsnachweis in der Fassung vom 22.04.2010. Geltung ab 01.11.2010.

[2] Gesetz über Personalausweise und den elektronischen Identitätsnachweis in der Fassung vom 18.06.2009 (BGBl. I S. 1346). Geltung ab 01.11.2010, § 21 gilt ab 01.05.2010.

5. Lichtbild
6. Unterschrift
7. Größe
8. Farbe der Augen
9. Anschrift, bei Anschrift im Ausland die Angabe „keine Hauptwohnung in Deutschland“
10. Staatsangehörigkeit
11. Seriennummer
12. Ordensname, Künstlername

Auf der Rückseite befindet sich weiterhin die sogenannte Machine Readable Zone (MRZ). In dieser Zone ist ein Teil der auf dem Ausweis aufgedruckten Daten noch einmal in einer speziellen maschinenlesbaren Schrift abgebildet. Das Format der maschinenlesbaren Zone ist in [ICA08, S. 45] standardisiert. Die MRZ des neuen Personalausweises enthält die folgenden Daten: die Zeichenfolge „IDD“, Familienname, Vorname, Seriennummer, die Abkürzung „D“, Tag der Geburt, letzter Tag der Gültigkeitsdauer. Das Zeichen „<“ wird zum Auffüllen dieser Datenfelder auf feste Längen und als Trennzeichen verwendet.

Die primäre Funktion des nPA wird weiterhin die Verwendung als *visueller Ausweis* sein. Das bedeutet, dass die Prüfung des Ausweises visuell ohne Hinzunahme eines zusätzlichen Gerätes erfolgen kann. Der Ausweis ist somit auch ohne den integrierten Chip gültig, beispielsweise im Falle eines Defekts: „Störungen der Funktionsfähigkeit des elektronischen Speicher- und Verarbeitungsmediums berühren nicht die Gültigkeit des Personalausweises.“ [PAuswG, § 28, Absatz 3]. Die Bindung zwischen Ausweisinhaber und Ausweisdokument wird in diesem Fall durch das Foto des Ausweisinhabers auf dem Personalausweis hergestellt. Die Authentizität des Ausweises, also die Tatsache, dass es sich um ein korrektes, hoheitliches Ausweisdokument handelt, kann in diesem Fall über die physikalischen Sicherheitsmerkmale überprüft werden. Dabei handelt es sich um Merkmale des Kartenkörpers, welche nur schwer zu fälschen sind, wie etwa Oberflächenprägung, Guillochen oder Laserbeschriftung. Zum Zeitpunkt der Erstellung dieser Arbeit waren die physikalischen Sicherheitsmerkmale des neuen Personalausweises noch nicht öffentlich bekannt.

3.1. Elektronische Funktionen des neuen Personalausweis

Zusätzlich zur Funktion als visuelles Ausweisdokument wird der neue Personalausweis über drei elektronische Applikationen verfügen: die ePass-Funktion, die eID-Funktion

und die eSign-Funktion. Von diesen drei Applikation ist lediglich die ePass-Funktion verpflichtend vorhanden. Die eID-Funktion ist optional und kann jederzeit auch nachträglich gegen eine Gebühr in den Meldeämtern aktiviert oder deaktiviert werden. Zur Nutzung der eSign-Funktion ist das Nachladen eines *qualifizierten Zertifikates* von einem geeigneten Zertifizierungsdiensteanbieter (ZDA) notwendig. Alle drei Applikationen werden im Folgenden genauer erläutert.

3.1.1. Die ePass-Applikation

Die ePass-Funktion (auch Biometriefunktion genannt) ist für hoheitliche Zwecke vorgesehen, also etwa für die Ausweiskontrolle durch Polizei- oder Grenzbeamte. Sie dient der elektronischen Überprüfung der Gültigkeit des Ausweisdokuments und ergänzt die visuelle Inspektion durch die Beamten. Die Applikation ist kompatibel zur ePass-Applikation im elektronischen Reisepass, welche durch [ICA08] standardisiert ist. Dieser Standard definiert eine Datenstruktur (*Logical Data Structure*) mit einer Reihe von obligatorischen oder optionalen Datengruppen.

Der nPA enthält lediglich eine Teilmenge dieser Daten (siehe Tabelle 3.1). Im Gegensatz zum elektronischen Reisepass ist die Speicherung von Fingerabdrücken beim neuen Personalausweis nicht verpflichtend, sie geschieht nur auf expliziten Wunsch des Ausweisinhabers [3]. Dies kann als Reaktion auf die starke Kritik an der Verwendung von Fingerabdrücken im elektronischen Reisepass gesehen werden.

Datei	Inhalt
`EF.COM`	Inhaltsverzeichnis der vorhandenen Datengruppen
`EF.SOD`	Signierte Hashwerte der Datengruppen
`DG1`	MRZ
`DG2`	Digitales Gesichtsbild
`DG3`	Optional: Zwei Fingerabdrücke

Tabelle 3.1.: Datengruppen der ePass-Applikation

Die visuelle Gültigkeitsprüfung eines neuen Personalausweises kann durch die Verwendung der ePass-Funktion um eine elektronische Prüfung ergänzt werden. Zusätzlich zu den Authentizitätsprüfungen durch die *General Authentication Procedure* (siehe Abschnitt 3.2) kann so eine Überprüfung der Integrität der auf dem Chip gespeicherten Daten durch die Verifikation der Signatur des `EF.SOD` im Rahmen der passiven Authentisierung (siehe Unterabschnitt 3.2.3) erfolgen. Darüber hinaus kann der Beamte, der das Ausweisdokument prüft, eine Authentisierung des Ausweisinhaber durch einen Abgleich des Gesichtsbildes sowie eventuell vorhandener Fingerabdrücke vornehmen. Das digitale Gesichtsbild hat dabei den Vorteil, dass es in hoher Auflösung vorliegt und somit einen besseren Abgleich erlaubt, als das recht kleine auf dem Ausweis abgedruckte Foto.

[3] vgl. PAuswG, § 9, Absatz 3

3.1.2. eID-Applikation

Die eID-Funktion dient der elektronischen Authentisierung des Nutzers im nichthoheitlichen Kontext. Die Applikation wurde für die Verwendung im Rahmen von eBusiness und eGovernment Anwendungen konzipiert. Explizit vorgesehen ist die Möglichkeit einer Authentisierung über das Internet.

Die eID-Applikation enthält genau wie die ePass-Applikation eine Reihe von Datengruppen, welche in Tabelle 3.2 abgebildet sind.

Datengruppe	Inhalt
DG1	Dokumenttyp
DG2	Ausgebender Staat
DG3	Ablaufdatum
DG4	Vorname
DG5	Familienname
DG6	Ordensname/Künstlername
DG7	Doktorgrad
DG8	Geburtsdatum
DG9	Geburtsort
DG17	Adresse
DG18	Wohnort-ID
	Vergleichgeburtsdatum für Altersverifikation
	Schlüssel für sektorspezifisches Sperrmerkmal
	Schlüssel für sektorspezifische Kennung

Tabelle 3.2.: Datengruppen der eID-Applikation

Zusätzlich zu den Datengruppen eins bis neun bzw. 17 und 18, umfasst die eID-Anwendung noch vier spezielle Funktionen: die Altersverifikation, die Wohnortabfrage (*Community-ID Verification*), das sektorspezifische Sperrmerkmal und die sektorspezifische Kennung (*Restricted Identification*).

Die **Altersverifikation** erlaubt die Überprüfung, ob der Ausweisinhaber alt genug ist, um einen Dienst in Anspruch zu nehmen, ohne die Übermittlung des tatsächlichen Geburtstages des Ausweisinhabers. Der Dienst übermittelt dafür ein Referenzdatum an den Ausweis. Dieser überprüft, ob das Geburtsdatum des Ausweisinhabers vor diesem Datum liegt und gibt eine positive oder negative Rückmeldung. Ein systematisches Ermitteln des tatsächlichen Geburtsdatums wird dadurch verhindert, dass lediglich ein Referenzdatum im Rahmen der *General Authentication Procedure* übertragen werden kann. Will ein Dienst die Überprüfung mit einem anderen Referenzdatum durchführen, so ist ein erneutes Durchführen dieser Prozedur und somit eine erneute Eingabe der PIN durch den Benutzer notwendig.

Die **Wohnortabfrage** erlaubt es, einen Dienst nur für die Bewohner einer bestimmten geografischen Region zugänglich zu machen, ohne die genaue Adresse der Nutzer auszulesen. Die Grundlage für die Wohnortabfrage bildet die in `DG18` gespeicherte *Wohnort-ID*. Hierbei handelt es sich um eine Ziffernfolge von 14 Dezimalzahlen, welche die folgenden Informationen codieren:

- Ländercode gemäß ISO 3611-1 (drei Ziffern ergänzt um eine führende Null)
- Bundesland (zwei Ziffern)
- Regierungsbezirk (eine Ziffer ergänzt um eine führende Null)
- Stadtkreis (zwei Ziffern)
- Gemeinde (Drei Ziffern ergänzt um eine führende Null)

Die Codierung aller Angaben – bis auf den Ländercode – erfolgt nach dem amtlichen Gemeindeschlüssel des statistischen Bundesamtes. Die Verifikation der Wohnort-ID kann gemäß allen Gliederungsstufen dieser Hierarchie erfolgen. Hierzu übermittelt der Dienstanbieter eine Referenz ID, welche von links nach rechts mit den vorderen Ziffern der im Ausweis gespeicherten *Wohnort-ID* verglichen wird. Folgendes Beispiel verdeutlicht die Funktionsweise: Will ein Dienstanbieter einen Dienst für die Einwohner des Bundeslandes Niedersachsen anbieten, so würde er die Referenz-ID 0276 03 übermitteln. Dies würde zu einer positiven Auskunft bei einem Bewohner der Stadt Hildesheim (*Wohnort-ID* 0276 03 02 54 0021) und einer negativen Auskunft bei einem Bewohner der Stadt Stuttgart (*Wohnort-ID* 0276 08 01 11 0000) führen. Analog zur Altersverifikation ist lediglich die Übermittlung eines einzigen Referenzwertes pro Durchlauf der *General Authentication Procedure* möglich, so dass die Bestimmung der genauen *Wohnort-ID* durch systematisches Abfragen verhindert wird.

Die **sektorspezifische Kennung** ist eine Funktion, welche es einem Dienstanbieter erlaubt, einem Nutzer ein Pseudonym zuzuweisen und ihn daran wiederzuerkennen, ohne weitere personenbezogene Daten über ihn zu erheben. Die erzeugten Pseudonyme sind dabei für den Dienstanbieter eindeutig. Allerdings sind die Pseudonyme, welche von einem Ausweis für verschiedene Dienstanbieter erzeugt werden, nicht miteinander verknüpfbar. Das bedeutet, dass ein Benutzer, dem durch seinen Ausweis bei einem Dienst ein bestimmtes Pseudonym zugeordnet ist, bei einem anderen Dienst ein vollkommen anderes Pseudonym erhält. Den beiden Dienstanbietern ist es dabei nicht möglich herauszufinden, ob die Pseudonyme zu ein und demselben Ausweisinhaber gehören.

Das **Sperrmerkmal** dient dazu, dass ein Dienstanbieter Ausweise, welche auf Grund von Verlust oder Diebstahl gesperrt wurden, erkennen kann. Genau wie bei der *Restricted Identification* ist das Sperrmerkmal ein dienstanbieterspezifisches Pseudonym. Jeder Dienstanbieter verfügt dabei über eine eigene Sperrliste, welche die

Pseudonyme aller gesperrten Ausweise enthält. Der Ausweisinhaber kann die Sperrung seines Ausweises unter Angabe eines im PIN/PUK Brief enthaltenen Sperrkennworts über eine Telefon-Hotline oder direkt bei den Ausweisbehörden veranlassen.
Ein wichtiger Unterschied zwischen eID-Anwendung und ePass-Anwendung besteht darin, dass die Daten der eID-Anwendung nicht signiert sind. Es gibt also kein Äquivalent zum `EF.SOD`. Dies geschieht aus Datenschutzgründen: Es macht das Weiterverkaufen von Daten, welche aus dem Ausweis ausgelesen wurden, weniger lukrativ, da ein Verkäufer nicht beweisen kann, ob die von ihm angebotenen Daten authentisch sind.

3.1.3. eSign-Funktion

Die eSign-Funktion ermöglicht es, den neuen Personalausweis zu nutzen, um eine Qualifizierte Elektronische Signatur (QES) im Sinne des deutschen Signaturgesetzes[4] zu leisten.

Eine QES ist eine fortgeschrittene elektronische Signatur, die auf einem (zum Zeitpunkt ihrer Erzeugung gültigen) qualifizierten Zertifikat beruht und mit einer sicheren Signaturerstellungseinheit (SSEE) erstellt wurde[5]. Dabei bezeichnet der Begriff *fortgeschrittene elektronische Signatur* eine elektronische Signatur, welche ausschließlich dem Signierenden zugeordnet ist, den Signierenden identifiziert und mit den Daten, auf die sie sich bezieht, so verknüpft ist, dass eine nachträgliche Veränderung der Daten erkannt werden kann. Des Weiteren muss der Signierende die Mittel zur Signaturerstellung unter seiner alleinigen Kontrolle haben. Diese letzte Anforderung wird unter anderem durch die Verwendung einer SSEE gewährleistet. Hierbei handelt es sich um „Software- oder Hardwareeinheiten zur Speicherung und Anwendung des jeweiligen Signaturschlüssels“, welche „die Fälschungen der Signaturen und Verfälschungen signierter Daten zuverlässig erkennbar machen und gegen unberechtigte Nutzung der Signaturschlüssel schützen“. Bei der Nutzung der eSign-Funktion dient der Personalausweis als SSEE. Die optionale Verwendung des neuen Personalausweises als sichere Signaturerstellungseinheit ist Teil der Chipkarten Strategie der Bundesregierung (eCard Strategie)[6]. Dieses Strategiepapier enthält unter anderem folgenden Absatz:

> Alle ausgegebenen Karten werden von vorne herein für die optionale Aktivierung von qualifizierten Zertifikaten zur Erzeugung von qualifizierten elektronischen Signaturen vorbereitet.

[4] Gesetz über Rahmenbedingungen für elektronische Signaturen (Signaturgesetz – SigG) vom 16. Mai 2001 (BGBl. I S. 876), das zuletzt durch Artikel 4 des Gesetzes vom 17. Juli 2009 (BGBl. I S. 2091) geändert worden ist

[5] vgl. SigG, § 2, Absatz 3

[6] *Chipkarten-Strategie der Bundesregierung (eCard-Strategie)*. Verfügbar unter: `http://www.einblick.dgb.de/hintergrund/2008/13/chipkartenstrategie.pdf`. Letzter Zugriff: 10.07.2010

Möchte der Ausweisinhaber die eSign-Funktion nutzen, so muss er zusätzlich zum Ausweis bei einem geeigneten ZDA ein so genanntes *qualifiziertes Zertifikat* erwerben. An die Vergabe dieser Zertifikate wird eine Reihe von Anforderungen gestellt, welche in § 5 des Signaturgesetzes ausformuliert sind. Eine weitere Besonderheit dieser Zertifikate ist, dass sie selbst über eine qualifizierte elektronische Signatur verfügen müssen.

Die Anforderung, welche sich aus SigG und SigV für die Verwendung von kontaktlosen Smartcards als Sichere Signaturerstellungseinheit ergeben, werden in der technischen Richtlinie 03117 [BSI09d] beschrieben.

3.2. Schutzmechanismen

Um die Gewährleistung der Schutzziele Integrität und Vertraulichkeit der Kommunikation, sowie Authentizität der beteiligten Kommunikationspartner bei der Verwendung des elektronischen Personalausweis sicherzustellen, kommen vier kryptografische Protokolle zum Einsatz: das Password Authenticated Connection Establishment (PACE) Protokoll, die passive Authentisierung, die Terminal Authentication (TA) und die Chip Authentication (CA). Die Ausführung der Protokolle in dieser Reihenfolge wird als General Authentication Procedure bezeichnet und ist in der technischen Richtlinie 03110 [BSI10a] beschrieben.

Um die einzelnen Protokolle nachvollziehen zu können, müssen zunächst die einzelnen Kommunikationsteilnehmer bei der Nutzung des elektronischen Personalausweises identifiziert werden. Grundsätzlich kommuniziert der Ausweis mit einem Terminal. Es werden drei Arten von Terminals unterschieden analog zu den drei Applikation des nPA: Inspektionssysteme für die ePass-Anwendung, Authentisierungsterminals für die eID-Funktion und Signaturterminals für die eSign-Funktion. Terminals stellen zum einen die Funkverbindung zum Ausweis her, zum anderen stellen sie den eigentlichen Dienst zur Verfügung. Im Falle der eID-Anwendung können diese beiden Funktionen von unterschiedlichen Geräten erbracht werden. In diesem Falle unterscheidet man das lokale Terminal, welches aus dem Lesegerät und dem Computer des Nutzers besteht und mit dem Ausweis kommuniziert, und dem entfernten Terminal, welches den eigentlichen Dienst erbringt. Die beiden Terminals kommunizieren in diesem Fall über das Internet miteinander.

3.2.1. PACE

Der im nPA integrierte Chip verfügt über eine Funkschnittstelle gemäß ISO-14443a [ISO00]. Die Verwendung einer Funkschnittstelle bringt eine Reihe von sicherheitstechnischen Problemen mit sich. Der Grund dafür ist, dass bei Funkübertragungen ein *geteiltes Medium* verwendet wird. Das bedeutet, dass grundsätzlich jede Person mit dem geeigneten Funkzubehör die Datenübertragung abhören und gegebenenfalls auch selber aktiv Daten einspielen kann. Voraussetzung hierfür ist, dass sich der Angreifer in Reichweite der Funkübertragung befinden muss. Die Funkreichweite hängt

von einer Reihe von Faktoren ab, wie etwa der verwendeten Frequenz, Antenne, Sendeleistung, etc. In [FK04] wird das Auslesen eines ISO 14443 konformen RFID Chips über eine maximale Distanz von 3 Metern beschrieben. Das schweizer Bundesamt für Kommunikation berichtet im Messbericht zur Abklärungen über die Datenauslesung auf Distanz beim biometrischen Pass[7] davon, das Signal einer aktiven Kommunikation zwischen Lesegerät und RFID Chip noch über eine Distanz von 25 Metern erkannt zu haben.

Zur Absicherung der Luftschnittstelle wird daher das PACE Protokoll verwendet. PACE dient dazu, einen gesicherten Kanal zwischen zwei Parteien auf der Grundlage eines schwachen, geteilten Geheimnisses zu etablieren. Hierzu werden zwei starke Sitzungsschlüssel für Verschlüsselung und Integritätssicherung der Daten vereinbart. PACE ist der Nachfolger des Basic Access Control (BAC) Protokolls, welches beim elektronischen Reisepass verwendet wird.

PACE wurde zum ersten Mal in [UKN+08] beschrieben. In [BFK09] wurde ein Sicherheitsbeweis von PACE innerhalb des real-or-random Modells vorgestellt.

Zur Durchführung von PACE müssen die beiden beteiligten Parteien über ein geteiltes Geheimnis verfügen. Für die Verwendung mit dem neuen Personalausweis sind die folgenden Geheimnisse vorgesehen:

Die **eID-PIN** ist eine sechsstellige Dezimalzahl, welche durch den Nutzer frei gewählt werden kann. Der elektronische Personalausweis wird mit einer zufälligen, fünfstelligen Transport-PIN ausgeliefert. Diese kann lediglich zum Ändern der eID-PIN auf einen vom Nutzer gewählten Wert verwendet werden. Im Folgenden werden die Begriffe eID-PIN und PIN synonym verwendet. Es sei jedoch darauf hingewiesen, dass zur Verwendung der eSign-Funktion eine gesonderte PIN, die Signatur-PIN, verwendet wird. Diese PIN wird allerdings nicht zur Durchführung von PACE verwendet, sondern in einem zuvor etablierten gesicherten Kanal direkt an den Ausweis übertragen.

Der **PIN Unblocking Key (PUK)** ist eine zehnstellige, zufällige Dezimalzahl, welche dazu verwendet wird, eine PIN nach dreimaliger Fehleingabe wieder zu entsperren.

Die **Card Access Number (CAN)** ist eine sechsstellige, zufällige Dezimalzahl, welche auf der Vorderseite der Ausweiskarte aufgedruckt ist.

Auch die **maschinenlesbare Zone** kann als „Geheimnis“ für PACE verwendet werden. Das Einlesen erfolgt dabei üblicherweise durch ein optisches Lesegerät. Die MRZ wurde als Alternative zur CAN beibehalten, um bereits vorhandene Lesegeräte für den elektronischen Reisepass auch mit dem elektronischen Personalausweis nut-

[7] Bundesamt für Kommunikation BAKOM: *Abklärungen über die Datenauslesung auf Distanz beim biometrischen Pass*. 28. November 2008. Verfügbar unter: `http://www.schweizerpass.admin.ch/content/dam/data/passkampagne/e-paesse/messbericht_bakom.pdf`. Letzter Zugriff: 02.06.2010

zen zu können.

Transport-PIN und PUK werden dem Ausweisinhaber nach der Ausweisbeantragung im *PIN/PUK Brief* per Post zugestellt.

Durch die Verwendung eines geteilten Geheimnisses findet je nach verwendetem Geheimnistyp auch eine implizite Authentisierung des Personalausweisinhabers statt, da nur dieser das notwendige Geheimnis kennt. Bei der Verwendung der CAN oder MRZ ist die Bindung an den Ausweisinhaber dagegen schwächer, da diese Geheimnisse potenziell jeder Person, die den Ausweis einmal gesehen hat, bekannt sind. Auf Grund dieser unterschiedlichen Qualitäten der verschiedenen Geheimnisse werden je nach verwendetem Geheimnis nur bestimmte Funktionen des Personalausweises freigeschaltet. Der Großteil der Funktionen ist jedoch erst nach der Terminal Authentication verfügbar. Lediglich das *PIN Management*, also etwa das Entsperren einer blockierten PIN mittels PUK oder das Setzen einer neuen PIN, kann bereits nach PACE durchgeführt werden.

CAN und MRZ sind nicht-blockierende Geheimnisse. Das bedeutet, dass auch bei mehrmaliger Fehleingabe keine Sperrung des Geheimnisses stattfindet. Das Blockieren eines Geheimnisses soll ein automatisiertes Durchprobieren aller Möglichkeiten verhindern. Da sowohl CAN als auch MRZ auf dem Kartenkörper aufgedruckt sind, verringert das Nicht-Blockieren das Sicherheitsniveau kaum (ein Angreifer, der in den Besitz der Karte gelangt, kennt automatisch CAN und MRZ und muss diese nicht erst durch Ausprobieren in Erfahrung bringen).

Im Gegensatz dazu darf die eID-PIN nur dem Ausweisinhaber bekannt sein. Sie ist daher durch einen Sperrmechanismus geschützt. Der eID-PIN ist ein Fehlbedienungszähler (FBZ) zugeordnet. Dieser ist im Normalzustand auf den Wert Drei gesetzt. Bei einer falschen PIN Eingabe (d.h. bei fehlgeschlagenem PACE mit eID-PIN) wird der Zähler dekrementiert. Sobald er den Wert Eins hat wird die eID-PIN in den Zustand *suspended* versetzt. In diesem Zustand ist sie noch nicht gesperrt, kann jedoch auch nicht mehr für PACE verwendet werden. Sie muss zunächst durch Eingabe der CAN wieder freigeschaltet werden. Erst danach ist eine erneute Verwendung der eID-PIN für PACE möglich. Sollte auch dieser Durchlauf fehlschlagen, so wird der FBZ auf 0 gesetzt und die eID-PIN gesperrt. Abbildung 3.2 veranschaulicht den beschriebenen Ablauf.

Der Grund für die Verwendung eines *suspended* Zustands ist das Verhindern von Denial of Service Angriffen. Ohne den *suspended* Zustand könnte ein Angreifer durch drei fehlerhafte PACE Durchläufe die eID-Funktion sperren. Durch das suspendieren der eID-PIN werden die Auswirkungen dieses Angriffes eingeschränkt. Der *suspended* Zustand ist leicht aufzuheben, wenn man im Besitz der Karte ist. Im Gegensatz zur PUK, welche der Nutzer wahrscheinlich nicht auswendig kennt, ist die CAN immer verfügbar. Außerdem hat der *suspended* Zustand keinerlei langfristige, negative Auswirkungen, wie das Erhöhen des Rücksetzzählers der eID-PIN durch das Entsperren mittels PUK.

Die Sperrung der eID-PIN kann durch Eingabe der PUK aufgehoben werden. Nach der Entsperrung kann allerdings keine neue PIN gesetzt werden, sondern es wird

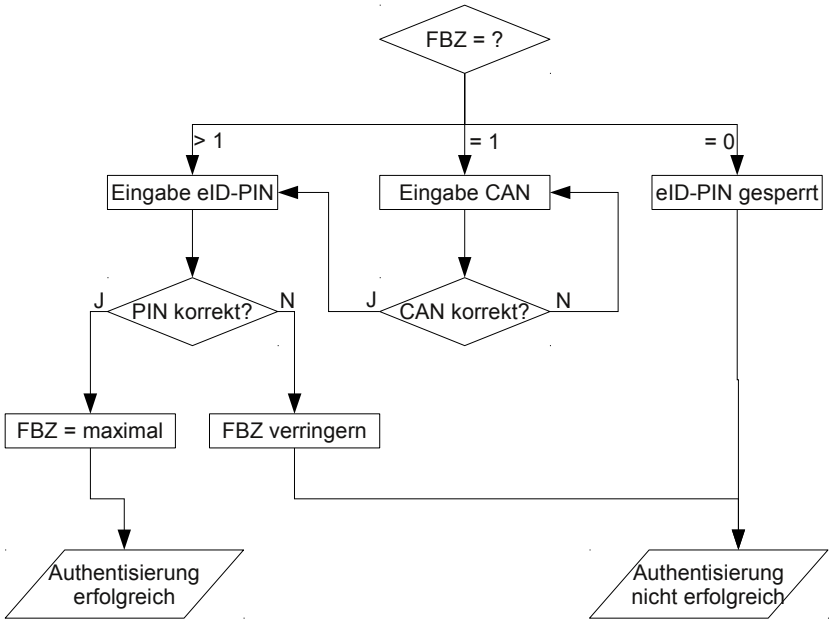

Abbildung 3.2.: Abblauf der PIN Verifikation. Quelle: BSI TR-03127 [BSI09b, S.17]

lediglich der Fehlbedienungszähler zurückgesetzt. Dadurch hat der Benutzer nach der Entsperrung mit der PUK drei weitere Versuche seine PIN korrekt einzugeben.

Die PUK ist rein technisch ein nicht-blockierendes Geheimnis. Allerdings ist die eID-PIN mit einem Rücksetzzähler versehen. Somit ist maximal ein zehnmaliges Entsperren der eID-PIN möglich. Es besteht jedoch jederzeit die Möglichkeit, eine neue PIN in den Bürgerämtern setzen zu lassen, auch wenn die korrekte PIN vergessen wurde. Dafür stehen in den Ämtern spezielle Terminals bereit, die über ein Terminalzertifikat mit der Berechtigung *PIN Management* verfügen.

3.2.2. EAC PKI

Für die passive Authentisierung und die Terminal-Authentisierung kommen Zertifikate zum Einsatz. Bei der passiven Authentisierung werden `X.509` Zertifikate verwendet, wie sie auch im Internet, beispielsweise beim SSL Protokoll, eingesetzt werden. Die Public Key Infrastructure (PKI), welche zur Ausstellung dieser Zertifikate verwendet wird, ist in [ICA08] beschrieben und besteht aus zwei Hierarchieebenen. Die Country Signing Certificate Authority (CSCA) ist die Wurzelinstanz der PKI. In Deutschland wird sie vom BSI betrieben. Die CSCA stellt Zertifikate an die Document Signer (DS), also an die Hersteller von Ausweisdokumenten, aus. In Deutschland ist dies die Bundesdruckerei. Die DS verwenden die ihren Zertifikaten zu Grunde liegenden Schlüssel, um bestimmte Daten auf dem Ausweis zu signieren (siehe Unterabschnitt 3.2.3), so dass die Authentizität des Dokuments elektronisch überprüft werden kann.

Im Gegensatz zu den bei der passiven Authentisierung verwendeten `X.509` Zertifikaten kommen bei der Terminalauthentisierung sogenannte *Card Verifiable Certificates (CV Zertifikate)* zum Einsatz. Der genaue Aufbau und Inhalt dieser Zertifikate wird in Unterabschnitt 3.2.4 beschrieben. Die technische Richtlinie 03128 [BSI10b] legt fest, dass für jede der drei elektronischen Applikationen des Personalausweises eine eigene PKI einzurichten ist. Diese Infrastrukturen sind jeweils dreistufig. Die grundsätzliche Struktur ist in Abbildung 3.3 abgebildet.

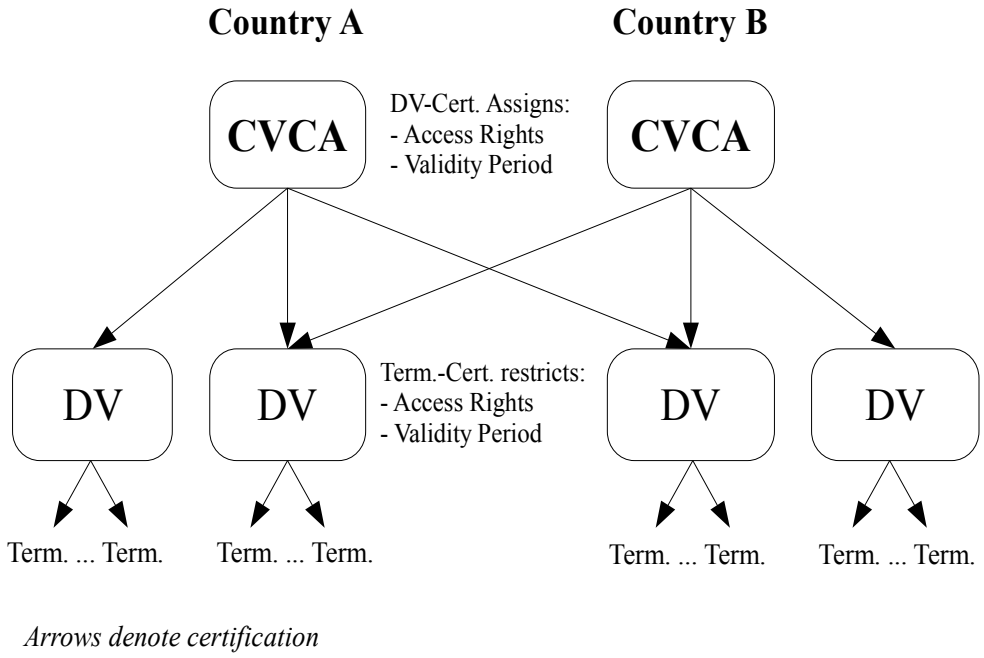

Abbildung 3.3.: Struktur der EAC PKI. Quelle: [BSI10a, S. 14]

Die in Abbildung 3.3 enthaltenen Bezeichnungen haben dabei die folgende Bedeutung:

- Die Country Validating Certificate Authority (CVCA) stellt die Wurzelinstanz der PKI dar. Sie wird in Deutschland vom BSI betrieben. Zwar bildet das BSI den alleinigen nationalen Sicherheitsanker, dennoch existiert für jede der drei elektronischen Anwendungen des Personalausweises eine unabhängige PKI Hierarchie (siehe [BSI10b, S. 18]).

- Die Document Verifier (DV) übernehmen die Rolle der *Intermediate CAs*. Anforderung und Struktur der DVs unterscheiden sich für die drei unterschiedlichen PKI Hierarchien. In der eID-PKI übernimmt die Vergabestelle für Berechtigungszertifikate (VfB), welche Teil des Bundesverwaltungsamtes ist, die Aufgaben einer *Registration Authority*. Das heißt, sie nimmt die Zertifikatsanträge der Dienstanbieter entgegen, prüft sie und erteilt die endgültigen Zugriffsrechte.

Für das eigentliche Ausstellen der Zertifikate sind durch die Bundesnetzagentur zertifizierte TrustCenter aus der Privatwirtschaft zuständig.

- Die Terminalzertifikate sind die Zertifikate, die an Dienstanbieter ausgestellt werden und diesen den elektronischen Zugriff auf die Personalausweise der Benutzer ermöglichen.

3.2.3. Passive Authentisierung

Bei der passiven Authentisierung handelt es sich um einen Mechanismus zum Integritätsschutz der Ausweisdaten. Hierzu werden ausgewählte Datengruppen auf dem Ausweis mit einer Signatur versehen. Zwei Datengruppen dienen im Personalausweis der passiven Authentisierung:

1. `EF.CardSecurity` beinhaltet die signierten Daten aus `EF.CardAccess`. In `EF.CardAccess` sind dabei die kryptografischen Parameter, welche von dem Ausweischip unterstützt werden, abgespeichert. Diese Datei wird zu Beginn der Kommunikation mit dem Personalausweis ausgelesen. Nach der Durchführung des PACE Protokolls kann das Terminal die Datei `EF.CardSecurity` auslesen und anhand der Signatur sicherstellen, dass PACE mit den korrekten Parametern durchgeführt wurde. Von zentraler Bedeutung ist die Signatur des öffentlichen Schlüssels für die Chip Authentication, welcher ebenfalls in `EF.CardAccess` enthalten ist. Durch diese Signatur wird sichergestellt, dass die Chip Authentication nur mit dem im Ausweischip gespeicherten privaten Schlüssel durchgeführt werden kann. Da dieser nicht aus dem Ausweis ausgelesen werden kann, wird durch die Kombination aus passiver Authentiserung und CA die Emulation eines Ausweises mit einem selbst erstellten CA-Schlüsselpaar verhindert.

2. `EF.SOD` ist Teil der Biometrieanwendung und enthält die signierten Hashes aller Datengruppen der Biometrieanwendung.

Passive Authentisierung schützt die auf dem Chip gespeicherten Daten vor nachträglichen Veränderungen, verhindert aber nicht die Anfertigung einer Eins-zu-Eins Kopie der Daten.

3.2.4. Terminalauthentisierung

Wie der Name bereits besagt, ist das Ziel des *Terminal Authentication (TA)* Protokolls die Authentisierung und Autorisierung des Dienstes. Ein Dienst muss nachweisen, dass er berechtigt ist, auf bestimmte Daten oder Funktionen des Personalausweises zuzugreifen. Dies geschieht über *Terminalzertifikate*. Dabei handelt es sich um die bereits erwähnten CV-Zertifikate, welche sehr viel kompakter als die im Internet gebräuchlichen `X.509` Zertifikate sind und dadurch besser durch Chipkarten mit ihren stark begrenzten Ressourcen verarbeitet und verifiziert werden können.

Zertifikat	
Body	
	Certificate Profile Identifier Certification Authority Reference Public Key Certificate Holder Reference Certificate Holder Authorization Template Certificate Effective Date Certificate Expiration Date Certificate Extensions
Signature	

Tabelle 3.3.: Aufbau der CV Zertifikate

Der Aufbau der CV-Zertifikate ist in Tabelle 3.3 dargestellt.
Die im Zertifikat enthaltenen Daten haben dabei folgende Bedeutung:

Certificate Profile Identifer gibt das verwendete Zertifikatsprofil an. Bisher ist lediglich das hier vorgestellte Profil spezifiziert, welches durch die Nummer 0 referenziert wird.

Die Certificate Holder Reference (CHR) identifiziert den Inhaber eines Zertifikates.

Certificate Authority Reference (CAR) ist der Verweis auf die Zertifizierungsstelle, welche ein Zertifikat ausgestellt hat. So entspricht beispielsweise die CAR eines Terminalzertifikats der CHR der ausstellenden DVCA.

Public Key enthält den öffentlichen Schlüssel, welcher dem Zertifikat zugeordnet ist. Es sind Schlüsselformate für RSA, Diffie-Hellmann und ECC Schlüssel spezifiziert. Nur CVCA Zertifikate enthalten die Domänenparameter eines Schlüssels. DV- und Terminalzertifikate müssen die selben Domänenparameter wie die CVCA verwenden.

Das Certificate Holder Authorisation Template (CHAT) codiert die Zugriffsrechte eines Zertifikatsinhabers auf die Daten und Funktionen eines Personalausweises. Der CHAT besteht aus einem OID und einem Bitstring. Der OID gibt dabei die Art des Terminals, also Authentication Terminal, Signaturterminal oder Inspektionssystem, an. Der Bitstring codiert zum einen die Rolle (CVCA, DV oder Terminal), zum anderen die eigentlichen Zugriffsrechte. Für die eID-Funktion ist ein Bitstring der Länge fünf Byte, für die eSign- und ePass-Funktion der Länge ein Byte spezifiziert.

Certificate Effective Date gibt den Zeitpunkt an, ab welchem ein Zertifikat gültig ist.

Certificate Expiration Date gibt den Zeitpunkt an, bis zu welchem das Zertifikat als gültig anzusehen ist.

Certificate Extensions enthält optionale Erweiterungen des Zertifikats. Zwei Erweiterungen sind bisher spezifiziert:

1. Die Erweiterung *Certificate Description* enthält einen Hashwert über die Certificate Description (siehe Abschnitt 3.3), welche im Rahmen der eCard API an das lokale Terminal übertragen wird. Anhand des Hashwertes kann das lokale Terminal überprüfen, dass es die dem Zertifikat zugehörige Certificate Description erhalten hat. Die Karte selbst ignoriert diesen Wert, da die Certficate Description nie an sie übertragen wird.
2. Die Erweiterung *Terminal Sector* kann bis zu zwei Hashwerte über öffentliche Schlüssel enthalten, welche im Rahmen der Restricted Identification verwendet werden.

Die eigentliche Terminal Authentication ist ein Challenge-Response Protokoll, in welchem der Dienstanbieter beweist, dass er über den zum Terminalzertifikat passenden privaten Schlüssel verfügt. Dadurch ist die Authentizität des Diensteanbieters nachgewiesen.

3.2.5. Chip Authentication

Die Chip Authentication dient dem Nachweis, dass es sich bei dem Ausweis um einen authentischen Personalausweis handelt. Dies ist notwendig, damit sich ein Dienstanbieter sicher sein kann, dass die Daten, die später im Rahmen der eID-Anwendung gelesen werden, korrekt sind. Das Chip Authentication Protokoll basiert auf einem geheimen, asymmetrischen Schlüssel, welcher unauslesbar im Ausweischip gespeichert ist. Genau wie bei der TA handelt es sich bei der CA um ein Challenge-Response Protkoll. Der Ausweischip weist hierdurch nach, dass er über den passenden privaten Schlüssel zum im `EF.CardAccess` enthaltenen öffentlichen Schlüssel verfügt. Wie bereits in Unterabschnitt 3.2.3 beschrieben, ist die Integrität des öffentlichen CA Schlüssels durch die passive Authentisierung sichergestellt.

Durch das Protokoll werden neue Schlüssel etabliert, welche anschließend für einen neuen Secure Messaging Kanal verwendet werden. Ist dieser Kanal etabliert, können nur noch der Dienstanbieter und die Karte gültige Nachrichten über diesen Kanal verschicken. Auch der Karteninhaber bzw. Software, die auf seinem Computer läuft, verfügt nicht über die notwendigen Schlüssel und kann somit keine eigenen Daten verschicken oder manipulieren. Dadurch ist der Dienstanbieter vor Manipulationen durch den Nutzer geschützt.

3.3. Die eID-Authentisierung aus Sicht des Nutzers

Aus der vorangegangenen Beschreibung des neuen Personalausweises wird ersichtlich, dass es sich hierbei um ein komplexes informationstechnisches System handelt. Das

Verständnis des gesamten Systems kann von einem durchschnittlichen Nutzer nicht erwartet werden und ist auch nicht notwendig, um die Funktionen des Personalausweises zu nutzen. Im Folgenden wird daher die eID-Authentisierung aus Sicht des Benutzers analysiert. Insbesondere werden alle notwendigen Interaktionen des Nutzers mit dem System herausgearbeitet, um das User-Interface des nPA vollständig zu erfassen.

Die eID-Authentisierung mit dem elektronischen Personalausweis wird von dem Dienst, den der Ausweisinhaber nutzen möchte, initiiert. Der gängigste Anwendungsfall wird dabei sein, dass der Nutzer auf der Webseite des Anbieters surft und dort einen Anmeldevorgang auslöst. Dies führt zu einem Aufruf der lokalen eID-Software, beispielsweise des Bürgerclients[8], auf dem PC des Nutzers. Diese Software muss anschließend drei Interaktionsphasen mit dem Nutzer durchführen:

1. Anzeige von Informationen über den Dienstanbieter
2. Anzeige und möglicherweise Abwahl der Zugriffsrechte
3. Eingabe der eID-PIN

In der ersten Phase werden dem Benutzer Informationen über seinen Kommunikationspartner, den Dienstanbieter, angezeigt. Dieser Schritt soll Transparenz für den Nutzer schaffen und sicherstellen, dass er sich tatsächlich mit dem gewünschten Anbieter verbindet. Welche Daten dem Nutzer zumindest angezeigt werden müssen ist in PAuswG, § 18, Absatz 4 festgelegt:

- Name, Anschrift und E-Mail-Adresse des Dienstanbieters
- erwünschte Zugriffsrechte
- Zweck der Datenübermittlung
- Hinweis auf die für den Dienstanbieter zuständige Datenschutzbehörde
- letzter Tag der Gültigkeitsdauer des Berechtigungszertifikats

Das Terminalzertifikat des Anbieters enthält von diesen geforderten Daten lediglich die Zugriffsrechte und die Gültigkeitsdauer. Die übrigen Daten müssen der *Certificate Description* entnommen werden. Diese Datenstruktur wird im Rahmen der eCard-API vom Dienstanbieter an das lokale Terminal übertragen (vergleiche [BSI09c, S. 45f]). Sie enthält die in Tabelle 3.4 aufgeführten Daten (vergleiche [BSI10a, S. 82]).

[8] Der Bürgerclient ist die offizielle Anwendungssoftware für den neuen Personalausweis. Sie wird von der Firma OpenLimit entwickelt und soll am 01. November 2010 veröffentlicht werden. Am 06. Juli 2010 wurde bekannt gegeben, dass die Software in AusweisApp umbenannt wurde (siehe OpenLimit: *Neues BürgerClient-Update an BMI übergeben.* 06.07.2010. https://www.openlimit.com/de/ueber-openlimit/presse/news/2010/buergerclient-update.html Letzter Zugriff: 30. Juli. 2010). In dieser Arbeit wird dennoch die Bezeichnung Bürgerclient beibehalten

Name	Bedeutung
`descriptionType`	Art der Certificate Description
`issuerName`	Name des Zertifikatausstellers
`issuerURL`	optionale URL
`subjectName`	Name des Dienstanbieters
`subjectURL`	optionale URL
`termsOfUsage`	Weiterführende Informationen zu Dienst und Anbieter
`redirectUrl`	optionale URL

Tabelle 3.4.: Certificate Description

Der Object Identifier `descriptionType` kann dabei einen von drei Werten annehmen: `id-plainFormat`, `id-htmlFormat` oder `id-pdfFormat`. Durch diesen Identifikator wird das Format der `termsOfUsage` Daten angegeben. Wie sich aus den möglichen Werten leicht ablesen lässt, können sie entweder als Klartext (UTF-8 codiert), im HTML oder im PDF Format angegeben werden. Leider ist nicht spezifiziert, welche Angaben im `termsOfUsage` Feld in welcher Reihenfolge enthalten sein müssen. Aus den oben aufgeführten Anforderungen des Personalausweisgesetzes ergibt sich aber, dass zumindest die Anschrift und E-Mail Adresse des Dienstanbieters, sowie Angaben zum zuständigen Datenschutzbeauftragten vorhanden sein müssen.

Im nächsten Schritt werden die im CHAT des Terminalzertifikats codierten Zugriffsrechte angezeigt. Der Nutzer hat nun die Möglichkeit, diese Berechtigungen durch gezielte Abwahl einzelner Rechte weiter einzuschränken. Dies kann jedoch dazu führen, dass ein Dienstanbieter nicht genügend Informationen erheben kann, um seine Dienstleitung zu erbringen. Für den Benutzer ist nicht ersichtlich, welche Rechte zwingend erforderlich sind und welche nicht.

Nach der Freigabe der Zugriffsrechte muss die eID-PIN vom Ausweisinhaber eingegeben werden. Spätestens zu diesem Zeitpunkt muss der Ausweis sich im Feld des Lesegerätes befinden. Aufgrund der bereits beschriebenen geringen Funkreichweite ISO-14443 kompatibler Chipkarten und Lesegeräte muss der Ausweis dabei direkt auf dem Lesegerät oder zumindest in unmittelbarer Entfernung platziert werden. Eventuell kann der Ausweis im Portemonnaie belassen werden, wobei es aber zu Störungen durch Münzen oder andere kontaktlose Chipkarten kommen kann.

Die Eingabe der PIN kann entweder über die Software auf dem PC oder aber direkt am Lesegerät eingegeben werden. Letztere Möglichkeit besteht aber nur, wenn der Nutzer über ein geeignetes Lesegerät verfügt. Die technische Richtlinie 03119 [BSI09a] spezifiziert drei Klassen von Lesegeräten: Basisleser, Standardleser und Komfortleser. Die wichtigsten Unterschiede zwischen den drei Klassen sind in Tabelle 3.5 aufgeführt.

Verfügt das Lesegerät des Nutzers über ein Pinpad, so kann er die PIN direkt am Lesegerät eingeben. Dies bietet ein höheres Maß an Sicherheit, da die PIN nicht durch

	Basisleser	Standardleser	Komfortleser
Kontaktlose Schnittstelle	X	X	X
Pinpad	O	X	X
Display	O	O	X
PACE	O	X	X
QES	O	O	X
Firmwareupdate	O	X	X

Tabelle 3.5.: Die drei Klassen von Lesegeräten nach BSI TR-03119. Mit einem X gekennzeichnete Attribute sind für die beschriebene Klasse verpflichtend vorgeschrieben. Ein O kennzeichnet optionale Attribute.

eventuell auf dem Computer vorhandene Schadsoftware abgefangen werden kann[9]. PACE wird in diesem Fall zwischen dem Lesegerät und dem Ausweis durchgeführt.

Verfügt das Lesegerät zusätzlich noch über ein Display, so kann dies dazu verwendet werden, dem Nutzer einen Teil der oben genannten Informationen dort anzuzeigen. Auch hier ist der Vorteil gegenüber einer Anzeige am Computer wieder die prinzipiell höhere Manipulationssicherheit des Lesegeräts. Allerdings sind die Displays, welche heutzutage in Chipkartenlesegeräten verbaut werden, sehr beschränkt. [BSI09a] fordert, dass das Display eines Komfortlesegerätes mindestens zwei Zeilen mit jeweils 16 alphanumerischen Zeichen darstellen können muss. Das genügt bei Weitem nicht, um die geforderten Informationen komfortabel anzuzeigen. Insbesondere die Verarbeitung und Anzeige von CertificateDescriptions im HTML oder PDF Format ist auf einem Lesegerät problematisch. Auch die Abwahl von Zugriffsrechten direkt am Lesegerät ist so nur schwer vorstellbar.

Nach der PIN-Eingabe läuft, für den Nutzer unsichtbar, die General Authentication Procedure zwischen Ausweis, lokalem Terminal und Dienstanbieter ab. Wenn keine Fehler auftreten (etwa durch eine fehlerhaft eingegebene PIN oder Fehler in der Datenübertragung), wird der Nutzer anschließend wieder zurück auf die Seite des Dienstanbieters umgeleitet und die Authentisierung war erfolgreich.

3.3.1. Mobiler Einsatz des elektronischen Personalausweis

Versucht man ein NFC-fähiges Telefon in die oben beschriebenen Kategorien für Lesegeräte einzuordnen, so stellt man zunächst fest, dass es grundsätzlich über alle geforderten Funktionen verfügt. Das Display eines Handys ist sehr viel komfortabler als die Displays, mit denen Chipkartenlesegeräte heutzutage ausgestattet sind. Auch

[9] Es sei jedoch darauf hingewiesen, dass es nicht unmöglich ist, ein Lesegerät zu manipulieren, selbst wenn dieses zertifiziert ist. So gelang es beispielsweise im April 2010 einem Hacker mit dem Pseudonym „Colibri“ eine modifizierte Firmware in zertifizierten Klasse 3 Leser Kaan Tribank von Kobil einzuspielen (siehe: Colibri: *Smartcard-Reader von Kobil geknackt.* 17.04.2010. http://colibri.net63.net/Smartcard-Reader%20von%20Kobil%20geknackt.pdf. Letzter Zugriff: 02.08.2010)

mehr oder weniger komfortable Eingabemöglichkeiten sind bei jedem Handy vorhanden. Notwendige Funktionen, wie etwa PACE, können in Software abgebildet werden und dank NFC steht auch eine geeignete Funkschnittstelle zur Verfügung.

Allerdings schreibt TR-03119 für Standardleser ein Sicherheitsgutachten, für Komfortleser sogar eine Sicherheitsevaluierung und -verifikation nach Common Criteria Level EAL3+ vor (siehe [BSI09a, S.27 und S. 34]). Diese ist auf Grund der vielfältigen Funktionalität moderner Telefone und der extrem kurzen Entwicklungszyklen von Handybetriebssystemen praktisch für kein Handy umsetzbar. Daher kann ein Handy lediglich als Basislesegerät eingestuft werden.

Insbesondere ist eine Verwendung der eSign Funktion mit einem NFC-Handy derzeit ausgeschlossen. Hierzu wäre neben der Zertifizierung der auf dem Handy laufenden Software noch eine sichere Speichereinheit für das Signaturterminalzertifikat und das zugehörige private Schlüsselmaterial notwendig. Eine sichere Speichereinheit ist im Handy zwar in Form der SIM-Karte vorhanden, eine Nutzung für andere Applikationen als die GSM Anwendung ist zur Zeit aber nicht vorgesehen. Zukünftig könnte sich das mit dem sogenannten Single-Wire Protokoll ändern. Dieses Protkoll befand sich zum Zeitpunkt des Verfassens dieser Arbeit noch im Prozess der Standardisierung durch die European Telecommunications Standards Institute (ETSI). Es soll eine Schnittstelle zwischen dem NFC-Chip eines Handys und der SIM-Karte definieren und vor allem für Handy-basierte Micropayment Systeme zum Einsatz kommen.

Für die Nutzung mittels NFC-Handy bleibt zunächst also nur die eID-Funktion des nPA. Der wesentliche Unterschied bei der Nutzung der eID-Authentisierung auf einem Mobiltelefon aus Nutzersicht ist, dass der Nutzer nur noch mit *einem* Gerät umgehen muss. Bildschirm, Eingabegeräte, Computer und RFID-Leser sind nicht länger getrennt, sondern in einem Gerät integriert. Dadurch ändert sich vor allem die physikalische Handhabung des Systems. Der Ausweis wird nun nicht mehr in ein Lesegerät auf dem Schreibtisch eingelegt, sondern muss zusammen mit dem Handy während der Bedienung in der Hand gehalten werden.

Weiterhin verändert sich bei der Verwendung eines Mobiltelefons der Nutzungskontext des Systems. Mobile Einsatzbedingungen sind weitaus vielfältiger als stationäre. In vielen mobilen Situationen ist der Nutzer ungünstigen Umgebungseinflüssen ausgesetzt. Beispielhaft sei hier etwa die Nutzung in Bus oder Bahn angeführt. Damit mobile Systeme auch in solchen Situationen noch gut zu bedienen sind, ist eine sorgfältige Gestaltung des gesamten Systems unumgänglich.

4. Handhabung von Gerät und Ausweis

Durch die Verwendung des Personalausweises zur Authentisierung des Nutzers wird es notwendig, sowohl den Ausweis als auch das Telefon gleichzeitig zu handhaben. Der Ausweis muss während der gesamten Dauer der General Authentication Procedure innerhalb des durch den NFC Chip erzeugten Feldes gehalten werden, also in einer Entfernung von maximal 10 cm vom Handy (in der Praxis zeigte sich beim weiter unten beschriebenen Nokia 6131 NFC sogar eine noch geringere Sendeweite). Diese Anforderung betrifft die physikalische Komponente der Benutzungsschnittstelle in dem Modell von Moran (siehe Unterabschnitt 2.1.1), also die direkte Handhabung des Geräts.

Um festzustellen, ob die gleichzeitige Handhabung von ePA und Mobiltelefon Auswirkungen auf die Usability hat, wurde ein Usability Test durchgeführt. Versuchspersonen sollten auf unterschiedlichen Mobiltelefonen eine vorgegebene PIN eingeben, während sie einen Testausweis am Gehäuse hielten.

4.1. Verwendete Geräte

Um den Einfluss verschiedener Bauformen von Mobiltelefonen bewerten zu können, wurden drei unterschiedliche Geräte für den Test verwendet: das OpenMoko Neo FreeRunner (im Folgenden OpenMoko genannt), das Nokia E 71[1] und das Nokia 6131 NFC. Diese drei Telefone wurden jeweils stellvertretend für eine bestimmte Klasse von Mobiltelefonen ausgewählt: das OpenMoko als Touchscreen Handy, das Nokia E 71 als Business Smartphone und das Nokia 6131 als gewöhnliches Consumer Handy. Auf diese Weise sollten möglichst unterschiedliche Bauarten und damit verbundene Bedienmethoden betrachtet werden.

Das **OpenMoko** verfügt über einen resistiven Touchscreen und wird mittels eines Stiftes bedient. Moderne Techniken, wie Multitouch oder Gestensteuerung, werden nicht unterstützt. Das Display hat eine Auflösung von 640*480 Pixeln.

Das OpenMoko wurde als Open Source Gerät konzipiert, wobei sich die Offenheit sowohl auf die Hardware, deren Spezifikation komplett freigelegt wurde, als auch auf die Software bezieht. Aufgrund dieser Offenheit existiert eine Vielzahl an (zumeist

[1] Technische Spezifikationen siehe: http://europe.nokia.com/find-products/devices/nokia-E\protect\relax\kern.16667em71/specifications

Abbildung 4.1.: Von links nach rechts: Nokia E 71, Nokia 6131, OpenMoko Neo FreeRunner

Linux-basierten) Betriebssystemen, welche sich hinsichtlich Funktionalität, Bedienkonzepten und Stabilität stark unterscheiden. Für den Test wurde die Distribution SHR (Stable Hybrid Release)[2] verwendet. Diese zeichnete sich zum Zeitpunkt des Verfassens dieser Arbeit durch eine besonders aktive Entwicklung, umfassende Funktionalität und eine vergleichsweise attraktive grafische Benutzeroberfläche aus. Das Betriebssystem stand in zwei Varianten, `testing` und `unstable`, zur Verfügung. Es wurde die Variante `testing` verwendet, da diese sich als sehr viel stabiler erwies, auch wenn die beinhalteten Programme weniger aktuell waren als in der `unstable` Variante.

Das **Nokia E 71** ist ein Smartphone mit einer kompletten `QWERTZ` Tastatur. Darüber hinaus sind noch eine 4-Wege-Wippe und insgesamt acht Softkeys enthalten. Der Bildschirm hat eine Diagonale von 2,36 Zoll und verfügt über eine Auflösung von 320*240 Pixeln. Als Betriebssystem wird Symbian S60 in der Version 3 Feature Pack 1 verwendet.

Beim **Nokia 6131 NFC** handelt es sich um ein sogenanntes *clamshell* Handy, umgangssprachlich besser bekannt als Klapphandy. Es verfügt über ein Tastenfeld mit 12 Tasten. Zusätzlich besitzt es eine 4-Wege-Wippe zur Navigation in Menüs und vier Softkeys. Das Display hat eine Diagonale von 5,59 cm und eine Auflösung

[2] *SHR Project*. Verfügbar unter: `http://www.shr-project.org`. Letzter Zugriff: 21.07.2010

von 240*320 Pixeln. Als Betriebssystem kommt Symbian S40 in der Version 3 zum Einsatz.

Für die PIN-Eingabe wurde für jedes der Handys eine grafische Nutzeroberfläche entwickelt. Auf den beiden Nokia Handys wurden dafür J2ME und das GUI-Framework Lightweight User Interface Toolkit (LWUIT)[3] verwendet. Dank der weitestgehenden Plattformunabhängigkeit von J2ME konnte auf beiden Handys dasselbe Programm verwendet werden. Aufgrund der unterschiedlichen Seitenverhältnisse waren jedoch kleinere Anpassungen des Oberflächenlayouts notwendig.

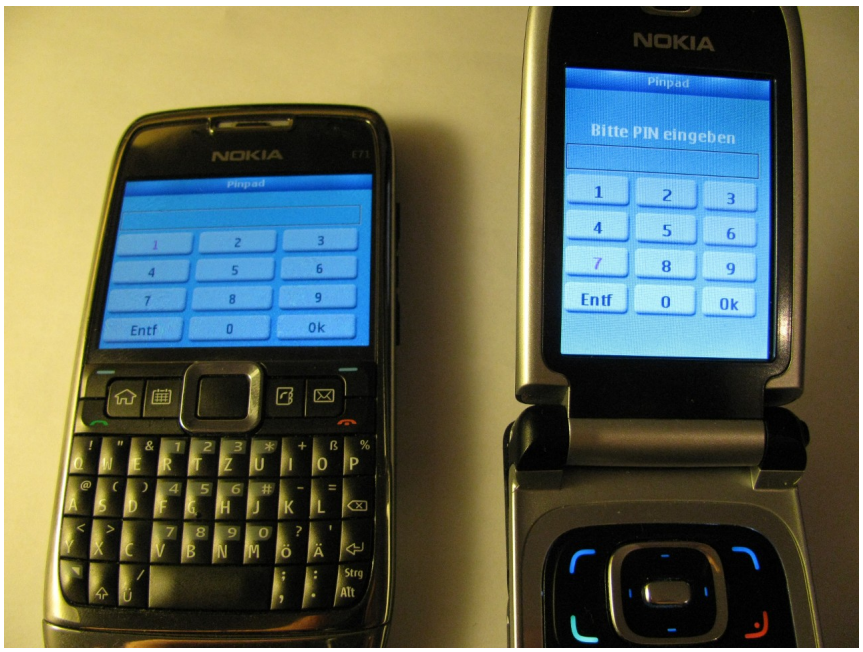

Abbildung 4.2.: Die grafische Nutzeroberfläche der beiden Nokia Telefone

Auf dem Bildschirm wurde die grafische Repräsentation eines Pinpads in Form von 12 Tasten und einem Textfeld zur Ausgabe angezeigt. Auf dem Nokia 6131 war noch Platz für einen Instruktionstext vorhanden. Diese Repräsentation wurde aus zwei Gründen ausgewählt: Zum einen sollte dem Nutzer ein ihm bekanntes Bild gezeigt werden. Die Annahme hierbei war, dass die meisten Anwender ein Pinpad beispielsweise von Bankautomaten her kennen. Zum anderen sollte dem Nutzer Rückmeldung zu seinen Aktionen gegeben werden. Dies geschah, indem bei der Eingabe einer Ziffer die entsprechende Taste auf dem Bildschirm durch Veränderung der Farbe als gedrückt dargestellt wurde. Außerdem wurde für jede eingegebene Ziffer ein Stern im Textfeld angezeigt.

Um die Eingabe der PIN so intuitiv wie möglich zu gestalten, wurden zwei alternative Eingabemethoden implementiert. Zum einen konnten die Ziffern der PIN direkt

[3] *LWUIT*. Verfügbar unter: https://lwuit.dev.java.net/. Letzter Zugriff: 27.04.2010

über die Nummerntasten der Mobiltelefone eingegeben werden. Zum anderen konnten die Tasten über die 4 Wege Wippe einzeln angewählt und betätigt werden. Die aktuell ausgewählte Taste wurde durch einen Wechsel der Farbe der Tastenbeschriftung angezeigt. Um die Eingabe der PIN zu bestätigen, war ein weiterer Tastendruck notwendig (Taste „Ok“ in der Bildschirmrepräsentation). Hierfür konnte entweder die Sterntaste oder der rechte Softkey verwendet werden. Das Löschen einer Ziffer konnte über den linken Softkey oder die Rautetaste bewerkstelligt werden. Beim E 71 bestand zusätzlich die Möglichkeit, Zeichen über die Backspace-Taste zu löschen und die Eingabe über die Enter-Taste zu bestätigen. Tasten, welche keine Funktionen für die Nutzerinteraktion erfüllten, wurden ignoriert. Es war jedoch nicht möglich solche Tasten abzufangen, welche Systemfunktionen erfüllten, wie zum Beispiel die Auflegen-Taste zum Beenden eines Programmes oder die Kurzwahltasten des E 71.

Da zum Zeitpunkt der Entwicklung keine J2ME Umsetzung für SHR zur Verfügung stand, musste ein eigenes Programm für das OpenMoko entwickelt werden. Die Umsetzung erfolgte in Python unter der Verwendung von GTK als Grafikbibliothek. Wiederum wurde auf dem Bildschirm die grafische Repräsentation eines Pinpads angezeigt. Die Eingabe erfolgte über direkte Manipulation, also durch die Auswahl der Taste über den Touchscreen. Auch in diesem Programm wurden die eingegebenen Ziffern mittels Sternchen in einem Textfeld repräsentiert.

Abbildung 4.3.: Die grafische Nutzeroberfläche des OpenMoko

Der Abschluss der Eingabe führte zur Anzeige eines Popup Dialoges, welcher die Versuchspersonen darüber informierte, ob die eingegebene PIN korrekt oder falsch

war. Bei korrekter Eingabe führte die Bestätigung des Dialogs zum Beenden des Programms. Bei Eingabe einer falschen PIN musste diese erneut eingegeben werden. Dazu musste zunächst der Dialog über einen Tastendruck bestätigt werden. Alternativ verschwand er nach drei Sekunden automatisch. Im PIN-Dialog war die vorige Eingabe gelöscht, die PIN musste komplett neu eingegeben werden.

Im Hintergrund sammelte das Eingabeprogramm Versuchsdaten. Es wurde der Zeitpunkt jedes Tastendruckes abgespeichert, zusammen mit der Information, welche Taste gedrückt wurde. Die Zeitmessung erfolgte dabei in Millisekunden relativ zum ersten Tastendruck.

4.2. Versuchsablauf

Zu Beginn des Versuchs wurde den Versuchspersonen der folgende Text vorgelesen:

> Ab dem ersten November wird der neue elektronische Personalausweis eingeführt werden. Man wird ihn unter anderem dazu benutzen können, sich im Internet an Webseiten anzumelden. Dazu verfügt er über eine sechsstellige PIN, ähnlich wie eine Bankkarte. Gleichzeitig nutzen immer mehr Menschen das Internet über ihr Handy. In meiner Diplomarbeit beschäftige ich mich mit der Möglichkeit, den elektronischen Personalausweis zusammen mit einem Handy zu nutzen. Besonderes Augenmerk lege ich dabei auf die Gebrauchstauglichkeit.
>
> In dem Versuch werden Sie einen Testausweis samt PIN und nacheinander drei verschiedene Mobiltelefone von mir erhalten. Auf jedem dieser Telefone läuft ein Programm zur PIN-Eingabe. Sie sollen die PIN eingeben und dabei gleichzeitig den Ausweis in einem Abstand von maximal fünf Zentimetern zum Handy halten.
>
> Im Anschluss werde ich Ihnen noch einige Fragen zu Ihren demografischen Daten und zum Versuch an sich stellen. Diese Daten werden später anonymisiert verarbeitet, sodass kein Rückschluss auf Sie als Versuchsperson möglich ist.

Jeweils nach dem Verlesen des Textes und nach dem Bearbeiten der eigentlichen Aufgabe wurde den Versuchspersonen die Gelegenheit gegeben, Fragen zu dem Versuch zu stellen. Den Versuch selbst mussten sie ohne weitere Hilfestellungen absolvieren.

Die Reihenfolge, in welcher die verschiedenen Handys von den Versuchspersonen verwendet wurden, wurde permutiert, um eventuelle Lerneffekte einzugrenzen. Nach Abschluss des Tests füllten die Versuchspersonen den in Anhang B abgebildeten Fragebogen aus. Dieser erfasste demografische Daten und Angaben zur Nutzung von Mobiltelefonen. Außerdem sollte die Gebrauchstauglichkeit der PIN-Eingabe für alle drei Handys auf einer siebenstufigen Likert-Skala bewertet werden. Hierbei bedeutete eine Wertung von eins „sehr schlecht“ und eine Wertung von sieben „sehr gut“.

Weiterhin konnten in einem Freitextfeld zusätzliche eigene Anmerkungen der Probanden gemacht werden. Insgesamt standen also nach der Durchführung der Tests Daten zu allen drei in [ISO98] definierten Dimensionen der Usability zur Verfügung: Es wurde erfasst, ob die Versuchspersonen in der Lage waren die PIN erfolgreich einzugeben. So wurde die Effektivität der Interaktion gemessen. Über die Zeitpunkte der Tastendrücke ließen sich Rückschlüsse auf die Effizienz der PIN-Eingabe treffen. Und durch die subjektive Wertung wurde die Zufriedenheit der Versuchspersonen erfasst.

Während des Versuches erfasste der Versuchsleiter darüber hinaus weitere qualitative Daten. Dazu gehörten vor allem die Art und Weise, wie die Karte gehalten wurde und eventuelle besondere Vorkommnisse, wie etwa das Fallenlassen von Karte oder Telefon. Auch Fragen, welche die Versuchspersonen vor oder nach der Bearbeitung der Aufgabe stellen konnten, wurden festgehalten. Außerdem wurden die Versuchspersonen, so weit sie dazu ihr Einverständnis gaben, bei der Durchführung des Versuches gefilmt. Das Videomaterial wurde später untersucht, um weitere qualitative Beobachtungen zu machen.

4.3. Stichprobe

Die Rekrutierung der Versuchspersonen erfolgte informell, also ohne Ausschreibung und vorherige Auswahl der Versuchspersonen. Stattdessen wurde eine willkürliche Stichprobe genommen, indem der Autor Nachbarn und Bekannte als Versuchspersonen gewann. Dieses Vorgehen wurde dadurch begünstigt, dass der Versuch innerhalb kurzer Zeit an einem beliebigen Ort durchgeführt werden konnte. Somit war es nicht notwendig, die Versuchspersonen in ein Labor einzuladen. Stattdessen konnte der Versuch bei ihnen zu Hause durchgeführt werden.

Diese Form der Rekrutierung von Versuchspersonen eignet sich aufgrund der kaum vorhandenen formalen Rekrutierungskriterien und der daraus resultierenden unzureichenden Vergleichbarkeit der einzelnen Versuchspersonen nur eingeschränkt für eine statistische Analyse. Die in Abschnitt 4.4 genannten Daten und Diagramme sollten daher nicht leichtfertig für Prognosen für die gesamte Bevölkerung genutzt werden. Dies war jedoch auch nicht das Ziel dieses Versuches. Es handelt sich hierbei nicht um ein kontrolliertes Experiment, sondern um einen Usability Test. Letztere dienen vor allem dem Aufdecken von potenziellen Usability Problemen und dem Auffinden konkreter Gestaltungsfehler (vergleiche [SP09, S. 157]).

Das einzige Kriterium, welches bei der Rekrutierung angelegt wurde, war das Alter der Versuchspersonen. Es wurde unterschieden zwischen jüngeren (jünger als 30 Jahre) und älteren (älter als 50 Jahre) Versuchspersonen. Diese Unterscheidung wurde getroffen, um festzustellen, ob ältere Personen eventuell mehr Probleme bei der PIN-Eingabe mit Ausweis und Handy haben als jüngere.

Die endgültige Stichprobe setzte sich wie folgt zusammen: Die Gruppe der älteren Versuchspersonen umfasste 15 Personen, sieben davon weiblich und acht männlich. Der Altersdurchschnitt lag bei 58,33 Jahren ($SD = 6,14$). 14 Probanden dieser

Gruppe besaßen ein Handy, zwei von ihnen hatten schon einmal das Internet mit einem Mobiltelefon genutzt. Die Gruppe der jüngeren Probanden bestand aus 18 Personen (zehn weiblich, acht männlich). Der Altersdurchschnitt betrug 25,56 Jahre ($SD = 2,99$). Alle Probanden besaßen ein Handy und zwölf verfügten über Interneterfahrung mit Mobiltelefonen.

4.4. Analyse der Versuchsdaten

Sämtliche Versuchsdaten befinden sich auf der dieser Arbeit beiliegenden CD im Ordner `daten/handhabung` in Form von Excel Tabellen. Auch die zur Auswertung der Daten erzeugten SPSS Dateien finden sich in diesem Verzeichnis.

Die erfassten quantitativen Daten können den drei Kategorien der Gebrauchstauglichkeit nach ISO 9241 – Effektivität, Effizienz und Nutzerzufriedenheit – zugeordnet werden.

4.4.1. Effektivität

Die Effektivität der Interaktion lässt sich daran feststellen, ob es den Versuchspersonen gelang, die PIN korrekt einzugeben oder nicht.

Allen Versuchspersonen der jüngeren Gruppe gelang es, auf allen drei Telefonen die PIN korrekt einzugeben. In der Gruppe der älteren Versuchspersonen war die Erfolgsquote wesentlich geringer. Hier gelang die PIN-Eingabe zwölf Versuchspersonen mit dem OpenMoko, neun mit dem Nokia 6131 und lediglich acht mit dem Nokia E 71.

Es gab vier verschiedene Ursachen für Misserfolge bei der PIN-Eingabe:

Versehentliches Beenden des Eingabeprogramms: Bei den beiden Nokia Telefonen geschah dies beispielsweise, wenn der Softkey zum Beenden eines Gespräches gedrückt wurde. Beim OpenMoko konnte es bei Klicks in die Menüleiste zum Verlassen des Programmes kommen. In diesen Fällen wurde der Versuch mit dem betroffenen Handymodell beendet und der Versuchsperson wurde das nächste Handy überreicht. Mit zwei Vorkommnissen beim OpenMoko, vier beim Nokia E 71 und drei beim Nokia 6131 war dies die häufigste Ursache für fehlgeschlagene PIN-Eingaben.

Unfähigkeit die PIN einzugeben: Ein Teil der Probanden realisierte nicht, dass es sich bei den Nokia Telefonen nicht um Geräte mit einem Touchscreen handelte und versuchte vergeblich die PIN über den Bildschirm einzugeben. Dies kam zweimal mit dem E 71 und dreimal mit dem Nokia 6131 vor.

Unfähigkeit die PIN-Eingabe zu bestätigen: Einigen Probanden gelang es zwar die PIN einzugeben, sie erkannten jedoch nicht, dass die Eingabe zusätzlich noch bestätigt werden musste. Andere Probanden erkannten diese Notwendigkeit, fanden jedoch keine Möglichkeit zur Bestätigung. Dies hatte einen Fehlschlag mit dem E 71 und einen mit dem Nokia 6131 zur Folge.

Programmfehler: Beim Eingabeprogramm des OpenMoko gab es einen Programmfehler, welcher in einem Fall die korrekte Eingabe der PIN verhinderte. Wurde nach einer Fehleingabe der Popup Dialog, welcher den Nutzer über die inkorrekte Eingabe informierte, nicht via okay bestätigt, sondern stattdessen direkt der PIN-Dialog wieder angewählt, so war eine weitere Eingabe nicht möglich. Der Grund dafür war, dass das Popup im Hintergrund weiter existierte und das Hauptfenster blockierte.

Zum Teil traten die Ursachen auch in Kombination auf. So kam es zum Beispiel vor, dass auf der Suche nach einer Möglichkeit zur Bestätigung der Eingabe das Programm beendet wurde.

4.4.2. Effizienz

Um die Effizienz der PIN-Eingabe zu bewerten, wurde die Zeitspanne vom ersten Tastendruck bis zur erfolgreichen Eingabe gemessen. Dies war natürlich nur bei Versuchspersonen möglich, denen es auch tatsächlich gelang, die PIN korrekt einzugeben, sodass die Stichprobe für die Auswertung der Effizienz kleiner ausfiel als die Originalstichprobe.

Die jüngeren Versuchspersonen erzielten mit dem Nokia 6131 die schnellsten Ergebnisse ($N = 18, M = 7618\text{ms}, Mdn = 6156\text{ms}, SD = 5570\text{ms}$), gefolgt von dem OpenMoko ($N = 18, M = 13498\text{ms}, Mdn = 12089\text{ms}, SD = 7259\text{ms}$). Am längsten dauerte die Eingabe mit dem Nokia E 71 ($N = 18, M = 17987\text{ms}, Mdn = 12873\text{ms}, SD = 13323\text{ms}$). Grafik Abbildung 4.4 stellt die Ergebnisse dieser Gruppe als Boxplots dar.

Bei der Versuchspersonengruppe der älteren Nutzer schnitt das OpenMoko am Besten ab ($N = 12, M = 19191\text{ms}, Mdn = 13402\text{ms}, SD = 11382\text{ms}$). Das Nokia 6131 folgte an zweiter Stelle ($N = 9, M = 44894\text{ms}, Mdn = 20718\text{ms}, SD = 50943\text{ms}$) und wiederum wurden mit dem E 71 die längsten Eingabezeiten benötigt ($N = 8, M = 64520\text{ms}, Mdn = 40595\text{ms}, SD = 59565\text{ms}$). Abbildung 4.4 zeigt die Verteilungen der Eingabedauer als Boxplots.

Zusätzlich wurde der Einfluss der einzelnen Handymodelle auf die benötigte Eingabedauer mittels einer Varianzanalyse (Analysis of Variance (ANOVA)) mit Messwiederholung analysiert. Zunächst wurden die beiden Versuchspersonengruppen gemeinsam betrachtet. Die Analyse zeigte einen tendenziell signifikanten Unterschied ($F(2, 24) = 3.14, \alpha = .05, p = .052$) zwischen den einzelnen Modellen. Bei einer getrennten Betrachtung der beiden Gruppen zeigt sich bei den jüngeren Versuchspersonen ein signifikanter Unterschied ($F(2, 17) = 4.82, \alpha = .05, p = .014$), bei den älteren Versuchspersonen hingegen nicht ($F(2, 5) = 2.06, \alpha = .05, p = .162$).

Als weiteres Maß für die Effizienz der Interaktion wurde die Anzahl der Tastendrücke analysiert, die zur PIN-Eingabe benötigt wurden. Bei der Aufzeichnung der Tastendrücke wurden auch diejenigen Tasten registriert, welche von der Benutzeroberfläche ignoriert wurden, da sie keine sinnvolle Eingabe darstellten.

Minimal waren sieben Tastendrücke notwendig, um die PIN korrekt einzugeben (sechs für die PIN, einer für die Bestätigung). Im Schnitt gelang es den Versuchsper-

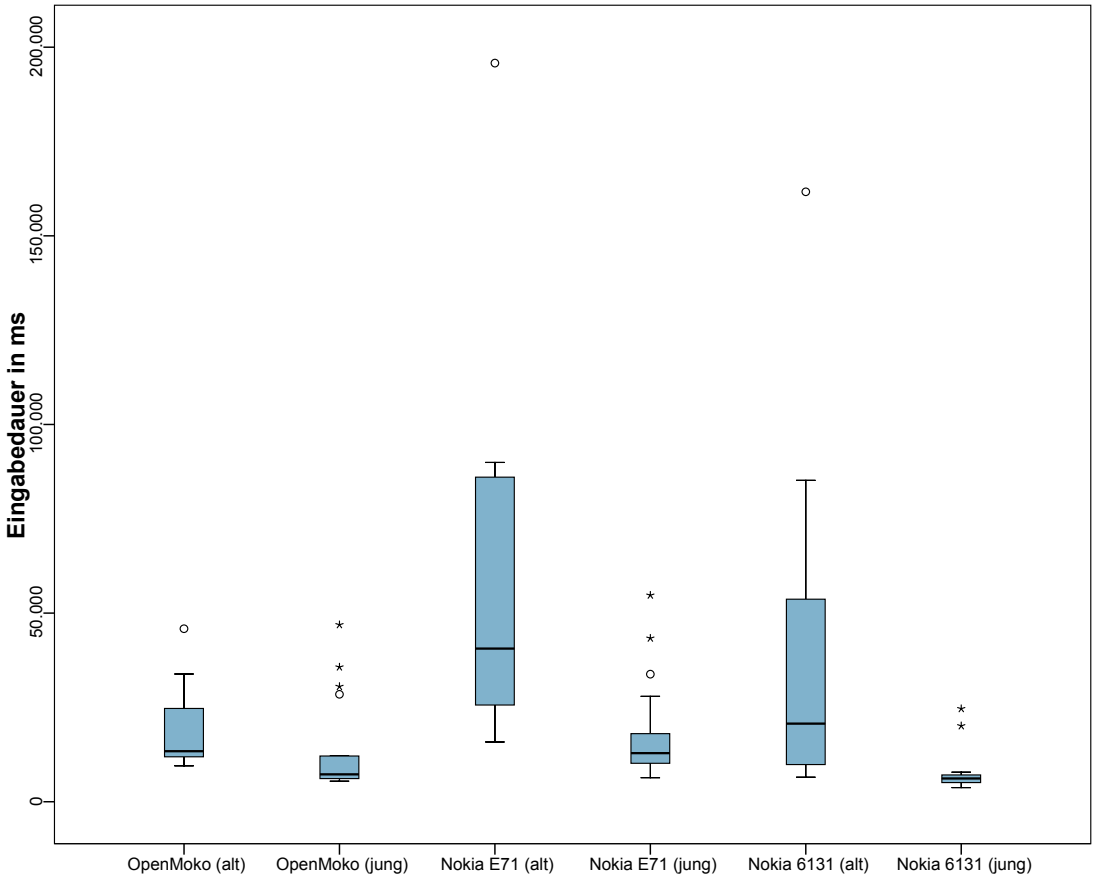

Abbildung 4.4.: Dauer der PIN-Eingabe

sonen der jüngeren Versuchspersonengruppe mit dem Nokia 6131, die PIN mit den wenigsten Tastendrücken einzugeben ($N = 18, M = 7.94, Mdn = 7.0, SD = 4.0$), gefolgt vom OpenMoko ($N = 18, M = 8.78, Mdn = 7.0, SD = 4.5$) und dem Nokia E 71 ($N = 18, M = 9.67, Mdn = 7.0, SD = 8.74$). In der Gruppe der älteren Versuchspersonen gelang die PIN-Eingabe auf dem OpenMoko mit den wenigsten Tastendrücken ($N = 12, M = 7.75, Mdn = 7.0, SD = 2.6$). An zweiter Stelle folgte das Nokia 6131 ($N = 9, M = 10.89, Mdn = 7.0, SD = 7.73$) und mit dem Nokia E 71 wurden die meisten Tastendrücke benötigt ($N = 8, M = 17.0, Mdn = 8.5, SD = 14.39$). Abbildung 4.5 stellt die Ergebnisse als Boxplot dar.

Um einen eventuellen Lerneffekt aufzudecken, wurde die Eingabezeit mit den einzelnen Telefonen einer Varianzanalyse mit Messwiederholung unterzogen, diesmal unter Berücksichtigung der Reihenfolge, in welcher die Handys an die Probanden ausgegeben wurden. Ziel war es festzustellen, ob die Probanden im Laufe des Versuches schnellere Eingabezeiten erzielten als zu Beginn. Bei der Analyse wurden die beiden Versuchspersonengruppen gemeinsam betrachtet. Ein signifikanter Lerneffekt war jedoch nicht festzustellen ($F(2, 24) = 1.18, \alpha = .05, p = .32$). Auch bei getrenn-

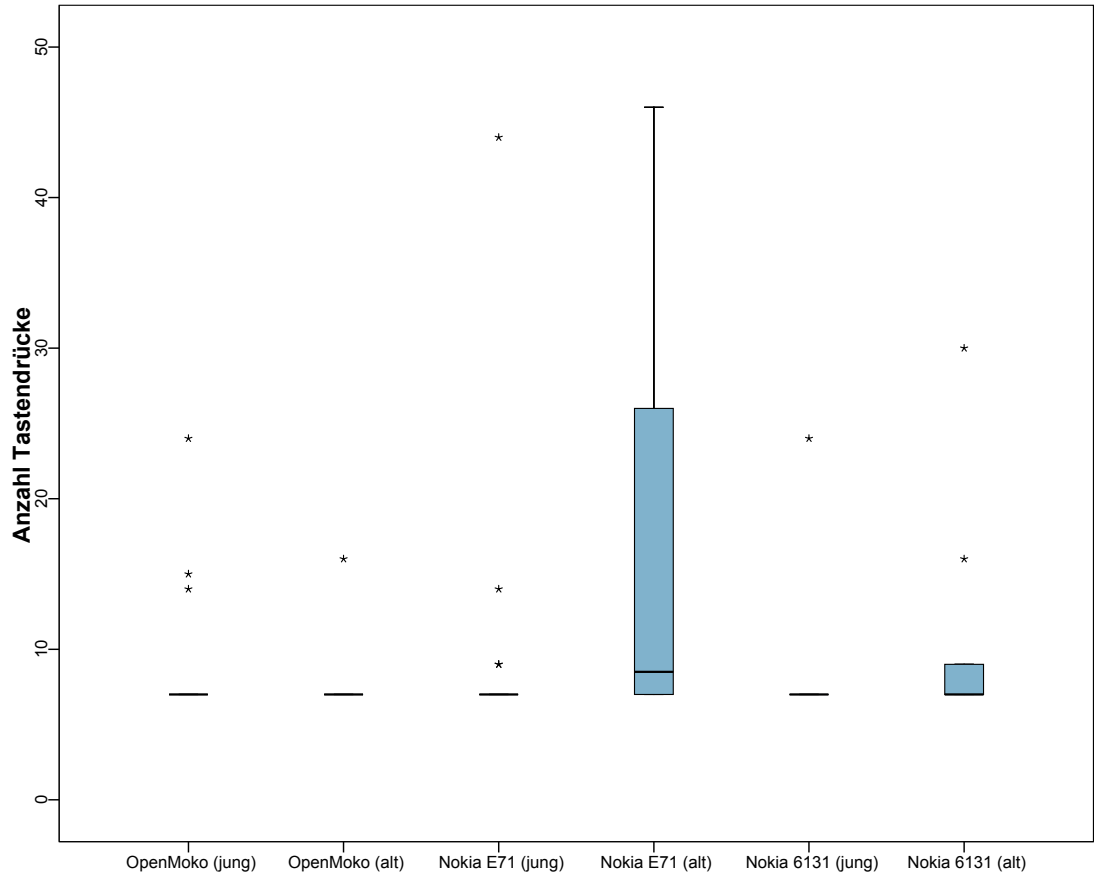

Abbildung 4.5.: Anzahl der Tastendrücke

ter Betrachtung der beiden Gruppen war weder bei den älteren ($F(2,5) = .86, \alpha = .05, p = .45$) noch bei den jüngeren Probanden ($F(1.38,17) = 2.5, p = .12$) ein signifikanter Effekt erkennbar. Da bei der jüngeren Probandengruppe die Sphärizitätsannahme verletzt war, wurde hier eine Greenhouse-Geisser Korrektur verwendet.

4.4.3. Zufriedenheit

Die subjektive Bewertung der PIN-Eingabe auf den verschiedenen Telefonen mittels der siebenstufigen Likert-Skala auf dem Fragebogen wurde zur Erfassung der Nutzerzufriedenheit verwendet. Abbildung 4.6 stellt die Bewertungen der einzelnen Telefone durch die beiden Versuchsgruppen nebeneinander.

Das OpenMoko erhielt hier in der Versuchspersonengruppe der älteren Probanden die besten Bewertungen ($M = 5.53, SD = 1.5$). An zweiter Stelle rangierte das Nokia 6131 ($M = 3.67, SD = 1.88$), gefolgt vom Nokia E 71 ($M = 2.6, SD = 1.4$). Bei den jüngeren Probanden schnitt das Nokia 6131 ($M = 5.33, SD = 1.68$) minimal besser ab als das OpenMoko ($M = 5.11, SD = 1.49$). Auch in dieser Gruppe wurde das

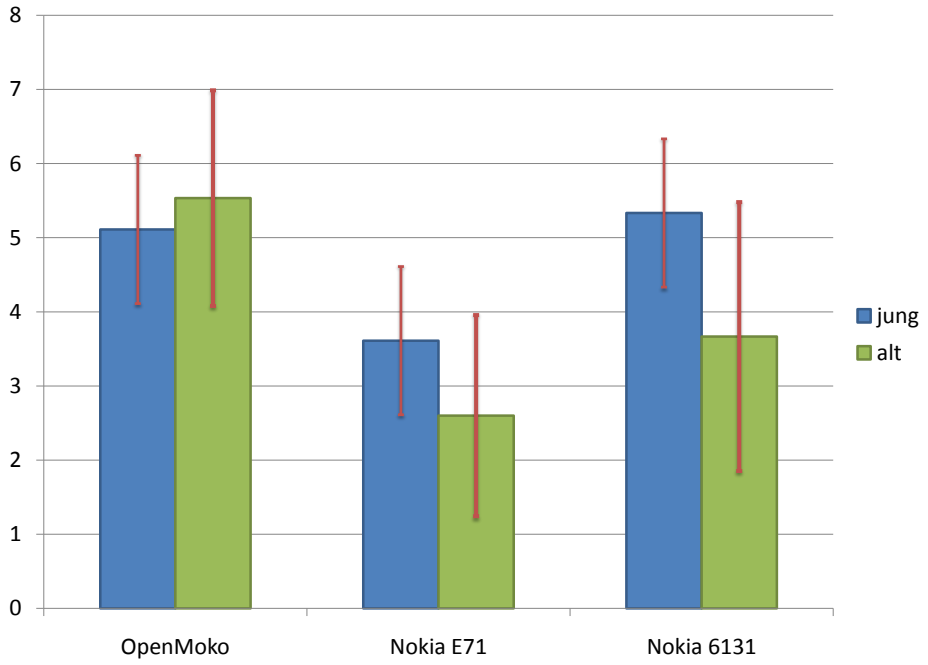

Abbildung 4.6.: Subjektive Bewertung der verwendeten Mobiltelefone

Nokia E71 am schlechtesten bewertet ($M = 3.61, SD = 1.75$).

T-Tests für unabhängige Stichproben bestätigen, dass es einen signifikanten Unterschied in der Bewertung des Nokia 6131 zwischen den beiden Versuchspersonengruppen gibt ($t[31] = 2.69, p = .011$). Die Bewertungen für das OpenMoko ($t[31] = -.8, p = .43$) und das E 71 ($t[31] = 1.8, p = .08$) lassen hingegen keine signifikanten Unterschiede zwischen den Gruppen erkennen.

4.5. Qualitative Versuchsdaten

Die qualitative Versuchsauswertung stützte sich auf die Anmerkungen, welche die Versuchspersonen auf dem Fragebogen in Freitextform machen konnten, auf Fragen, die vor oder nach dem Versuch gestellt wurden, und auf während des Versuches geäußerte Bemerkungen der Probanden. Außerdem flossen Beobachtungen des Versuchsleiters in die Auswertung mit ein.

Eine erste Beobachtung war, dass ein großer Teil der Versuchspersonen sich nicht darüber bewusst war, dass ein neuer Personalausweis geplant ist. Einige Versuchspersonen reagierten überrascht auf das Vorlesen der Versuchsbeschreibung. Besonders verärgert reagierte eine Versuchsperson, die als Polizist arbeitet und erst durch den Versuch von dem neuen Personalausweis erfuhr. Der Versuch fand Ende April 2010 statt, also circa ein halbes Jahr vor Einführung des neuen Ausweises. Einige der Ver-

suchspersonen zeigten Anzeichen von Nervosität und hatten offensichtlich Angst sich zu blamieren. In diesen Fällen wurden sie vom Versuchsleiter darauf hingewiesen, dass sie nichts kaputt machen können und dass es genau das Ziel des Versuches sei, Probleme im Umgang mit den Telefonen und dem Ausweis zu finden.

Die weiteren Beobachtungen lassen sich in die Themengebiete Handhabung, Technik und Sicherheit sowie grafische Nutzeroberfläche einteilen.

4.5.1. Handhabung

Wenig überraschend nach den Ergebnissen der quantitativen Analyse ist, dass sich insgesamt neun Versuchspersonen über eine schlechte Bedienbarkeit des Nokia E71 beklagten. Hauptkritikpunkt waren die kleinen Tasten. Ein Proband der älteren Versuchspersonengruppe bezeichnete dieses Gerät als „völlig ungeeignet“. Mit fünf jüngeren und vier älteren Versuchspersonen, die sich zu den zu kleinen Tasten äußerten, waren beide Versuchspersonengruppen betroffen. Weitere Kritik an den verwendeten Telefonen bezog sich auf schlechte Ablesbarkeit der Displays in hellem Sonnenlicht und auf die sich lösende Schutzfolie des OpenMoko.

Vier der jüngeren Versuchspersonen beklagten zusätzlich eine schlechtere Handhabbarkeit der Telefone in Verbindung mit dem Personalausweis. Zwei dieser Personen wiesen darauf hin, dass dies vor allem beim Nokia 6131 der Fall sei. Dieses Telefon ist schmaler als die Ausweiskarte, wodurch es schwieriger ist, den Ausweis hinter dem Telefon zu halten.

Einige Versuchspersonen stellten Fragen zur korrekten Haltung des Ausweises. In diesen Fällen wurden sie gebeten, den Ausweis so an das Handy zu halten, wie es ihnen intuitiv richtig erschien. Auf diese Art sollten bevorzugte Haltungsarten erkannt werden.

Da nur vage Anweisungen gegeben wurden, wie Karte und Handy zu halten waren, variierte die Haltung sehr stark. Folgende vier Kategorien ließen sich unterscheiden:

- Die Versuchsperson hielt das Handy in der Hand und legte den Ausweis vor sich auf einen Tisch. Dies war die häufigste Art der Haltung. Bei den älteren Versuchspersonen hielten 13 Probanden das OpenMoko, neun das Nokia E 71 und ebenfalls neun das Nokia 6131 auf diese Weise. Von den jüngeren Versuchspersonen waren es acht beim OpenMoko, neun beim E 71 und neun beim Nokia 6131.

- Die Versuchsperson hielt Ausweis und Karte in der Hand. Diese Haltung wählten vor allem die jüngeren Probanden mit jeweils neun Versuchspersonen bei jedem der drei Handymodellen. Bei den älteren Versuchspersonen waren es jeweils zwei Versuchspersonen pro Handymodell. Zumeist wurde der Ausweis hierbei hinter dem Handy gehalten. Lediglich vereinzelt war zu beobachten, dass er vor dem Handy oder aber in der zweiten Hand gehalten wurde. Bei zwei der Versuchspersonen, die den Ausweis vor das Handy hielten, stellte er eine Behinderung bei der Eingabe der PIN dar.

- Zwei der älteren Versuchspersonen legten jeweils das E 71 und das Nokia 6131 mitsamt dem Ausweis auf den Tisch. Eine der jüngeren Versuchspersonen wählte diese Haltungsweise beim OpenMoko.
- Zwei der älteren Versuchspersonen hielten den Ausweis in der Hand, während sie die PIN auf dem auf dem Tisch liegenden Telefon eingaben. Auf diese Weise handhabten sie das E 71 und das Nokia 6131.

Sechs der älteren und zwei der jüngeren Versuchspersonen wechselten die Haltung zwischen den einzelnen Handymodellen.

In vielen Versuchen wurde der Ausweis in einem verhältnismäßig großen Abstand zum Telefon gehalten, sodass in der Praxis keine Kommunikation zwischen Chipkarte und Mobiltelefon möglich gewesen wäre. Zwei der älteren Versuchspersonen legten den Ausweis nach der Eingabe der PIN auf das Handy. Eine weitere Versuchsperson versuchte zunächst den Ausweis in das Handy zu stecken. Sie ging offenbar von der Funktionsweise einer kontaktbehafteten Karte aus.

4.5.2. Sicherheit und Technik

Bedenken zur Sicherheit des Personalausweises waren vielfältig und weit verbreitet. Zum einen äußerten die Versuchspersonen Bedenken, dass ihre Daten von unberechtigten Dritten abgefangen werden könnten (drei ältere und zwei jüngere Versuchspersonen). Vor allem bezüglich der Funktechnik äußerten sich die Versuchspersonen kritisch (drei ältere, zwei jüngere).

Zusätzlich gab es Bedenken zur Überwachung durch den Staat (sechs junge und eine alte Versuchsperson), vor allem in Hinblick auf die eingesetzte Biometrie (vier junge Versuchspersonen). Eine der älteren Versuchspersonen meinte hierzu: „Als ich jünger war, habe ich mit Begeisterung 1984 von George Orwell gelesen. Heute lachen die da nur drüber“. Diese Bedenken können als erste Indikatoren für eine mangelnde Akzeptanz der verwendeten Technologien angesehen werden.

Die verwendete Technik stellte einige Versuchspersonen vor Verständnisschwierigkeiten. Zwei der jüngeren Versuchspersonen fragten nach, ob sie den Ausweis vor die Kamera des Handys halten müssten, um die Funkverbindung herzustellen. Einer der älteren Probanden versuchte auf dem OpenMoko zunächst die PIN mit dem Stift in das Ausgabefeld der GUI zu schreiben.

Andere Probanden stellten gezielte Detailfragen zur Technik, etwa zur Funkreichweite oder zur Position des Chips im Ausweis.

4.5.3. Grafische Oberfläche

Die Gestaltung der grafischen Oberfläche verwirrte einen großen Teil der Versuchspersonen. Durch die Abbildung eines Pinpads wurde suggeriert, dass es sich bei allen drei Telefonen um Geräte mit Touchscreen handele, auch wenn die beiden Nokia Modelle über Tastaturen verfügten. Dies hatte zur Folge, dass eine Vielzahl an Probanden versuchten, die PIN über den Bildschirm einzugeben. Offenbar erzeugte die

grafische Oberfläche beim Nutzer ein falsches mentales Modell (vergleiche [Nor02]) des Systems. Dieser Effekt war beim Nokia E 71 stärker ausgeprägt (zwölf der älteren, neun der jüngeren Versuchspersonen) als beim Nokia 6131 (zehn der älteren, zwei der jüngeren Probanden). Eine mögliche Ursache dafür ist der größere Bildschirm des E 71, welcher eher als Touchscreen geeignet wäre.

Die Stärke des erzeugten mentalen Modells zeigte sich bei einigen der Versuchspersonen: selbst nachdem sie herausgefunden hatten, dass die PIN über die Tastatur und nicht über den Bildschirm eingegeben werden muss, versuchten sie die Eingabe über den Bildschirm zu bestätigen. Generell war zu beobachten, dass es den jüngeren Versuchspersonen meist gelang, die PIN noch korrekt einzugeben - trotz anfänglicher Irrtümer über die Funktionsweise der Handys . Sie waren also in der Lage, ihr anfängliches mentales Modell anzupassen. Den älteren Versuchspersonen gelang dies oft nicht. Sie versuchten teilweise mehrere Minuten lang, die PIN über das Display einzugeben und schafften es oft nicht, eine neue Strategie zu entwickeln. Die Zeit, welche bis zum ersten Tastendruck verging, wurde jedoch nicht genauer ausgewertet.

Darüber hinaus deckten die Probanden eine Reihe sehr konkreter Gestaltungsfehler in den Benutzeroberflächen auf und gaben wertvolles Feedback für mögliche Verbesserungen. Bemängelt wurden beispielsweise eine zu kleine Beschriftung der Tasten auf dem OpenMoko und allgemein eine zu träge Reaktion der GUI auf diesem Handy.

Mehrere Versuchspersonen wünschten sich ausführlichere Instruktionen auf dem Bildschirm oder in Form einer Bedienungsanleitung. Ein Proband schlug vor, beim Starten des Programmes einen Ladebildschirm mit Hinweisen zur korrekten Haltung von Ausweis und Handy anzuzeigen. Ein anderer Proband schlug vor, nach der Eingabe von sechs Ziffern einen Pfeil auf die Bestätigungstaste einzublenden, um das Problem zu beseitigen, die richtige Taste hierfür zu finden. Ein weiterer Vorschlag zu diesem Problem war, komplett auf eine Bestätigung der Eingabe zu verzichten und statt dessen direkt die Eingabe zu verwenden, sobald sechs Ziffern eingegeben sind.

4.6. Diskussion der Versuchsergebnisse

Bei der Interpretation der geringen Effektivität der PIN-Eingabe in der Versuchsgruppe der älteren Nutzer muss beachtet werden, dass die den Probanden gestellte Aufgabe sehr viel anspruchsvoller war, als sie auf den ersten Blick erscheint. Die Versuchspersonen hatten größtenteils keine Erfahrungen mit den eingesetzten Handy Modellen. Die Verwendung des Ausweises war neuartig und band die Aufmerksamkeit der Versuchspersonen. Man kann davon ausgehen, dass einige der Probleme aus dem Versuch so in der Praxis nicht auftreten würden. Beispielsweise wüssten die Probanden bei ihrem eigenen Handy vermutlich, ob es per Touchscreen oder per Tastatur bedient werden muss.

Weiterhin liegt die Vermutung nahe, dass die Versuchspersonen in der Praxis den Umgang mit dieser neuen Authentisierungsmethode erlernen würden, auch wenn im Versuch kein signifikanter Lerneffekt bei den älteren Versuchspersonen festgestellt werden konnte. Ein Grund hierfür ist, dass die Stichprobe durch die hohe Anzahl

älterer Versuchspersonen, denen es nicht gelang die PIN korrekt einzugeben, stark dezimiert wurde. Wertet man lediglich die Daten der jüngeren Versuchspersonengruppe aus, so lassen sich zumindest erste Anzeichen für eine Abhängigkeit der Dauer der PIN-Eingabe vom verwendeten Handymodell und einen Lerneffekt bei wiederholter PIN-Eingabe verzeichnen. Um statistisch signifikante Aussagen zu treffen, wäre jedoch ein Versuch mit einer größeren Stichprobe und der Verwendung von Kontrollgruppen notwendig.

Die Auswertung der Anzahl der Tastendrücke zeigt, dass der Großteil der Versuchspersonen, denen es gelang die PIN einzugeben, dies auch mit der minimalen Anzahl von sieben Tastendrücken schafften. Einige Ausreißer führten allerdings zu einer starken Verschiebung der Mittelwerte. Dies wird auch bei Betrachtung der Boxplots aus Abbildung 4.5 deutlich. Es gab zwei Ursachen für diese Ausreißer: fehlerhafte PIN-Eingaben und die Verwendung der alternativen Eingabestrategie bei den beiden Nokia Telefonen. Beide Ereignisse traten nur selten auf. So gab lediglich eine Versuchsperson auf den Nokia Telefonen die PIN mittels des Steuerkreuzes ein. Auch die Anzahl der fehlerhaften PIN-Eingaben war gering. Jedoch hätte sie bei zwei Probanden eine Sperrung des Personalausweises zur Folge gehabt (mehr als drei Fehleingaben). Zwei weitere Versuchspersonen gaben jeweils zweimal eine falsche PIN ein und hätten die eID-PIN somit suspendiert.

Der Versuch macht deutlich, dass die Gebrauchstauglichkeit der eID-Authentisierung von einer Reihe von Faktoren abhängig ist. Der Formfaktor des verwendeten Mobiltelefons und die Gestaltung der grafischen Benutzeroberfläche scheinen dabei eine wichtige Rolle zu spielen. Der Versuch lieferte keine konkreten Anhaltspunkte dafür, dass die gleichzeitige Handhabung von Ausweis und Mobiltelefon grundsätzlich problematisch ist, da viele Versuchspersonen das Handy mitsamt dem Ausweis vor sich auf einen Tisch legten. Bei einer Wiederholung des Versuches sollten daher auch Verbesserungen im Versuchsaufbau vorgenommen werden. So sollten die Nutzer instruiert werden, die Telefone in der Hand zu halten, anstatt sie auf einen Tisch zu legen. Auch wäre die Durchführung des Versuches im Stehen eventuell schwieriger für die Versuchspersonen als im Sitzen. Dadurch könnten möglicherweise weitere Probleme bei der Handhabung aufgedeckt werden, sodass der Versuch stärker auf das Problem der Handhabung fokussiert werden würde.

Trotz aller Kritikpunkte liefert der Versuch wichtige Anhaltspunkte für die weitere Entwicklung. Zum einen gaben die Versuchspersonen konkrete Gestaltungshinweise zur Verbesserung der Nutzeroberfläche. Zum anderen ergaben sich aus den im Versuch aufgetretenen Problemen wertvolle Hinweise darauf, welche Punkte bei der weiteren Entwicklung zu beachten sind. Beispielsweise wurde die Notwendigkeit von Benutzungsanweisungen in der grafischen Nutzeroberfläche deutlich. Daraus, dass viele Nutzer den Ausweis außerhalb der Funkreichweite hielten, kann man schließen, dass es wichtig ist, dem Nutzer eine Rückmeldung zur Erreichbarkeit des Chips zu geben.

Insgesamt zeigt der Versuch, dass Usability Untersuchungen in den frühen Entwicklungsphasen eines Produktes dabei helfen können, mögliche Probleme, wie etwa das durch die Nokia GUIs erzeugte falsche mentale Modell, frühzeitig aufzudecken.

Die Entfernung eines solchen fundamentalen Fehlers wäre in den späteren Phasen der Entwicklung um ein Vielfaches aufwendiger und damit kostspieliger.

5. Implementierung

Nach den Untersuchungen aus Kapitel 4 stand die prototypische Entwicklung eines Systems zur Nutzung der eID-Authentisierung auf einem Mobiltelefon an. Ziel der Entwicklung war es, von einem Handy aus erfolgreich das PACE-Protokoll mit dem nPA durchzuführen. Weiterhin sollte eine grafische Benutzeroberfläche für die eID-Authentisierung entwickelt werden, welche als Basis für weitere Usability-Untersuchungen genutzt werden kann.

Explizit von der Entwicklung ausgenommen war die sogenannte eCard-API. Hierbei handelt es sich um ein Framework zur einheitlichen Anbindung aller aktuellen Smartcard-basierten Projekte der deutschen Bundesregierung. Hierzu zählen neben dem elektronischen Personalausweis auch die elektronische Gesundheitskarte, ELSTER und ELENA. Die Spezifikation der eCard-API ist sehr umfangreich und eine Implementierung hätte den Rahmen dieser Diplomarbeit gesprengt.

Teil der eCard-API ist die Spezifikation des Protokolls zur Kommunikation zwischen dem lokalen Terminal und dem Dienstanbieter über das Internet. Aufgrund des Fehlens dieser Schnittstelle ist die Kommunikation mit einem Dienstanbieter und somit die tatsächliche Durchführung der eID-Authentisierung mit dem hier vorgestellten System noch nicht möglich. An der Entwicklung einer für Mobiltelefone geeigneten Variante der eCard-API wurde zum Zeitpunkt des Verfassens dieser Arbeit jedoch bereits gearbeitet. (siehe [Bei10]). Sobald diese Entwicklung abgeschlossen wird, kann das hier vorgestellte System um diese „mobile“ eCard-API erweitert werden, um eine voll funktionsfähige Lösung zu erreichen.

5.1. Hardware

Als Mobiltelefon-Plattform wurde das OpenMoko ausgewählt. Diese Entscheidung wurde zum einen aufgrund des guten Abschneidens in der Usability Untersuchung aus Kapitel 4, zum anderen aufgrund der guten Programmierbarkeit und Offenheit der Plattform gewählt. Als Betriebssystem kam wieder SHR-Testing zum Einsatz.

5.1.1. Erweiterung um eine NFC-Schnittstelle

Da dass OpenMoko nicht über eine NFC-Schnittstelle verfügt, musste zunächst eine Möglichkeit gefunden werden, diese nachzurüsten. Hierfür wurden die folgenden Alternativen evaluiert:

MyMax NFC Sticker[1] Der MyMax NFC Sticker ist ein Aufkleber, welcher auf einem Handy aufgebracht werden kann, um dieses um NFC-Funktionalität zu erweitern. Die im Aufkleber enthaltene Elektronik stellt zum einen eine NFC-Schnittstelle, zum anderen ein Bluetooth Interface zur Verfügung. Somit kann der Sticker vom Mobiltelefon aus über Bluetooth angesprochen und auf diese Art und Weise seine NFC-Schnittstelle angesteuert werden.

Externer Anschluss eines RFID Lesegeräts Eine der Besonderheiten des OpenMoko ist, dass der USB-Anschluss des Telefons sowohl im Host als auch im Device Modus betrieben werden kann. Der Device Modus ist dabei der Standard Modus, in welchem das Telefon an einen Computer angeschlossen und dort als USB-Gerät verwendet werden kann. Im Host Modus kann man hingegen USB-Geräte an das Telefon anschließen und diese von dort aus verwenden. Welche Geräte dabei unterstützt werden, hängt von dem auf dem Handy laufenden Betriebssystem ab. Der verwendete Kernel muss die Module für die jeweiligen USB-Geräteklassen enthalten. Prinzipiell können mit dem OpenMoko alle vom Linux Kernel unterstützen USB-Geräte verwendet werden.

Der Wechsel zwischen USB Host Modus und USB Device Modus kann zur Laufzeit via Software erfolgen. Listing 5.1 zeigt die Befehle zum Umschalten in den Host Modus und zur Aktivierung der USB Stromversorgung.

```
1 echo host > /sys/devices/platform/s3c-ohci/usb_mode
2 echo 1 > /sys/devices/platform/s3c2440-i2c/i2c-adapter/i2c
    -0/0-0073/neo1973-pm-host.0/hostmode
```

Listing 5.1: Umschalten des OpenMoko in den USB Host Modus

Im Host Modus kann nun ein RFID Lesegerät an den externen USB-Anschluss des OpenMoko angeschlossen werden. Da der USB Port des OpenMoko das Format Mini-USB B hat, RFID Lesegerät typischerweise jedoch einen USB-A Stecker haben, ist dafür noch ein Adapterkabel notwendig.

Interner Anschluss eines RFID-Lesegeräts Auf der Platine des OpenMoko sind Kontakte für einen weiteren USB-Anschluss und für den I^2C (sprich „I square C“) Bus vorhanden. Bei Letzterem handelt es sich um einen seriellen Bus, welcher nach dem Master-Slave Prinzip operiert, wobei auch mehrere Master an einem Bus angeschlossen sein können. Die Spezifikation des Kommunikationsprotokolls ist frei verfügbar und darf lizenzkostenfrei umgesetzt werden.

Sowohl bei der Verwendung des I^2C Busses als auch des internen USB-Anschlusses müssen zusätzliche Bemühungen zur Versorgung des Lesegerätes mit Strom getroffen werden, da für beide Busse lediglich die Datenleitungen auf der Platine des OpenMoko verfügbar sind.

[1] Twinlinx: *MyMax: Mobile Contactless Sticker*. Verfügbar unter http://twinlinx.com/mymaxsticker.php. Zuletzt überprüft am: 07.08.2010

Beim Anschluss an den USB-Bus kann prinzipiell jeder RFID-Reader mit USB-Anschluss verwendet werden. Zum Anschluss an den I^2C eignen sich beispielsweise der OpenPCD[2] oder der OpenPICC[3].

Vergleicht man die oben beschriebenen Lösungsansätze, so erscheint zunächst der MyMax Sticker als die vielversprechendste Lösung. Das Anbringen des Aufklebers erfordert keine aufwendige Hardwareinstallation und die Ergonomie des Telefons wird durch den Aufkleber kaum beeinflusst. Leider waren zum Zeitpunkt der Anfertigung dieser Arbeit kaum weiterführende Informationen über den MyMax Sticker öffentlich verfügbar. Eine Anfrage beim Hersteller erbrachte die Auskunft, dass der Aufkleber über eine J2ME Software angesprochen werden kann. Des Weiteren ist das Bluetooth Protokoll, welches zur Kommunikation zwischen Mobiltelefon und Aufkleber verwendet wird, in der Dokumentation offengelegt. Somit wäre auch die Ansteuerung des Aufklebers mittels selbst entwickelter Software denkbar.

Leider ist der Aufkleber nicht in der Lage APDUs, welche größer als 64 Byte sind, zu verschicken. Für die Kommunikation mit dem Ausweis sind aber wesentlich größere Datenpakete notwendig, allein die Terminalzertifikate sind typischerweise mehrere Hundert Byte groß. Aus diesem Grund konnte der MyMax NFC Sticker nicht verwendet werden.

Die anderen beiden Möglichkeiten unterscheiden sich vor allem im Aufwand der Umsetzung. Der Vorteil der Verwendung des USB Anschlusses ist, dass diese Lösung auch ohne Lötarbeiten durchgeführt werden kann und der externe USB Anschluss den Leser mit Strom versorgt. Bei der Verwendung des I^2C Busses oder der USB Testpunkte auf der Vorderseite der OpenMoko Platine müsste man sich hingegen noch um eine zusätzliche Stromversorgung bemühen.

Sowohl der interne als auch der externe Anschluss eines RFID-Readers erfordern die Unterbringung eines zusätzlichen Gerätes (des RFID-Readers) im oder am Telefon. Die Lösung dieses Problems wird weiter unten in Unterabschnitt 5.1.2 beschrieben. Beim externen Anschluss geht außerdem der externe USB-Port des OpenMoko für andere Anwendungen verloren.

Nach der Abwägung dieser Vor- und Nachteile wurde der externe Anschluss eines RFID-Readers per USB als Lösung ausgewählt. Ausschlaggebend für diese Entscheidung war zum einen die leichte Umsetzbarkeit der Lösung. Zum anderen sollte das Risiko, das OpenMoko bei Lötarbeiten auf der Platine zu beschädigen, vermieden werden.

Die Wahl des zu verwendenden RFID-Readers fiel auf den „Touchatag“[4]. Dieser zeichnet sich zum einen durch einen günstigen Preis aus (zum Zeitpunkt der Anfertigung dieser Arbeit ca. 40 Euro). Zum anderen ist in diesem Lesegerät Standard-

[2] *Open RFID reader*. Verfügbar unter: http://www.openpcd.org/about.0.html. Letzter Zugriff: 09.08.2010

[3] *OpenPICC*. Verfügbar unter: http://www.openpcd.org/openpicc.0.html. Letzter Zugriff: 09.08.2010

[4] Touchatag: *The touchatag RFID reader*. Verfügbar unter: http://www.touchatag.com/touchatag-rfid-reader. Letzter Zugriff: 07.08.2010

Hardware verbaut. Intern wird eine ACR-122u Platine verwendet, auf welcher wiederum ein NXP PN532 Funkchip verbaut ist. Die Platine ist via CCID ansprechbar und der Funkchip wird von libnfc unterstützt (genauere Informationen zur Software finden sich in Abschnitt 5.2). Dadurch lässt sich der Reader einfach ansprechen und benötigt keine proprietären Treiber, welche für eine Mobilfunkplattform höchstwahrscheinlich nicht verfügbar wären.

Ein Nachteil des Touchatag Readers ist, dass der verbaute PN532 Chip keine extended length APDUs - also APDUs mit mehr als 255 Byte payload - unterstützt, sondern lediglich bis zu 250 Byte Daten versenden bzw. empfangen kann. Dies reicht aber, im Gegensatz zum MyMax NFC Sticker, zumindest für das Durchführen von PACE aus. Soll das System zu einem späteren Zeitpunkt ausgebaut werden, sodass die eID-Authentisierung voll funktionsfähig ist, so muss auch das verwendete Lesegerät ausgetauscht werden. Auf diese Tatsache wurde bei der Implementierung der Anbindung des Lesegeräts (siehe Unterabschnitt 5.2.2) Rücksicht genommen, indem die Anbindung so flexibel wie möglich gestaltet wurde, um den Austausch der verwendeten Reader-Hardware zu vereinfachen.

5.1.2. Gehäuseintegration

Die Notwendigkeit, ständig ein zusätzliches RFID-Lesegerät mit sich herum zutragen und dieses gleichzeitig mit dem OpenMoko zu handhaben, wäre für die meisten Nutzer keine praktikable Lösung. Daher musste eine Möglichkeit gefunden werden, das Lesegerät im Telefon zu integrieren.

Im Gehäuse des OpenMoko ist nicht genug Platz für eine zusätzliche Platine vorhanden. Daher musste das Gehäuse verändert werden, um zusätzlichen Platz zu schaffen. Hierfür kam die vom OpenMokast Projekt[5] entworfene Gehäuseerweiterung zum Einsatz. Diese wurde ursprünglich entworfen, um einen DAB Receiver zum Fernsehempfang in das OpenMoko einzubauen.

Auch das umgebaute Gehäuse verfügte nicht über genügend Platz, um den kompletten Touchatag aufzunehmen. Daher wurde zunächst die Platine des Touchatag aus ihrem Gehäuse entfernt. Sie ist 9,1 cm lang und 5,9 cm breit und war damit minimal zu breit für die Unterbringung im Gehäuse des OpenMoko. Da sich an den Rändern der Platine jedoch keine Bauteile oder Leiterbahnen befinden, war es möglich, einen Teil der Platine abzufräsen, ohne die Funktionalität als Lesegerät zu beeinträchtigen.

Weiterhin wurde noch das USB Kabel des Touchatag entfernt und durch ein Flachbandkabel mit einem Mini-USB Typ B Stecker ersetzt. Abbildung 5.1 zeigt die bearbeitete Platine.

[5] Communications Research Centre Canada: *Case extension*. Verfügbar unter: `http://openmokast.org/cad-files.html`. Letzter Zugriff: 07.08.2010

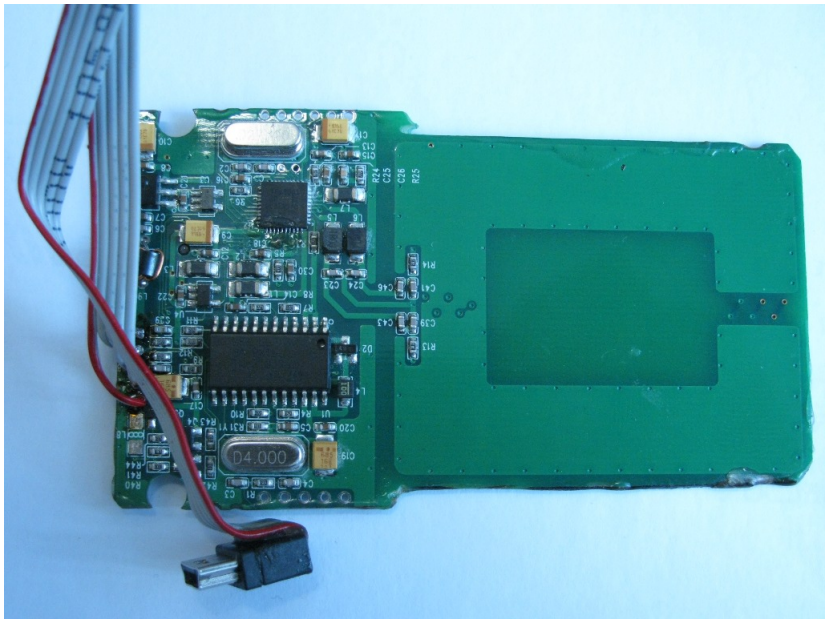

Abbildung 5.1.: Bearbeitete Platine des Touchatag

Um das Flachbandkabel aus dem Gehäuse herauszuführen, wurde noch eine zusätzliche Aussparung in die Seite des Gehäuses gefräst. Anschließend konnte die Platine im Gehäuse untergebracht und an den externen USB-Port angeschlossen werden. Das fertig zusammengebaute Telefon mit eingebauten RFID-Reader ist in Abbildung 5.2 zu sehen.

Abbildung 5.2.: OpenMoko mit Gehäuseerweiterung und eingebautem Reader

5.2. Software

Die Software, welche für die eID Authentisierung auf dem Handy benötigt wird, lässt sich in drei Schichten einteilen:

1. Grafische Benutzeroberfläche: Sie realisiert die Interaktion mit dem Nutzer.
2. Applikationslogik: In dieser Schicht sind die Funktionen zum Zugriff auf den Personalausweis realisiert. Dazu gehören zum einen die für die general authentication procedure benötigten kryptografischen Mechanismen, zum anderen die eigentliche Funktionalität, wie etwa das Ändern der PIN.
3. Smartcard Anbindung: Hierzu zählen alle Komponenten, die den eigentlichen Datenaustausch mit dem Personalausweis ermöglichen.

Jede dieser drei Schichten besteht wiederum aus mehreren Softwarekomponenten. Abbildung 5.3 zeigt die Softwarearchitektur.

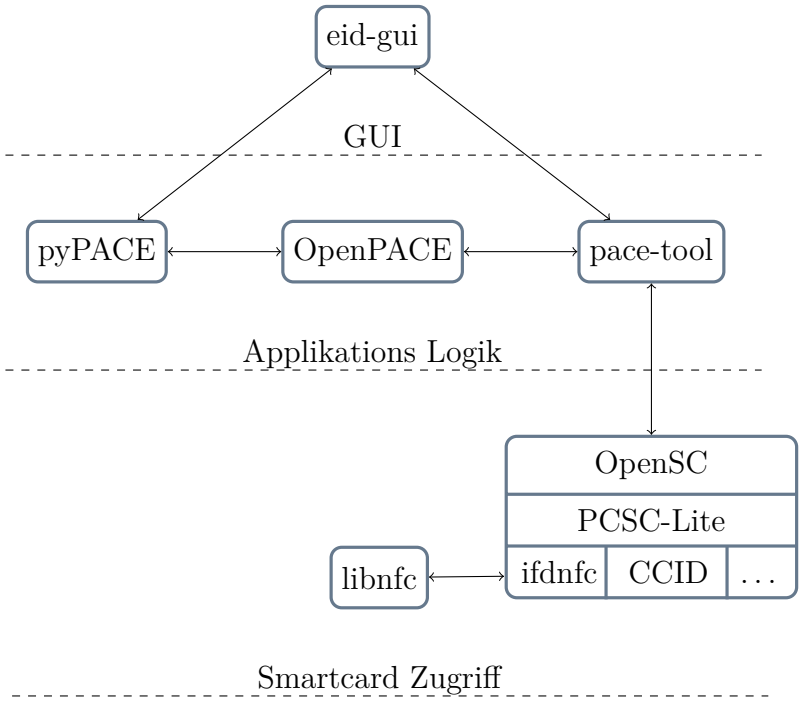

Abbildung 5.3.: Software Architektur

Der Quellcode aller Programme der Applikations- und GUI-Schicht befindet sich auf der beigelegten CD im Verzeichnis `src`. Im Folgenden werden die einzelnen Softwarekomponenten der Applikations- und der Smartcardebene erläutert. Auf die Gestaltung der grafischen Oberfläche wird gesondert in Abschnitt 5.3 eingegangen. Eine Anleitung zum Kompilieren des gesamten Systems findet sich in Anhang C.

5.2.1. Anwendungsbene

Die Kernkomponente der Anwendungsschicht ist **OpenPACE**[6]. Diese Bibliothek stellt die notwendigen kryptografischen Funktionen für die Kommunikation mit dem Personalausweis bereit. OpenPACE ist eine Erweiterung für die Bibliothek libcrypto, welche Teil von OpenSSL ist. Sie ergänzt OpenSSL um Funktionen für die Durchführung der einzelnen Schritte des PACE Protokolls und zur Verarbeitung von CV-Zertifikaten. PACE wurde mit Unterstützung von Diffie-Hellmann und elliptische Kurven umgesetzt. Es wird in beiden Fällen sowohl das generic mapping als auch das integrated mapping unterstützt (zur Erläuterung der verschiedenen mapping Verfahren siehe Anhang A).

OpenSSL wurde aufgrund der weiten Verbreitung und der hohen Portabilität als Grundlage für OpenPACE gewählt. Diese beiden Anforderungen waren wichtig, da OpenPACE auch auf mobilen Plattformen eingesetzt werden soll. Mit der zunehmenden Verbreitung von Linux im Bereich der mobilen Betriebssysteme ist anzunehmen, dass OpenPACE auch in Zukunft auf einer Vielzahl von Plattformen eingesetzt werden kann.

OpenPACE bietet eine generische Implementierung des PACE Protokolls. Die Bibliothek kann nicht nur in Verbindung mit dem elektronischen Personalausweis genutzt werden, sondern es wäre auch denkbar sie einzusetzen, um das Protokoll in anderen Szenarien zu verwenden. Die Formatierung der APDUs oder andere Smartcard-spezifische Funktionen sind daher nicht in OpenPACE enthalten.

OpenPACE wurde vom Autor dieser Arbeit zusammen mit Frank Morgner entwickelt.

OpenPACE ist lediglich eine library, welche kryptografische Funktionen zur Verfügung stellt. Um tatsächlich mit dem Personalausweis zu kommunizieren, wurde das Kommandozeilenprogramm **pace-tool**[7] entwickelt. Die primäre Funktion von pace-tool ist die Durchführung des PACE-Protokolls mit dem nPA. Zusätzlich übernimmt es die Formatierung der Nachrichten und das Secure Messaging.

Nach der Etablierung des Secure Messaging Kanals können APDUs über die Kommandozeile eingegeben und durch diesen Kanal versand werden. Pace-tool kann also auch für die weitere Kommunikation der eID-Authentisierung (für die Durchführung von TA und CA) verwendet werden, beispielsweise unter Verwendung des Automatisierungstools expect.

Des Weiteren kann pace-tool noch über Kommandozeilenparameter für Funktionen des PIN-Managements, wie etwa das Ändern der PIN oder das Zurücksetzen des Fehlbedienungszählers, verwendet werden. Pace-tool wurde von Frank Morgner entwickelt.

Nicht alle von der GUI benötigten Funktionen aus OpenPACE sind über pace-

[6] OpenPACE ist auf Sourceforge im Quellcode veröffentlicht, siehe: `http://sourceforge.net/projects/openpace/`

[7] pace-tool ist Teil der Virtual Smartcard Architecture und wurde auf Sourceforge im Quellcode veröffentlicht, siehe `http://sourceforge.net/projects/vsmartcard/`. Der Quelltext von pace-tool befindet sich dabei im Unterverzeichnis `/ccid/src/`

tool zugänglich. Um die Funktionen zur Verarbeitung von CV-Zertifikaten nutzen zu können, muss die GUI weiterhin auf OpenPACE zugreifen. Da die GUI nicht in C programmiert werden sollte, musste ein Mechanismus zum Zugriff auf die C-Bibliothek OpenPACE aus einer Hochsprache heraus gefunden werden. Daher wurde die Bibliothek **pyPACE**[8] entwickelt, welche ein Interface zu OpenPACE für die Programmiersprache Python zur Verfügung stellt.

PyPACE ist mithilfe des Simplified Wrapper and Interface Generator (SWIG) implementiert. SWIG ist ein Werkzeug, welches auf der Grundlage einer Interface Beschreibung für in C oder C++ geschriebenen Programmcode automatisiert Bindings für eine Vielzahl an Hochsprachen erzeugen kann. Dank der Verwendung von SWIG sollte in Zukunft die Bereitstellung von Anbindungen für OpenPACE an weitere von SWIG unterstütze High-level Programmiersprachen mit verhältnismäßig geringem Aufwand möglich sein. PyPACE wurde vom Autor dieser Arbeit entwickelt.

5.2.2. Smartcardzugriff

Das Hauptziel bei der Entwicklung des Subsystems zum Smartcardzugriff war es, eine möglichst flexible Lösung zu entwickeln. Im Bereich der kontaktbehafteten Smartcards existieren bereits etablierte Standards und Softwarelösungen zum Zugriff auf eine Vielzahl unterschiedlicher Lesegeräte und Karten. Von Bedeutung sind hier vor allem die Middleware PC/SC und der CCID-Standard für USB-Smartcardleser.

Auf den NFC-fähigen Telefonen von Nokia kommt das J2ME-API *JSR 257*[9] zum Einsatz. Dieses definiert allerdings nur eine Softwareschnittstelle, die Anbindung der Hardware an dieses Interface ist hingegen nicht spezifiziert.

Viele der verfügbaren RFID-Reader mit USB-Anschluss verwenden nicht das standardisierte CCID Protokoll, sondern proprietäre Eigenentwicklungen. Zugriff auf einzelne Lesegeräte ermöglichen die beiden Bibliotheken libnfc (für Lesegeräte auf Basis des Chipsätze PN531, PN532 und PN533) und librfid (für die Lesegeräte OpenPCD, Omnikey 5121 und Omnikey 5321). Ein Nachteil der direkten Verwendung einer dieser beiden Bibliotheken ist, dass ein späterer Wechsel des verwendeten Lesegerätes potenziell einen Austausch der Bibliothek und damit einen sehr hohen Programmieraufwand zur Folge hat.

Um dieses Problem zu umgehen, sollte der Zugriff auf das verwendete Lesegerät so weit wie möglich abstrahiert werden. Hierzu wurde von Frank Morgner der IFD Handler ifdnfc entwickelt. Ein IFD-Handler ist eine Bibliothek, welche die Kommunikation mit einem oder mehreren Smartcardlesern realisiert und welche ein standardisiertes API[10] bereitstellt. Dieser IFD-Handler ermöglicht es, den Touchatag wie einen kontaktbehafteten Reader via PC/SC zu nutzen. Das Lesegerät kann dabei

[8] pyPACE ist auf Sourceforge im Quellcode veröffentlicht, siehe: http://sourceforge.net/projects/pypace/

[9] Oracle: *JSR 257: Contactless Communication API.* Verfügbar unter http://jcp.org/en/jsr/detail?id=257. Letzter Zugriff: 08.08.2010

[10] Corcoran, David und Rousseau, Ludovic: *MUSCLE PC/SC IFD Driver API.* Verfügbar unter: http://pcsclite.alioth.debian.org/ifdhandler-3/. Letzter Zugriff: 09.08.2010

mit geringem Aufwand gegen ein anderes Gerät ausgetauscht werden, solange auch für dieses ein IFD-Handler existiert.

Um den Zugriff auf angeschlossene Smartcard-Reader noch weiter zu erleichtern, wurde zusätzlich die Library OpenSC verwendet. Diese Softwarebibliothek stellt eine Reihe von Funktionen zum komfortablen Zugriff auf Smartcards zur Verfügung. Ein Nebeneffekt der Verwendung dieser Bibliothek ist, dass nicht nur PC/SC sondern auch weitere Standards, wie etwa das CT-API, unterstützt werden.

5.3. Benutzeroberfläche

Die grafische Benutzeroberfläche wurde in der Programmiersprache Python unter Verwendung der Grafikbibliothek pygtk entwickelt. Diese Technologien wurden gewählt, da sie sich sehr gut für ein rapid-prototyping Vorgehen eignen. Der Nachteil dieser Technologien ist, dass sie mehr Rechenleistung benötigen, welche in eingebetteten Geräten oftmals nicht verfügbar ist. Bei einem finalen Produkt, welches tatsächlich durch Endnutzer verwendet werden soll, sollten daher andere Technologien verwendet werden, welche eventuell einen höheren Entwicklungsaufwand benötigen, dafür aber eine bessere Performanz und damit eine flüssige Bedienung bieten. Beispielsweise könnte die Programmierung in der Sprache Vala [11] erfolgen, welche einen einfachen Zugriff auf das GObject System bietet und dank der Übersetzung in C-Code sehr performant ist.

5.3.1. Vorgehensweise

Bei der Entwicklung der grafischen Oberfläche wurde ein partizipativer Entwicklungsansatz verfolgt. Bei dieser Vorgehensweise werden potenzielle Nutzer eines Systems direkt in den Entwicklungsprozess involviert. Die Erfahrungen und Anforderungen der Endnutzer sollen dabei in Designentscheidungen einfließen. Auf diese Weise soll ein System von Anfang an so entworfen werden, dass es den Ansprüchen der späteren Endnutzer gerecht wird.

Ein häufig genannter Vorteil dieser Entwicklungsmethode ist, dass durch die Teilnahme am Entwicklungsprozess ein Gefühl der Teilhabe beim Nutzer entsteht, welches wiederum zu einer erhöhten Akzeptanz des finalen Systems beiträgt (vergleiche beispielsweise [Kuj03]).

Die Methode des partizipativen Designs ist jedoch nicht unumstritten. Einer der Kritikpunkte, die häufig gegen diese Methode eingebracht werden ist, dass Nutzer keine Designer sind, also nicht über das notwendige Hintergrundwissen zur Gestaltung gebrauchstauglicher Systeme verfügen. Weitere Kritikpunkte sind erhöhte Kosten und die Ablehnung des Systems durch Nutzer, die nicht in die Entwicklung einbezogen wurden (vergleiche [IO84]).

[11] The GNOME Project: *Vala – Compiler for the GObject type system.* Verfügbar unter: `http://live.gnome.org/Vala`. Letzter Zugriff: 09.08.2010

In der vorliegenden Arbeit wurde der Prozess der partizipativen Entwicklung folgendermaßen umgesetzt: Die Benutzeroberfläche wurde iterativ entwickelt. Nach jeder Veränderung wurde sie durch einige Tester begutachtet. Diese wurden zum einen nach ihrem allgemeinen Eindruck, zum anderen zu den konkret vorgenommenen Veränderungen befragt. Abhängig vom Urteil der Tester wurden die Veränderungen beibehalten, verworfen oder angepasst. Meist wurden zwei bis drei Tester befragt. Ergab sich aus diesen Befragungen kein klares Meinungsbild, so wurden weitere Tester hinzugezogen. Maximal wurden fünf Personen befragt.

Bei den Testern handelte es sich größtenteils um junge Personen aus dem Umfeld des Autors dieser Arbeit. Eine formale Auswahl der Tester fand nicht statt. Auf die Einbeziehung älterer Tester, wie in Kapitel 4, wurde aus Zeitgründen verzichtet. Die Tests erfolgten informell und dauerten meist nur wenige Minuten.

5.3.2. eID-Dialog

Das primäre Ziel bei der Entwicklung der grafischen Nutzeroberfläche war die Abbildung des eID-Authentisierungsvorgangs. Notwendig war also die Entwicklung von Dialogen zum Anzeigen von Informationen über den Dienstanbieter, zur Auswahl der Zugriffsrechte und zur PIN-Eingabe (vergleiche Unterabschnitt 3.2.5).

Diese drei Fenster werden dem Benutzer nacheinander angezeigt. Die Dialogführung entspricht dem Wizard Muster, der Nutzer kann vor und zurück navigieren, weitere Verzweigungen sind nicht möglich.

Jeder erfolgreiche Durchlauf des Dialogs wird in einer Logdatei protokolliert. Dabei werden das Datum, die Uhrzeit, das Terminalzertifikat und die Certificate Description abgespeichert. Diese Logdatei soll die Möglichkeit der Kontrolle der getätigten Authentisierungsvorgänge geben. Ein zusätzliches Programm zur übersichtlichen Anzeige der Logeinträge wäre wünschenswert, konnte jedoch nicht rechtzeitig fertiggestellt werden.

Dienstanbieterinformationen: Im ersten Fenster des eID-Dialogs werden dem Nutzer Informationen aus dem Terminalzertifikat und der Certificate Description angezeigt. Das Fenster besteht aus einem Titel, den Dienstanbieterinformationen (versehen mit einer vertikalen Scrollbar) und den Buttons zur Navigation. In diesem Fenster musste eine Vielzahl von Informationen untergebracht werden. Jedes Datum wird durch einen (fett gedruckten) Titel und das darunter eingerückt abgedruckte Datum selbst dargestellt.

Als problematisch erwies sich das Datum `termsOfUsage`. Die Gui unterstützt lediglich den Umgang mit UTF-8 formatierten Terms of Usage Informationen (OID: `id-plainFormat`, vergleiche Abschnitt 3.3). Der Umgang mit HTML- oder PDF-formatierten Daten könnte zu einem späteren Zeitpunkt über Plugins oder die Verwendung von externen Anzeigeprogrammen nachgerüstet werden.

Leider ist keine Spezifikation zur Strukturierung der in diesem Feld enthaltenen Informationen verfügbar, sodass eine weitere Aufbereitung zur besseren Darstellung

Abbildung 5.4.: Verschiedene Entwicklungsstufen des Dialogs zur Anzeige der Dienstanbieterinformationen. Links eine ältere, rechts die finale Variante

der Information in der GUI nicht ohne Weiteres möglich ist. In den Certificate Descriptions von Teilnehmern des offenen Anwendungstests des neuen Personalausweises waren die einzelnen Einträge des `termsOfUsage` Feldes durch Kommata getrennt. Diese werden durch Zeilenumbrüche ersetzt, damit die Informationen ohne vertikales scrollen darstellbar sind. Eine weitere Aufbereitung, wie etwa das Umwandeln von im Text enthaltenen URLs in anklickbare Links, findet nicht statt.

Die befragten Tester empfanden den Dialog zumeist als unnötig. „Das klicken doch eh alle weg, ohne es zu lesen“, kommentierte ein Tester. Der Dialog wurde dennoch beibehalten, da nicht sicher war, ob die Anzeige der Informationen verpflichtend ist oder nicht. In PAuswG § 18, Absatz 4 heißt es dazu: „Vor Eingabe der Geheimnummer durch den Personalausweisinhaber müssen insbesondere die folgenden Angaben aus dem Berechtigungszertifikat zur Anzeige übermittelt werden: [...]“. Sollte die Anzeige der Informationen nicht verpflichtend sein, so könnte das Fenster grundsätzlich weggelassen und lediglich auf Wunsch des Nutzers angezeigt werden. Hierzu könnte beispielsweise ein zusätzlicher Button zur Anzeige der Dienstanbieterinformationen in den Dialog zur Auswahl der Zugriffsrechte eingebaut werden.

Zugriffsrechte: Das nächste Fenster zeigt dem Nutzer die im CHAT des Terminalzertifikats codierten Zugriffsrechte des Dienstanbieters an und bietet ihm die Möglichkeit, diese Berechtigungen weiter einzuschränken. Hierzu werden die einzelnen Berechtigungen in einer Liste angezeigt. Jeder Eintrag verfügt über eine Checkbox, welche zur Abwahl der Berechtigungen verwendet werden kann. Initial sind alle Berechtigungen angewählt.

Im Gespräch mit den Testern stellte sich heraus, dass ein Problem in diesem Fenster die Bezeichnung der einzelnen Berechtigungen war. Vor allem die speziellen Funk-

Abbildung 5.5.: Verschiedene Entwicklungsstufen des Dialogs zur Auswahl der Zugriffsrechte. Links eine ältere, rechts die finale Variante

tionen Altersverifikation, Restricted Identification und Community-ID Verification bedurften der Erklärung. Weiterhin war der Unterschied zwischen dem Zugriff auf die Datengruppen für Geburtsdatum und Wohnort-ID und die zugehörigen Verifikationsfunktionen für einige Tester nur schwer verständlich. Diesem Problem wurde durch zwei Maßnahmen begegnet:

1. Umbenennung der Berechtigungen: Im Gespräch mit den Testern wurde versucht, allgemein verständliche Bezeichnungen für die einzelnen Datengruppen und Funktionen zu finden. So wurde beispielsweise die Restricted Identification in „Pseudonymfunktion“ umbenannt.

2. Hilfefunktionen für schwer verständliche Berechtigungen: Für eine Reihe von Datengruppen und Funktionen wurden kurze Hilfstexte verfasst. Falls für ein Zugriffsrecht ein Hilfstext existiert, so wird ein Info-Icon neben der entsprechenden Berechtigung angezeigt. Bei einem Klick auf dieses Icon erscheint ein Popup-Fenster, welches den Hilfetext anzeigt.

PIN-Eingabe: Das letzte Fenster dient der Eingabe der PIN. Als Grundlage dieses Fensters diente das PIN-Eingabe-Programm für den Handhabbarkeitstest aus Kapitel 4. Zunächst wurden die im Test gefundenen Gestaltungsmängel, wie etwa die zu kleine Schrift behoben. Anschließend wurde eine Reihe von Verbesserungen der Oberfläche mit dem Ziel der besseren Gebrauchstauglichkeit vorgenommen.

Um dem Benutzer besseres Feedback zu liefern, ob sich die Karte im Funkfeld des NFC-Chips befindet oder nicht, wurde ein Statuslabel eingeführt. Die Farbe des Textes fungiert dabei als zusätzlicher visueller Hinweis: Befindet sich der Ausweis im Feld, so ist der Text grün gefärbt, ansonsten rot. Nach Tests mit einigen Anwendern wurde der Text zusätzlich um Icons ergänzt.

Abbildung 5.6.: Verschiedene Entwicklungsstufen des Dialogs zur PIN-Eingabe. Links eine ältere, rechts die finale Variante

Um die Verfügbarkeit des Ausweises zu überprüfen, findet ein konstantes Polling statt. Das Programm fragt dabei einmal pro Sekunde das Answer To Reset (ATR) der Karte ab. Ist die Abfrage erfolgreich, so befindet sich die Karte im Feld. Aus Gründen der Ressourcensparsamkeit wäre ein Push-Mechanismus wünschenswert, welcher Programme über Statusänderungen der Karte informiert. Ein solcher Mechanismus sollte sich prinzipiell mit einem Aufruf von `opensc-tool -w` realisieren lassen. Dieses Kommando wartet bis eine neue Karte vom Lesegerät erkannt wird. Leider führte die Verwendung dieses Kommandos auf dem OpenMoko lediglich zu einer Fehlermeldung, sodass der Polling-Mechanismus beibehalten wurde.

Die Anzeige der eingegeben Ziffern wurde ebenfalls überarbeitet. Zum einen wurde die Anzeige linksbündig formatiert anstatt zentriert wie zuvor. Der Grund dafür war, dass der Text bei jeder Veränderung neu zentriert werden musste, wodurch ein unruhiger Eindruck entstand. Zum anderen wurden die eingegeben Ziffern nicht sofort durch Stern-Zeichen ersetzt, sondern zunächst 750 Millisekunden lang im Klartext dargestellt. Durch diese Änderung sollten Fehleingaben vermieden werden, indem dem Nutzer zusätzliches Feedback zu seiner Eingabe gegeben wird.

Um fehlerhaften PIN-Eingaben vorzubeugen, wurde außerdem der „Ok“ Button erst aktiviert, wenn zuvor bereits sechs Ziffern eingegeben wurden. Somit kann eine Eingabe nicht mehr aus Versehen zu früh bestätigt werden.

5.3.3. Einrichtungsassistent

Zusätzlich zu der Benutzeroberfläche für die eID-Authentisierung war noch ein Konfigurationsassistent eingeplant. Vorgesehen hierfür waren das optionale Abspeichern der CAN. Die gespeicherte CAN könnte nach zwei Fehleingaben dazu genutzt werden, die supspendierte PIN automatisch wieder freizuschalten. Weitere Aufgaben des

Assistenten wären das Setzen einer neuen eID-PIN und die Konfiguration der Logdatei.

Leider konnte dieser Assistent nicht rechtzeitig zur in Kapitel 6 beschriebenen Evaluation der Oberfläche fertiggestellt werden. Aus diesem Grund wurde das Fenster zum Ändern der eID-PIN aus dem Assistenten herausgelöst und als eigenständige Anwendung zur Verfügung gestellt.

Zum Ändern der PIN ist zunächst die Eingabe der alten PIN und anschließend die zweifache Eingabe der gewünschten neuen PIN notwendig. Die Wiederholung der neuen PIN soll dabei verhindern, dass - etwa durch einen Tippfehler - eine falsche PIN gesetzt wird, ohne dass der Benutzer es bemerkt. Direkt nach der Eingabe der alten PIN wird im Hintergrund ein PACE Durchlauf mit dem Personalausweis durchgeführt. Dadurch wird überprüft, ob die alte PIN vom Nutzer korrekt eingegeben wurde. Der Vorteil dieses Vorgehens ist, dass der Nutzer im Falle einer fehlerhaft eingegeben PIN nicht den kompletten Dialog und damit die dreimalige PIN-Eingabe erneut durchlaufen muss. Der Nachteil ist, dass mehrfache Fehleingaben der alten PIN zu einer Sperrung der eID-PIN führen können. Außerdem führt der PACE Durchlauf zu einer Verzögerung zwischen der Eingabe der alten und der ersten Eingabe der neuen PIN.

Eine alternative Gestaltung des Dialoges hätte beispielsweise folgendermaßen aussehen können: Zunächst würden alle drei PIN-Eingaben des Nutzers durchgeführt. Stimmen die beiden Eingaben der neuen PIN überein, so wird eine PIN-Änderung mit dem Ausweis angestoßen. Schlägt diese fehl, so muss lediglich die alte PIN erneut eingegeben werden, die Ziel-PIN bleibt erhalten. Grundsätzlich sollten beide Verhaltensweisen implementiert und anschließend mit potenziellen Nutzern getestet werden. Dies war jedoch aus den bereits erwähnten Zeitgründen nicht möglich.

6. Heuristische Evaluation

Um das System zur mobilen Nutzung der eID-Authentisierung zu bewerten, wurde die Methode der *heuristischen Evaluation* eingesetzt. Diese Methode wurde ursprünglich von Jakob Nielsen in [NM90] vorgestellt und später in [Nie94] weiterentwickelt. Zwei Prüfgegenstände wurden evaluiert: eine Testversion des Bürgerclients – der offiziellen Anwendungssoftware für den neuen Personalausweis – und die in Kapitel 5 beschriebene mobile Lösung. Der Bürgerclient wurde als Referenzimplementierung in die Evaluation mit aufgenommen, um einen Vergleichswert für die Ergebnisse der Evaluation zu schaffen. Das Ziel war es zu untersuchen, ob die Verwendung von Methoden des Usability-Engineerings bei der Entwicklung des mobilen Systems tatsächlich zu einem gebrauchstauglicheren System geführt haben als ein nach herkömmlicher Vorgehensweise entwickeltes Programm.

Bei der heuristischen Evaluation wird ein Prüfgegenstand von mehreren Evaluatoren auf der Grundlage von etablierten Heuristiken beurteilt. Es handelt sich um eine Expertenevaluation, die Evaluatoren müssen also über Kenntnisse im Bereich des Usability-Engineerings verfügen. Aufgrund dieser Expertise genügt es meist, die Evaluation mit einer geringen Anzahl von Evaluatoren (typischerweise drei bis fünf) durchzuführen. Dadurch ist diese Methode mit sehr viel weniger Aufwand und Kosten verbunden als etwa ein Nutzertest, bei welchem eine vergleichsweise hohe Anzahl an Versuchspersonen benötigt wird. Jakob Nielsen entwickelte die heuristische Evaluation als Teil einer Sammlung von Methoden unter dem Namen *Discount Usability-Engineering*. Das Ziel dabei war es, preisgünstig durchzuführende Methoden vorzustellen, um dem in der Praxis häufig geäußerten Argument zu begegnen, dass das Sicherstellen der Gebrauchstauglichkeit eines Produktes sehr teuer sei und sich deshalb nur in Ausnahmefällen lohne.

Die Heuristiken dienen der Strukturierung der Evaluation und der Kategorisierung der gefundenen Fehler. Es handelt sich dabei nicht um konkrete Gestaltungsvorschriften, sondern um mehr oder weniger abstrakte Prinzipien gebrauchstauglicher Gestaltung. Es existieren zahlreiche Vorschläge für geeignete Heuristiken. Am bekanntesten sind dabei wohl die von Nielsen in [Nie94] vorgestellten Heuristiken. Ein weiteres Beispiel sind die Cognitive Engineering Principles von Jill Gerhardt-Powals [GP96].

Generell empfiehlt es sich, Heuristiken einzusetzen, die auf den speziellen Anwendungsbereich des zu untersuchenden Prüfgegenstandes angepasst sind. Leider konnten im Vorfeld der Evaluation keine spezialisierten Heuristiken für Authentisierungsmethoden gefunden werden. Für mobile Endgeräte existieren hingegen eine Reihe von Heuristiken. Um die Vergleichbarkeit zwischen den beiden Evaluationsdurchgängen sicherzustellen, wurden jedoch für beide Prüfgegenstände dieselben Heuristiken

verwendet. Die Wahl fiel dabei auf die zehn Heuristiken von Nielsen, da diese weit verbreitet und in der Praxis erprobt sind.

Die Heuristiken (in der Übersetzung nach [SB06], Erläuterungen durch den Autor dieser Arbeit) lauten:

H1 Sichtbarkeit des Systemzustands: Der Nutzer sollte durch geeignetes Feedback über den aktuellen Systemzustand informiert werden.

H2 Übereinstimmung zwischen System und realer Welt: Das System sollte durch Bezeichnungen und Analogien, die dem Nutzer vertraut sind, ein geeignetes mentales Modell des Systems erzeugen.

H3 Benutzerkontrolle und Freiheit: Der Nutzer sollte frei durch die Nutzeroberfläche navigieren und seine Aktionen jederzeit einfach wieder rückgätngig machen können.

H4 Konsistenz und Standards: Die Konventionen der Plattform sollten berücksichtigt werden.

H5 Fehler vermeiden: Wo möglich sollten Bedienfehler durch geeignete Maßnahmen vermieden werden.

H6 Erkennen vor Erinnern: Die Gedächtnislast des Benutzers sollte so gering wie möglich gehalten werden. Dies kann unter anderem durch die Verwendung geeigneter Symbole erreicht werden.

H7 Flexibilität und effiziente Nutzung: Erfahrenen Nutzern sollten effiziente Bedienmöglichkeiten – wie etwa Tastaturkürzel oder Makros – zur Verfügung stehen, die aber unerfahrene Nutzer nicht verwirren sollten.

H8 Ästhetisches und minimalistisches Design: Dialoge sollten möglichst nur die in der jeweiligen Situation benötigten Informationen enthalten.

H9 Unterstützung beim Erkennen, Verstehen und Bearbeiten von Fehlern: Fehlermeldungen sollten verständlich und konstruktiv sein.

H10 Hilfe und Dokumentation: Dem Nutzer sollte eine verständliche, durchsuchbare Hilfe zur Verfügung gestellt werden, die sich auf die eigentliche Aufgabe des Nutzers fokussiert.

Auch bei der Bewertung der gefundenen Probleme wurde Nielsens ursprüngliche Methode verwendet. Sie sieht die Bewertung der einzelnen Probleme auf einer fünfstufigen Skala vor, wobei eine Bewertung von Null „Kein Usability-Problem“ und eine Bewertung von Vier „Usability Katastrophe“ bedeutet. In die Bewertung der einzelnen Probleme fließen dabei die folgenden vier Faktoren ein (ebenfalls nach [SB06]):

1. *Frequenz* des Auftretens (einmalig bis laufend)
2. *Einfluss* auf die Arbeitsabläufe (Nichtigkeit bis schwere Störung)
3. *Persistenz* des Auftretens (zufälliges bis regelmäßiges Auftreten)
4. *Markteinfluss* (Ein insgesamt ungünstiger Eindruck sollte vermieden werden.)

6.1. Vorgehen

Drei Evaluatoren konnten für die heuristische Evaluation gewonnen werden. Alle drei waren Studenten, die Vorlesungen im Bereich Ingenieurpsychologie besucht hatten. Zwei Evaluatoren waren Informatikstudenten mit Nebenfach Psychologie. Einer von ihnen verfügte über seine Ausbildung im Usabilitybereich hinaus zusätzlich noch über Domänenwissen zum elektronischen Personalausweis. Die dritte Evaluatorin war eine Diplom-Psychologin.

Die Evaluation wurde jeweils in Einzelsitzungen mit einem Evaluator durchgeführt. Zu Beginn gab es eine Einführung in die heuristische Evaluation durch den Autor. Diese sollte für die Evaluatoren als Wiederholung der Methodik dienen und eine gemeinsame Basis für die Evaluation sicherstellen. Am Ende der Einführung wurde eine Übung aus [Nie94] durchgeführt, bei welcher das Mock-Up eines Wetterdienstes evaluiert wurde. Diese Übung diente vor allem dazu, Beispiele für Verstöße gegen die einzelnen Heuristiken aufzuzeigen.

Die Durchführung der eigentlichen Evaluation orientierte sich an den Empfehlungen aus [Nie05] und [SB06]. Für beide Prüfgegenstände wurden dabei jeweils drei Phasen durchlaufen:

Vorstellung des Prüfgegenstandes: Der Prüfgegenstand wurde zunächst vom Versuchsleiter genau beschrieben. Hierbei lag der Schwerpunkt auf der Erläuterung der vorhandenen Funktionalität und der Bedienung der jeweiligen Software. Technische Details wurden weitestgehend außen vor gelassen.

Evaluation: Im Anschluss führten die Evaluatoren die eigentliche Prüfung durch. Sie wurden dazu aufgefordert, die gesamte Nutzeroberfläche mehrfach durchzugehen. Zur Dokumentation der gefundenen Probleme lagen vorgefertigte Berichtsbogen bereit (siehe Anhang D). Zusätzlich standen den Evaluatoren Merkblätter mit Erläuterungen zu den Heuristiken und Bewertungskriterien in ausgedruckter Form zur Verfügung (siehe ebenfalls Anhang D). Bei Fragen oder Problemen gab der Versuchsleiter Hilfestellung.

Vorstellung der Ergebnisse: Nach der Evaluation präsentierten die Evaluatoren dem Versuchsleiter ihre Ergebnisse. Sie zeigten ihm die Fundstellen, beschrieben den Verstoß gegen die entsprechende Heuristik und begründeten ihre Bewertung des jeweiligen Problems.

Nach der Auswertung der einzelnen Bewertungen der Evaluatoren wurden diese zu einer kombinierten Liste zusammengefasst. Hierbei wurden zunächst die einzelnen Listen vereinheitlicht – etwa die Bezeichnung der Fundorte – und die stichpunktartigen Beschreibungen teilweise ausgearbeitet. Einige Nennungen wurden zusammengefasst und einzelne Einträge wurden weggelassen. So bemängelte ein Evaluator etwa, dass auf dem PIN-Brief zwar korrekt vermerkt sei, dass es sich hierbei um eine vorläufige PIN handele, die PIN selbst aber mit der Überschrift „PIN (Bitte hier frei rubbeln) “versehen sei. Der Evaluator kritisierte nun, dass einem Ausweisinhaber, der den Brief nicht sorgfältig lese, nicht klar werde, dass die PIN zur Verwendung der eID-Funktion geändert werden müsse. Diese Einschätzung ist zwar nachvollziehbar, bezieht sich aber nicht auf den eigentlichen Prüfgegenstand. Daher wurde sie nicht in die konsolidierte Liste aufgenommen.

Das Zusammenstellen der kombinierten Liste wurde vom Autor dieser Arbeit durchgeführt. Grundsätzlich wäre es wünschenswert gewesen, die kombinierte Liste mit allen Evaluatoren gemeinsam abzusprechen, um sich auf eine gemeinsame Bewertung der Probleme zu einigen. Ein solches gemeinsames Treffen konnte jedoch aus Zeitgründen nicht durchgeführt werden. Wurde ein Verstoß von mehreren Evaluatoren notiert, wurde daher als Bewertung das arithmetische Mittel der Einzelbewertungen verwendet.

6.1.1. Bürgerclient

Um die eID-Authentisierung mit dem Bürgerclient zu erproben, standen den Evaluatoren die vom Fraunhofer Institut FOKUS (Fraunhofer Institut für Offene Kommunikationssysteme) erarbeiteten Testszenarien zur Verfügung. Dabei handelte es sich um eine Webseite mit vier Einsatzszenarien für den nPA:

- eine Online-Videothek mit Altersverifikation zum Zugriff auf ab 18 Jahren freigegebene Filme
- ein Blog, bei welchem man sich per Restricted Identification anmelden kann, um die Kommentarfunktion zu nutzen
- eine Website zur Registrierung als Wahlhelfer
- eine Bankseite zur Kontoeröffnung

Durch die Verwendung authentischer Nutzungsszenarien innerhalb der heuristischen Evaluation sollte der Nutzungskontext der eID-Funktion in den Fokus der Evaluation gerückt werden. Einer der häufig geäußerten Kritikpunkte an der Methode der heuristischen Evaluation ist nämlich, dass sie den Kontext einer Software außer Acht lässt und stattdessen häufig lediglich minder schwere Probleme in der Gestaltung der grafischen Oberfläche aufdeckt (vergleiche z.B. [DR93]). Um diesem Problem zu begegnen, kann die heuristische Evaluation mit Elementen des *Cognitive*

Walkthrough – wie etwa Nutzungsszenarien – kombiniert werden. Teilweise wird dieses Verfahren auch als *heuristic walkthrough* bezeichnet (beispielsweise in [PHVS04]), auch wenn Nielsen selbst bereits die Verwendung von Szenarien in der heuristischen Evaluation empfahl (vergleiche [Nie05]).

Zur Evaluation des Bürgerclients stand den Evaluatoren ein Arbeitsplatz, bestehend aus einem PC samt Lesegerät, sowie jeweils ein Testausweis zur Verfügung. Die Testausweise wurden den Evaluatoren im Auslieferungszustand übergeben, also mit gesetzter Transport-PIN, sowie PIN- und PUK-Brief. Es wurden zwei verschiedene Modelle von Lesegeräten verwendet: der Basisleser Omnikey 6321 von HID und eine Vorabversion des Komfortlesers CyberJack RFID komfort von ReinerSCT. Durch den Einsatz unterschiedlicher Lesegeräte sollten die verschiedenen Formen der Interaktion bei der PIN-Eingabe untersucht werden (vergleiche Unterabschnitt 3.2.5).

Da die Ausweise im Auslieferungszustand übergeben wurden, bestand die erste Aufgabe der Evaluatoren darin, eine eID-PIN zu setzen, da mit der Transport-PIN keine Durchführung der eID-Authentisierung möglich ist. Anschließend konnten sie mithilfe der oben beschriebenen Szenarien die Authentisierung mehrfach durchspielen.

Als zusätzliche Dokumentation standen den Evaluatoren noch zwei PDF-Dateien mit Informationen zum Bürgerclient zur Verfügung.

6.1.2. OpenMoko

Bei der Evaluation der mobilen Lösung war die Verwendung der Szenarien nicht möglich, da die hierzu notwendige Kommunikation mit dem Dienstanbieter über die eCard-API fehlte. Auf die Entwicklung lokaler Testszenarien, welche nicht auf die eCard-API angewiesen wären, wurde aus Zeitgründen verzichtet. Statt dessen wurde die Authentisierungsoberfläche mit einem statischen CV-Zertifikat und der entsprechenden Certificate Description gestartet. Auf diese Weise fiel die Interaktion mit dem eigentlichen Dienst, also das Anstoßen des Authentisierungsvorgangs über die Webseite des Dienstanbieters, weg. Durch die Verwendung authentischer Daten entsprach der Vorgang dennoch einem realen Authentisierungsprozess.

Kurz vor Durchführung der Evaluation traten bei dem im OpenMoko eingebauten Lesegerät gehäufte Paketverluste und Übertragungsfehler bei der Kommunikation mit dem Personalausweis auf. Mögliche Ursachen für diese Fehler könnten Kabelbrüche oder Defekte an den Lötstellen gewesen sein. Da nicht genug Zeit zur Verfügung stand den Fehler zu beseitigen, wurden für die Evaluation Lesegeräte extern an die Telefone angeschlossen.

6.2. GUI des Bürgerclients

Für das Verständnis der Evaluationsergebnisse ist eine kurze Beschreibung der grafischen Nutzeroberfläche des Bürgerclients notwendig. Alle Beschreibungen beziehen sich dabei auf die Windows Version des Bürgerclients.

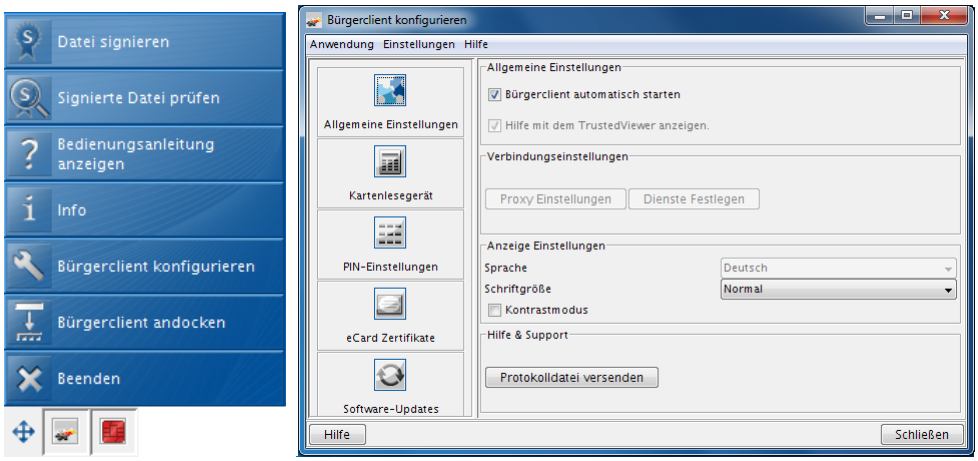

Abbildung 6.1.: GUI des Bürgerclients: links das Widget mit geöffnetem Kontextmenü, rechts das Konfigurationsfenster

Nach der Installation ist der Bürgerclient standardmäßig immer aktiv. Er kann dabei entweder als kleines Fenster, welches nur aus wenigen Icons besteht und immer im Vordergrund ist (im Folgenden als *Widget* bezeichnet) oder aber als Reihe von Symbolen in der Taskleiste neben der Systemuhr, angezeigt werden. Dabei verfügt das Widget über ein Icon zum Öffnen des Kontextmenüs und jeweils ein Icon pro angeschlossenem Lesegerät. Ein zusätzliches Icon in Form eines Pfeilsymbols dient dem Verschieben des Widgets.

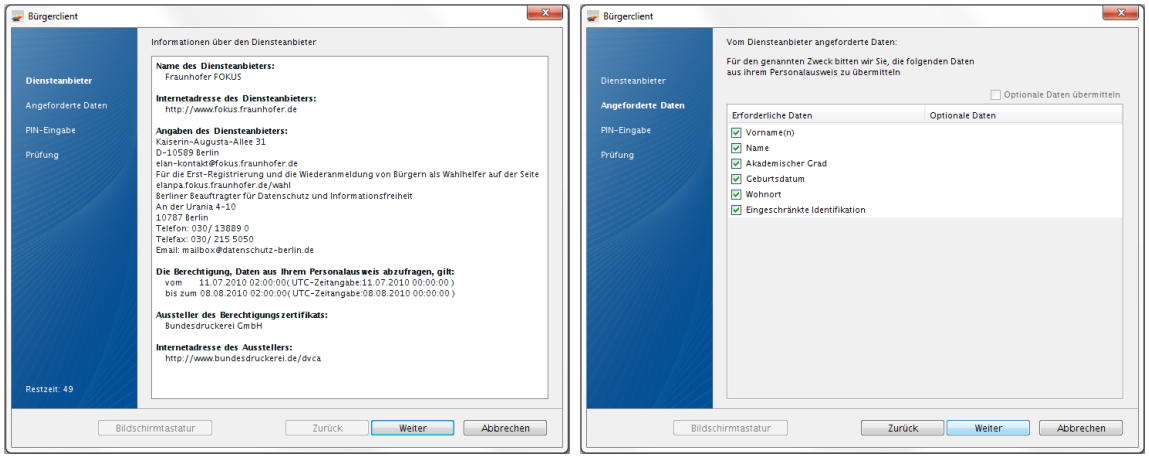

Abbildung 6.2.: eID-Authentisierung mit dem Bürgerclient

Über das Kontextmenü kann der Konfigurationsdialog aufgerufen werden. Dieser beinhaltet fünf Kategorien von Einstellungsoptionen: allgemeine Einstellungen,

PIN-Einstellungen, sowie Einstellungen zu Lesegeräten, Zertifikaten und Software-Updates. Die Evaluatoren mussten hier zumindest die PIN-Einstellungen verwenden, um eine neue eID-PIN zu setzen.

Surft man eine Webseite an, die die eID-Authentisierung unterstützt, so öffnet sich der eID-Dialog. Dieser entspricht den in Abschnitt 3.3 beschriebenen drei Authentisierungsphasen. Abbildung 6.2 zeigt die Fenster zu Anzeige der Dienstanbieterinformationen und zur Auswahl der Zugriffsrechte.

6.3. Ergebnisse

Im Folgenden werden sowohl die Einzelergebnisse der Evaluatoren als auch die aus allen drei Einzellisten zusammengeführte konsolidierte Liste der Usability-Probleme besprochen. Alle Listen befinden sich darüber hinaus auch als Excel-Tabellen auf der beiliegenden CD im Ordner `daten/evaluation`.

6.3.1. Ergebnisse der einzelnen Evaluatoren

Die Evaluatoren fanden in beiden untersuchten Systemen eine Reihe von Verstößen gegen Nielsens Heuristiken. Der erste Evaluator fand 29 Probleme im Bürgerclient und neun im OpenMoko, der zweite 34 beim Bürgerclient und neun beim OpenMoko und der dritte Evaluator fand zwölf Probleme im Bürgerclient und fünf im OpenMoko.

Abbildung 6.3 zeigt die Verteilung der gefundenen Probleme auf die zehn Heuristiken. Auffällig ist dabei vor allem, dass es insgesamt nur eine einzige Nennung für die Heuristik H6 – Erkennen vor Erinnern – gab. Diese Nennung bezog sich auf die Verwendung des Komfortlesers mit dem Bürgerclient und kritisierte, dass der Benutzer bei der Anzeige der Zugriffsrechte auf dem Lesegerät eventuell schon vergessen haben könnte, welche Rechte er zuvor am Bildschirm ausgewählt hatte.

Am häufigsten wurden beim Bürgerclient Verstöße gegen die Heuristik H4 – Konsistenz und Standards – genannt (15 Nennungen). Dabei kritisierten die Evaluatoren eins und zwei vor allem eine Reihe minder schwerer Verstöße (siehe Tabelle 6.1 für die durchschnittlichen Bewertungen), wie etwa die ungewöhnliche Formatierung der Datumsangaben als UTC oder die ungewöhnliche Gestaltung des Widgets.

Am zweithäufigsten wurden Verstöße gegen die Heuristik H5 – Fehler vermeiden – (10 Nennungen) genannt, gefolgt von H8 – ästhetisches und minimalistisches Design – (9 Nennungen). Als Verstöße gegen Heuristik H5 wurden beispielsweise das Zeitlimit des Authentisierungsdialoges und die randomisierte Positionierung der Ziffern bei der PIN-Eingabe über die Bildschirmtastatur genannt. Ein Beispiel für einen Verstoß gegen die Heuristik H8 ist die prominente Platzierung des „Info“ Buttons im Kontextmenü des Widgets. Dieser Button führt lediglich zum Aufruf eines Fensters mit dem Logo des Bürgerclients.

Beim OpenMoko wurden die meisten Verstöße der Heuristik H1 – Sichtbarkeit des Systemzustands – zugeordnet (6 Nennungen). So wurde beispielsweise kritisiert, dass

bei einer Fehleingabe der PIN nicht die verbleibende Anzahl von Eingabeversuchen angezeigt wird. An zweiter Stelle folgten Verstöße gegen Heuristik H8, etwa die große Schriftart der Titel der einzelnen Dialogfenster, und H10 – Hilfe und Dokumentation. Als Verstoß gegen letztere Heuristik wurde zum Beispiel das Fehlen eines Hilfetextes für das Zugriffsrecht – „Gemeindeschlüssel “ moniert.

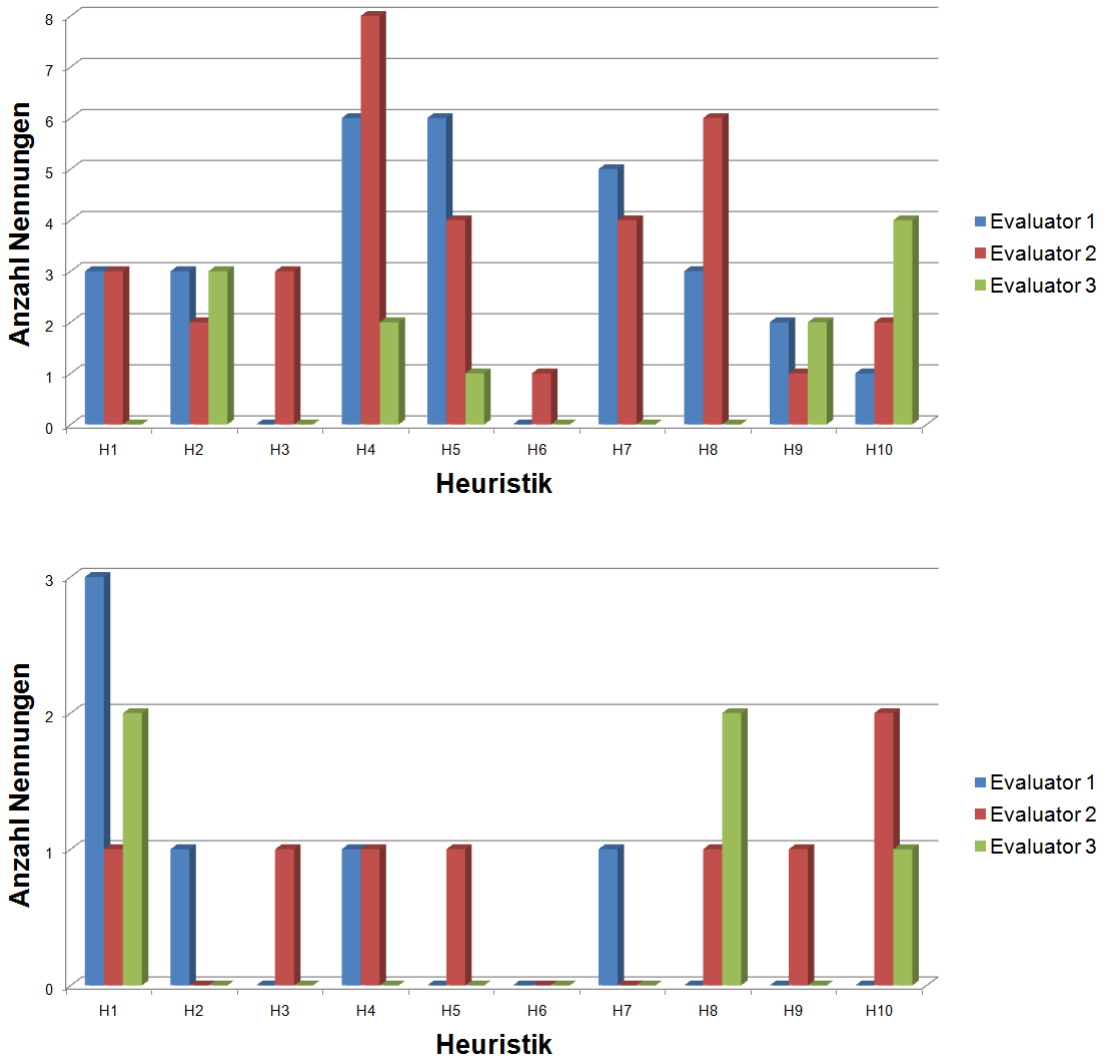

Abbildung 6.3.: Anzahl der Nennungen von Verstößen gegen Nielsens Heuristiken im Bürgerclient (oberes Diagramm) und beim OpenMoko (unteres Diagramm)

Im Durchschnitt vergab der erste Evaluator eine Bewertung von 1.72 für die Probleme im Bürgerclient ($SD = 0.87$), der zweite eine Bewertung von 1.86 ($SD = 0.97$) und der dritte von 2.25 ($SD = 0.97$). Beim OpenMoko betrugen die durchschnittlichen Bewertungen 2.33 ($SD = 0.82$), 2.25 ($SD = 0.97$) und 1.6 ($SD = 0.49$). Schlüsselt

man die durchschnittlichen Bewertungen nach den zugehörigen Heuristiken auf, so erhält man die in Tabelle 6.1 abgebildeten Werte.

	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Evaluator 1 (Bürgerclient)	2	1.33	-	1.33	2	-	1.8	1	1.33	2
Evaluator 2 (Bürgerclient)	1.67	1	3.67	1.75	3	2	1.75	1.67	1	2.5
Evaluator 3 (Bürgerclient)	-	1.33	-	3	3	-	-	-	2.5	2.25
Evaluator 1 (OpenMoko)	2.67	1	-	3	-	-	2	-	-	-
Evaluator 2 (OpenMoko)	2	-	2	3	3	-	-	1	4	1.5
Evaluator 3 (OpenMoko)	1.5	-	-	-	-	-	-	1.5	-	2

Tabelle 6.1.: Durchschnittliche Bewertungen der einzelnen Evaluatoren nach Heuristik

Die einzelnen Evaluatoren fanden an unterschiedlichen Orten in den Anwendungen Fehler. So fand beispielsweise der erste Evaluator die meisten Fehler beim Bürgerclient im Konfigurationsdialog. Insgesamt entdeckte er hier 15 Verstöße gegen die Heuristiken, davon fünf im Unterdialog „PIN-Einstellungen“. Beim zweiten Evaluator waren hingegen die genannten Probleme recht gleichmäßig über die gesamte Anwendung verteilt. Er fand neun Fehler im Hauptfenster des Bürgerclients, neun im Konfigurationsdialog, sechs bei der PIN-Eingabe, fünf bei der Anzeige der Dienstanbieterinformationen und drei bei der Anzeige der Zugriffsrechte. Eine weitere Nennung bezog sich auf den Authentisierungsdialog im Allgemeinen. Beim OpenMoko bezogen sich die meisten Anmerkungen auf die PIN-Eingabe, wobei der erste Evaluator sich hauptsächlich auf den Dialog zur PIN-Änderung fokussierte (vier Nennungen), der zweite hingegen auf die Eingabe der PIN im eID-Dialog (ebenfalls vier Nennungen).

6.3.2. Konsolidierte Ergebnisse

Die aus den Einzelbewertungen konsolidierte Liste der Usability-Probleme umfasste 54 Einträge für den Bürgerclient und 18 Einträge für das OpenMoko. Die einzelnen Evaluatoren bemängelten recht unterschiedliche Probleme, sodass es nur wenige Überschneidungen zwischen den Einzellisten gab. Lediglich sieben Probleme des Bürgerclients und vier des OpenMokos wurden von mehr als einem Evaluator genannt.

Beim Bürgerclient bemängelten alle drei Evaluatoren den bereits erwähnten „Info“ Button im Kontextmenü. Zwei Evaluatoren bemängelten das langwierige Öffnen des

Konfigurationsdialogs ohne jede Rückmeldung (wie z.B. eine Sanduhr). Ebenfalls von zwei Evaluatoren wurde bemängelt, dass beim Ändern der PIN das korrekte Lesegerät manuell ausgewählt werden muss, falls mehrere Geräte angeschlossen sind. Hier hätten sich beide Evaluatoren gewünscht, dass automatisch das Lesegerät mit dem Personalausweis ausgewählt werden würde.

Beim OpenMoko wurden beispielsweise die fehlende Anzeige der Anzahl der verbleibenden PIN-Eingabeversuche und das kurze Aufblinken des Desktops beim Fensterwechsel von mehreren Evaluatoren bemängelt.

Die meisten der beim OpenMoko gefundenen Probleme lassen sich der Heuristik H1 zuordnen, die übrigen Nennungen entfallen recht gleichverteilt auf die anderen Heuristiken. Beim Bürgerclient traten die Probleme gehäuft bei den Heuristiken H4 und H5 auf. Abbildung 6.4 zeigt die Verteilung der Probleme auf die einzelnen Heuristiken.

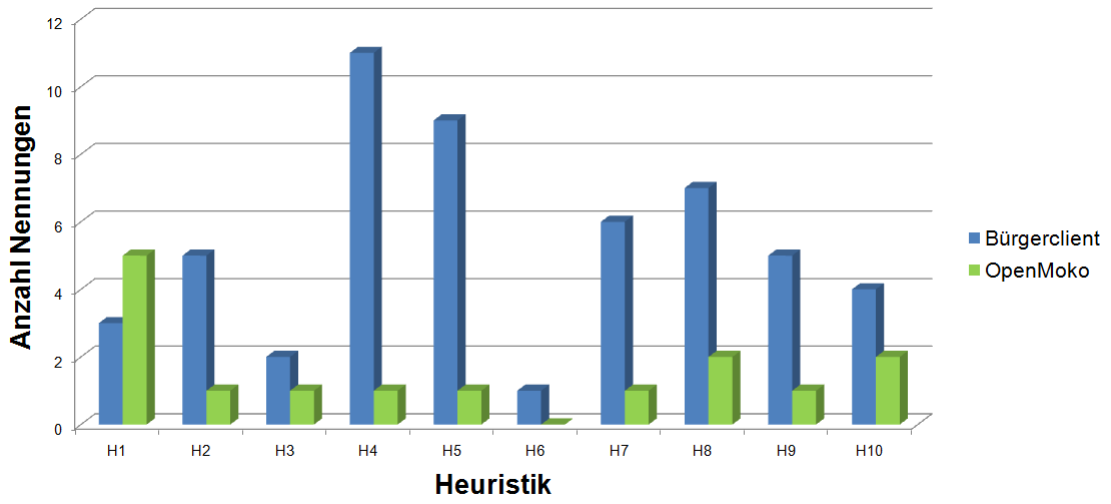

Abbildung 6.4.: Anzahl der gefundenen Usability-Probleme (konsolidierte Liste) aufgeteilt nach zugeordneter Heuristik.

Die meisten Probleme des Bürgerclients befinden sich im Konfigurationsdialog. Die zusammengefasste Liste enthält insgesamt 19 Einträge zu diesem Dialog, davon zwei zum Menüeintrag „Hilfe“, sieben zum Unterdialog „PIN-Einstellungen “ und zehn allgemeine Bemerkungen.

Die konsolidierte Liste enthält jeweils vier Einträge zu den drei einzelnen Dialogfenstern der eID-Authentisierung auf dem OpenMoko. Weitere vier Einträge sind dem Dialog zur Änderung der PIN zugeordnet. Des Weiteren gibt es noch zwei allgemeine Einträge, die keinem konkreten Fenster zugeordnet sind.

Die durchschnittliche Bewertung der im Bürgerclient gefundenen Probleme beträgt 1.96 ($SD = 0.92$), beim OpenMoko sind es 2.06 ($SD = 0.87$). Der Anteil an „schwerwiegenden“ Problemen ist dabei bei beiden Prüfgegenständen in etwa gleich groß.

	H1	H2	H3	H4	H5	H6	H7	H8	H9	H10
Bürgerclient	2	1.5	3.5	1.64	2.56	2	2	1.14	2.2	2.38
OpenMoko	2.1	1	2	3	3	-	2	1.25	4	1.5

Tabelle 6.2.: Durchschnittliche Bewertungen der Probleme nach Heuristik (konsolidierte Liste)

Beim Bürgerclient erhielten 13 von 53 Nennungen eine Bewertung von drei oder vier ($\approx$ 25%), beim OpenMoko waren es vier von 18 Nennungen ($\approx$ 22%). Tabelle 6.2 listet die durchschnittlichen Bewertungen nach zugehöriger Heuristik auf.

6.4. Diskussion

Die Evaluatoren zeigten sehr unterschiedliche Vorgehensweisen. Ein Evaluator ging beispielsweise sehr defektorientiert vor. Er verbrachte viel Zeit damit, Fehler in den Anwendungen zu provozieren, indem er beispielsweise den Ausweis während des Authentisierungsvorgangs aus dem Lesegerät entfernte oder auf dem OpenMoko mehrere Instanzen des eID-Programms gleichzeitig startete. Ein anderer Evaluator beschäftigte sich hingegen weitestgehend mit dem Erfolgsfall der Authentisierung und konzentrierte sich vor allem auf die Bewertung der Verständlichkeit aller Programmelemente.

Die unterschiedliche Vorgehensweise zeigt sich auch an der geringen Überschneidung der Einzellisten der gefundenen Probleme. Dadurch, dass sich die einzelnen Evaluatoren auf unterschiedliche Aspekte und Bereiche der Anwendungen konzentrierten, wurde also ein breiteres Spektrum an Usability-Problemen aufgedeckt. Durch das Hinzuziehen von zusätzlichen Evaluatoren hätten vermutlich noch weitere Probleme aufgedeckt werden können. Allerdings zeigen Erfahrungsberichte aus der Praxis, dass drei bis fünf Evaluatoren meist ausreichen, um etwa 75 % der vorhandenen Usability-Probleme aufzudecken.

Es existieren mehrere Ursachen dafür, dass im Bürgerclient sehr viel mehr Usability-Probleme gefunden wurden als in dem mobilen System. Zum einen enthielt der Bürgerclient noch sehr viele funktionale Fehler, welche von den Evaluatoren festgehalten wurden. So führte beispielsweise die Abwahl von Zugriffsrechten zum Abbruch der Authentisierung und beim Ablauf des Zeitlimits musste der Ausweis zunächst aus dem Lesegerät entfernt werden, bevor ein neuer Durchlauf gestartet werden konnte.

Darüber hinaus verfügte der Bürgerclient über sehr viel mehr Funktionen als das mobile System. Durch den vergleichsweise geringen Umfang der Programme auf dem OpenMoko war hier prinzipiell weniger Potenzial für Usability-Probleme vorhanden. Durch die Verwendung zweier unterschiedlicher Lesegeräte wurde der überprüfte Umfang des Bürgerclients im Vergleich zum OpenMoko noch weiter erhöht.

Bei einer Reihe der aufgedeckten Probleme handelte es sich nicht um fehlerhafte

Gestaltung, sondern um funktionale Fehler der Anwendungen. Teilweise handelte es sich dabei um sporadisch auftretende Fehler. So stürzte beispielsweise der Bürgerclient beim Aktivieren der PIN einmal ab, als der Evaluator den Ausweis aus dem Lesegerät entfernte. Der Absturz ließ sich allerdings nicht reproduzieren. Andere Fehler waren permanent vorhanden. So führte das Abwählen einzelner Zugriffsrechte beim Bürgerclient immer zum Scheitern der Authentisierung. Ein Evaluator entdeckte, dass es auf dem OpenMoko zu Fehlern kam, wenn mehrere Instanzen des eID-Programmes gleichzeitig gestartet wurden. Dieser Anwendungsfall wurde während der Entwicklung des Programmes nicht bedacht und daher auch nicht im Voraus getestet. Es zeigt sich, dass jede Form von Usability-Testing eben auch klassisches Software-Testing ist, bei dem Programmfehler aufgedeckt werden und so die Qualität der Software erhöht wird.

Einige Anzeichen ließen erkennen, dass die eID-Anwendung auf dem OpenMoko tatsächlich gebrauchstauglicher gestaltet ist als der Bürgerclient. Die Reduzierung der Funktionalität auf das notwendige Minimum kann beispielsweise auch als gebrauchstaugliche Gestaltung angesehen werden. Zusätzliche Funktionen, welche selten oder gar nicht benötigt werden und dennoch an prominenter Stelle im User Interface angeboten werden, können die Benutzer verwirren und von ihren eigentlichen Aufgaben ablenken. Sie stellen damit einen Verstoß gegen Nielsens Heuristik H7 dar. Die Evaluatoren bemängelten beim Bürgerclient eine Reihe an Funktionen, welche ihnen überflüssig erschienen, wie zum Beispiel die Möglichkeit, sich technische Details des Ausweises - wie etwa das ATR des Ausweises - anzeigen lassen zu können.

Ein weiterer Hinweis auf den Erfolg der verwendeten Usability-Engineering-Methoden war das explizite Lob einzelner Details des eID-Programmes auf dem OpenMoko durch die Evaluatoren. Positiv erwähnt wurden zum Beispiel die Hilfefunktion bei den Zugriffsrechten und die übersichtliche Gestaltung der einzelnen Dialogfenster.

Einige der von den Evaluatoren für das OpenMoko genannten Usability-Probleme lassen sich auf die recht geringe Rechenleistung des Handys zurückführen. So wurden beispielsweise ein stockendes Scrollen, das kurzzeitige Aufblinken des Desktops beim Fensterwechsel und eine Verzögerung zwischen der Eingabe und der Darstellung einer Ziffer bemängelt. Bei diesen Punkten handelt es sich nicht um prinzipielle Gestaltungsfehler. Sie ließen sich durch die Verwendung effizienterer Technologien (wie zum Beispiel zu Beginn von Abschnitt 5.3 beschrieben) oder den Umstieg auf ein leistungsfähigeres Handy beheben.

7. Fazit

7.1. Zusammenfassung

In dieser Arbeit wurde die Verwendung der eID-Authentisierung des neuen Personalausweises auf einem Mobiltelefon untersucht. Ziel der Arbeit war es, diese neue Authentisierungsmethode für ein existentes Handy umzusetzen. Als Zielplattform wurde dabei das OpenMoko gewählt. In dieses Telefon wurde ein RFID-Lesegerät eingebaut, um mit dem Personalausweis kommunizieren zu können. Des Weiteren wurde eine Reihe von Softwarekomponenten entwickelt, welche den Zugriff auf den Personalausweis gemäß den technischen Richtlinien des BSI realisieren.

Sicherheit und Gebrauchstauglichkeit werden häufig als sich zwangsläufig gegenseitig ausschließende Eigenschaften informationstechnischer Systeme angesehen. Die aus dieser Haltung teilweise resultierende mangelhafte Gebrauchstauglichkeit heutiger Sicherheitsmechanismen kann dazu führen, dass diese Mechanismen fehlerhaft oder gar nicht genutzt werden. Um diesem Problem zu begegnen, lag der Fokus bei der Entwicklung auf der Sicherstellung der Usability der eID-Authentisierung. Hierzu kamen in allen Entwicklungsphasen Methoden des Usability-Engineerings zum Einsatz.

In der Planungsphase wurde ein Usability-Test zur gleichzeitigen Handhabbarkeit von Ausweis und Handy durchgeführt, um prinzipielle Schwierigkeiten in der Bedienung so früh wie möglich zu erkennen. Bei diesem Test sollten Versuchspersonen auf drei verschiedenen Handymodellen eine PIN eingeben und gleichzeitig einen Testausweis an das Handy halten. Der Versuch wurde sowohl mit älteren als auch mit jüngeren Versuchspersonen durchgeführt. Der Test lieferte eine Reihe von Hinweisen auf Gestaltungsfehler in der verwendeten Oberfläche zur PIN-Eingabe. Älteren Benutzern fiel es schwerer, die PIN-Eingabe zu bewältigen, was aber eher auf die genannten Gestaltungsfehler als auf ein grundsätzliches Handhabbarkeitsproblem zurückzuführen war.

Die eigentliche Entwicklung erfolgte partizipativ. Nach jedem Iterationszyklus wurde die jeweils aktuelle Version von potenziellen Nutzern begutachtet und kommentiert. Ihr Feedback floss direkt in die Entwicklung ein und half dabei, das Programm an die Bedürfnisse realer Benutzer anzupassen.

Nach dem Abschluss der Implementierung wurde das System einer heuristischen Evaluation durch mehrere Usability-Experten unterzogen. Es wurde dabei mit der offiziellen Software für den neuen Personalausweis - dem Bügerclient - verglichen. Die Evaluatoren deckten eine Reihe von Verstößen gegen Prinzipien gebrauchstauglicher Gestaltung im Bürgerclient auf. Auch in dem Handy-basierten System konnten noch

einige Verstöße entdeckt werden. Insgesamt lassen die Ergebnisse der Evaluation jedoch die Schlußfolgerung zu, dass die gebrauchstaugliche Umsetzung der mobilen eID-Authentisierung im Rahmen dieser Arbeit gelungen ist.

Die Arbeit zeigt, dass die Methoden des Usability-Engineerings auch bei der Entwicklung von neuen Verfahren im Bereich der IT-Sicherheit, welche traditionell als besonders benutzerunfreundlich gelten, erfolgreich einsetzbar sind. Durch eine bessere Bedienbarkeit von Sicherheitsmechanismen - wie etwa Authentisierungsmethoden - können Bedienungsfehler vermieden und die Nutzerakzeptanz erhöht werden. Dadurch kann die Verwendung dieser Methoden zu einer Erhöhung des tatsächlich erreichbaren Sicherheitsniveaus beitragen.

7.2. Ausblick

Das im Rahmen dieser Arbeit entwickelte System bietet vielfältige Möglichkeiten für weitere Entwicklungen und Untersuchungen. An erster Stelle sollte dabei die Integration der eCard-API stehen, damit das System tatsächlich für die eID-Authentisierung mit realen Diensten verwendet werden kann.

Anschließend können weitere Usability-Tests durchgeführt werden. Besonders interessant dürften dabei Feldtests sein, welche unter realen Einsatzbedingungen durchgeführt werden. Dadurch könnte untersucht werden, ob die eID-Authentisierung auch unter mobilen Einsatzbedingungen und wechselnden Umwelteinflüssen - wie etwa Beleuchtung und Lautstärke der Umgebung - noch gut bedienbar ist.

Ein entscheidender Faktor für den Erfolg der eID-Authentisierung wird die Akzeptanz der neuen Technologie in der Bevölkerung sein. Erfahrungen aus anderen Ländern zeigen, dass die neuen Funktionen elektronischer Ausweisdokumente in der Praxis häufig ungenutzt bleiben (siehe beispielsweise [AS09] und [LYC09]). Erste Untersuchungen deuten darauf hin, dass es bei der Einführung des neuen Personalausweises in Deutschland möglicherweise zu ähnlichen Akzeptanzproblemen kommen könnte. So gaben bei einer Studie des Branchenverbandes BITKOM 21 Prozent der befragten Personen an, sich vor der Einführung des neuen Personalausweises noch einen alten Ausweis zulegen zu wollen[1]. Weitere zwölf Prozent gaben an, dies bereits getan zu haben. Eine ähnliche Studie des TeleTrusT e.V. kam zu vergleichbaren Ergebnissen[2]. Auf dieser Grundlage ist es anzuraten, auch für den mobilen Einsatz der eID-Authentisierung mittels NFC-Handy eine Akzeptanzstudie durchzuführen.

Sowohl für die Entwicklung von sicherer als auch von gebrauchstauglicher Software existieren spezielle Vorgehensmodelle. Wünschenswert wäre ein kombiniertes Modell für die Entwicklung von Systemen, das sowohl gebrauchstauglich als auch sicher ist.

[1] BITKOM – Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e.V.: *Studie „Neuer Personalausweis“ – Aktuelle Akzeptanz in der Bevölkerung* Verfügbar unter: http://www.bitkom.org/de/publikationen/38338_63570.aspx. Letzter Zugriff: 11.08.2010

[2] TeleTrusT e.V.: *Neuer Personalausweis: Für breite Akzeptanz dringend weitere Aufklärung nötig.* 10. März 2010. Verfügbar unter: http://www.teletrust.de/startseite/pressemeldung/?tx_ttnews[tt_news]=165. Letzter Zugriff: 11.08.2010

In den letzten Jahren sind zwar zahlreiche Methoden für die Planung, Entwicklung und Bewertung sicherer und gebrauchstauglicher Systeme vorgestellt worden, eine ganzheitliche Methodik für den kompletten Entwicklungszyklus existiert aber derzeit noch nicht.

A. Die kryptografischen Protokolle der Extended Access Control

A.1. PACE

Das PACE-Protokoll dient der Schlüsselvereinbarung zwischen dem Ausweis (PICC) und dem Terminal (PCD). Basis des Protokolls ist das Diffie-Hellman (DH) Protokoll. Dieses kann entweder in der klassischen Variante oder aber auf der Basis elliptischer Kurven (ECDH) verwendet werden. Da Diffie-Hellman in seiner Reinform jedoch anfällig für Man-in-the-middle (MITM) Angriffe ist, muss ein komplexeres Protokoll zur Schlüsselvereinbarung verwendet werden. Die Grundlage zum Ausschluss von MITM Angriffen bildet dabei das geteilte Geheimnis der beiden Parteien (z.B. die eID-PIN).

Aus diesem Geheimnis wird zunächst ein symmetrischer Schlüssel abgeleitet. Dazu wird folgende Funktion verwendet:

$$\mathrm{KDF}(\Pi) := \mathbf{H}(\Pi||C)$$

dabei bezeichnet Π das geteilte Geheimnis und der Operator $||$ die Konkatenation. C ist ein 32-Bit big-endian Zähler, der auf den Wert Drei gesetzt wird.

Der Ausweis erzeugt nun eine Nonce s und verschlüsselt sie mit dem aus Π abgeleiteten Schlüssel K_Π zum Kryptogramm z. Dieses Kryptogramm wird anschließend zusammen mit weiteren Hilfdsdaten an das Terminal übermittelt und anschließend dazu verwendet einen neuen (EC)DH-Generator zu finden. Dafür existieren verschiedene mapping-Verfahren, die in Unterabschnitt A.1.1 beschrieben sind. Der neue Generator ist nur den beiden Parteien PICC und PCD bekannt. Mit diesem Generator kann nun das (EC)DH-Protokoll durchgeführt werden, ohne dass die Gefahr von Man-in-the-middle Angriffen besteht.

Aus dem Ergebnis der EC(DH)-Schlüsselvereinbarung werden anschließend zwei symmetrische Schlüssel $\mathrm{K_{Enc}}$ und $\mathrm{K_{Mac}}$ abgeleitet. Dazu kommt wieder die oben beschrieben Schlüsselableitungsfunktion KDF zum Einsatz, wobei der Zähler C für die Ableitung von $\mathrm{K_{Enc}}$ auf den Wert Eins und für die Ableitung von $\mathrm{K_{Mac}}$ auf den Wert Zwei gesetzt wird.

$\mathrm{K_{Mac}}$ wird anschließend noch dazu verwendet einen Prüfwert aus dem jeweiligen öffentlichen Ephemeralschlüssel der Gegenseite zu berechnen. Durch die Verifikation dieser Prüfwerte können sich beide Parteien sicher sein, dass die jeweilige Gegenseite nun im Besitz der korrekten Sitzungsschlüssel für das Secure Messaging ist.

PICC		PCD
statische Domänenparameter D_{PICC}		
zufällige Nonce $s \in_R \text{Dom}(E)$		
$z = \mathbf{E}(K_\pi, s)$	$\xrightarrow[z]{D_{\text{PICC}}}$	$s = \mathbf{D}(K_\pi, z)$
Zusätzliche Daten für **Map**()	$\longleftrightarrow$	Zusätzliche Daten für **Map**()
$\widetilde{D} = \mathbf{Map}(D_{\text{PICC}}, s)$		$\widetilde{D} = \mathbf{Map}(D_{\text{PICC}}, s)$
Wähle Ephemeralschlüssel $(\widetilde{SK_{\text{PICC}}}, \widetilde{PK_{\text{PICC}}}, \widetilde{D})$		Wähle Ephemeralschlüssel $(\widetilde{SK_{\text{PCD}}}, \widetilde{PK_{\text{PCD}}}, \widetilde{D})$
	$\xleftrightarrow[\widetilde{PK_{\text{PICC}}}]{\widetilde{PK_{\text{PCD}}}}$	
Überprüfe, dass $\widetilde{PK_{\text{PCD}}} \neq \widetilde{PK_{\text{PICC}}}$		Überprüfe, dass $\widetilde{PK_{\text{PICC}}} \neq \widetilde{PK_{\text{PCD}}}$
$K = \mathbf{KA}(\widetilde{SK_{\text{PICC}}}, \widetilde{PK_{\text{PCD}}}, \widetilde{D})$		$K = \mathbf{KA}(\widetilde{SK_{\text{PCD}}}, \widetilde{PK_{\text{PICC}}}, \widetilde{D})$
	$\xleftarrow{T_{\text{PCD}}}$	$T_{\text{PCD}} = \mathbf{MAC}(K_{\text{MAC}}, \widetilde{PK_{\text{PICC}}})$
$T_{\text{PICC}} = \mathbf{MAC}(K_{\text{MAC}}, \widetilde{PK_{\text{PCD}}})$	$\xrightarrow{T_{\text{PICC}}}$	

Abbildung A.1.: Ablauf des PACE-Protokoll

A.1.1. Mapping Verfahren

Sowohl für das klassische, als auch das elliptic curve Diffie-Hellman Protokoll wurden jeweils zwei verschiedene Mapping-Verfahren spezifiziert: das Generic Mapping und das Integrated Mapping. Beim Generic Mapping findet zunächst ein klassischer (EC)DH Durchlauf statt. Das Ergebnis h (bzw. der Punkt H bei der Verwendung elliptischer Kurven) fließen anschließend zusammen mit der Nonce s in die Berechnung des neuen Generators ein:

- DH Generic Mapping: $\tilde{g} = g^s * h$
- ECDH Generic Mapping: $\tilde{g} = G * s + H$

Die Berechnungen finden dabei in den durch die Domänenparameter D_{PICC} definierten Körpern $\mathbb{F}_p$, bzw. $E(\mathbb{F}_p)$ statt.

Beim Integrated mapping wählt das Terminal einen zufälligen Schlüssel K. Dieser wird im Klartext an den Ausweis übertragen. Beide Parteien nutzen K um die Nonce s zu verschlüsseln. Das resultierende Kryptogramm z' wird anschließend in den Restklassenring der Domänenparameter abgebildet. Bei der Verwendung des klassischen Diffie-Hellman Verfahrens geschieht das durch folgende Funktion:

$$f(z') := \text{int}(z')^a \bmod p$$

Wobei int die Umwandlung des Oktet-Strings in eine big-endian integer Repräsentation bezeichnet und a der Cofaktor der Domänenparameter ist.

Bei der Verwendung elliptischer Kurven wird das in [Ica09] beschriebene Verfahren verwendet, um z' auf einen Punkt der Kurve abzubilden. Voraussetzung dafür ist, dass der der Kurve zu Grunde liegende Körper $\mathbb{F}_{p^n}$ die folgenden beiden Voraussetzungen erfüllt:

1. $p > 3$
2. $p^n = 2 \bmod 3$

A.2. Terminal authentication

Zu Beginn der TA erzeugt das Terminal einen Ephemeralschlüsselpaar und sendet den komprimierten öffentlichen Schlüssel an den Ausweis. Die Art der Kompression hängt davon ab ob das klassische Diffie-Hellman-Verfahren oder ECDH genutzt wird. Beim klassischen Verfahren besteht die Kompression darin den SHA-1 Hash des öffentlichen Schlüssels zu berechnen. Beim ECDH wird lediglich die X-Koordinate des öffentlichen Punktes übertragen.

Zusätzlich zum komprimierten öffentlichen Schlüssel werden noch evenutell vorhandene *Authenticated Auxiliary Data* A_{PCD} übermittelt. Dabei handelt es sich um

die für die Sonderfunktionen der eID-Anwendung benötigten Daten, wie etwa das Referenzdatum für die Altersverifikation.

Der Ausweis generiert daraufhin erneut eine Nonce r_{PICC} und übermittelt diese an das Terminal. Dieses bildet nun die Konkatenation des öffentlichen Ephemerlschlüssels des Ausweis aus dem PACE-Protokoll mit der Nonce, dem eigenen komprimierten TA-Ephemeralschlüssel und A_{PCD} und signiert dieses Datum mit seinem statischen TA-Schlüssel.

Der Ausweis kann die resultierende Signatur s_{PCD} mit dem im Termialzertifikat enthaltenen öffentlichen TA-Schlüssel verifizieren und somit das Terminal authentifizieren.

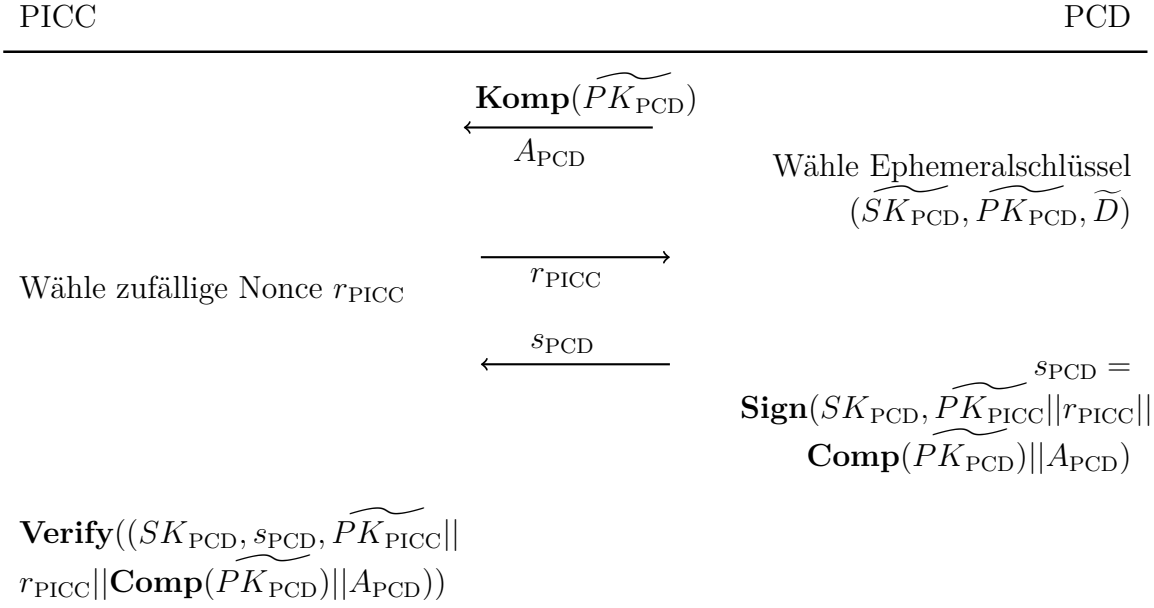

Abbildung A.2.: Der Ablauf der TA

A.3. Chip authentication

Zu Beginn des Protokolls liest das Terminal den statischen CA-Schlüssel des Ausweises aus. Anschließend findet wiederum ein (EC)DH Schlüsselaustausch zwischen Ausweis und Terminal statt. Dazu wird das statische CA-Schlüsselpaar des Ausweises und das im Rahmen der TA erzeugte Ephemeralschlüsselpaar des Terminals verwendet. Da die Schlüsselvereinbarung innerhalb eines (durch PACE etablierten) Secure Messaging Kanals stattfindet, ist sie nicht für Man-in-the-middle Angriffe anfällig.

Aus dem Ergebnis der Schlüsselvereinbarung werden Schlüssel für einen neuen Secure Messaging Kanal abgeleitet. Diese verwendet der Ausweis anschließend um einen Message Authentication Code über eine weitere Nonce zu berechnen. Zufallszahl und

MAC werden anschließend an das Terminal übermittelt. Durch die Verifikation des Tokens kann sich das Terminal versichern, dass der Ausweis im Besitz des korrekten MAC Schlüssel ist. Damit ist implizit sichergestellt, dass der Ausweis im Besitz des privaten CA-Schlüssels ist.

PICC	PCD
Statisches Schlüsselpaar $(SK_{\text{PICC}}, PK_{\text{PICC}}, D_{\text{PICC}})$	
	$\xrightarrow[D_{\text{PICC}}]{PK_{\text{PICC}}}$
$K = \mathbf{KA}(SK_{\text{PICC}}, \widetilde{PK_{\text{PCD}}}, D)$	$K = \mathbf{KA}(\widetilde{SK_{\text{PCD}}}, PK_{\text{PICC}}, D)$
Wähle zufällige Nonce r_{PICC}	
	$\xrightarrow[r_{\text{PICC}}]{T_{\text{PICC}}}$
$T_{\text{PICC}} = \mathbf{MAC}(K_{\text{MAC}}, \widetilde{PK_{\text{PCD}}})$	

Abbildung A.3.: Der Ablauf des CA Protokolls

B. Materialien des Usabilitytest zur Handhabbarkeit

Für den Usability Test zur gleichzeitigen Handhabbarkeit von Handy und Ausweis wurden die folgenden Materialien verwendet:

Fragebogen zum Benutzertest „PIN-Eingabe mit Personalausweis und Handy“

1. Fragen zur Person:

Alter	Geschlecht		Höchster Bildungsabschluss
	M	W	

2. Fragen zur Techniknutzung:

Besitzen Sie ein Handy		Haben Sie schon einmal mit dem Handy im Internet gesurft?	
Ja	Nein	Ja	Nein

3. Fragen zum Versuch:

Bewerten Sie die Benutzerfreundlichkeit der PIN Eingabe mit Handy und Personalausweis Dabei ist „1“ die schlechteste Bewertung und „7“ die beste.						
1. Telefon:						
1	2	3	4	5	6	7
Sehr schlecht						Sehr gut
2. Telefon:						
1	2	3	4	5	6	7
Sehr schlecht						Sehr gut
3. Telefon:						
1	2	3	4	5	6	7
Sehr schlecht						Sehr gut

4. Anmerkungen:

Checkliste Versuchsbeobachtungen

1. Reihenfolge Handys:
2. Stehend/Sitzend:
3. Fragen vor Versuchsbeginn:

Erstes Handy:

1. Wie wird das Handy gehalten:
2. Fallen Karte oder Handy herunter:
3. Subjektiver Eindruck:

4. Sonstiges:

Zweites Handy:

4. Wie wird das Handy gehalten:
5. Fallen Karte oder Handy herunter:
6. Subjektiver Eindruck:

4. Sonstiges:

Drittes Handy:

7. Wie wird das Handy gehalten:
8. Fallen Karte oder Handy herunter:
9. Subjektiver Eindruck:

4. Sonstiges:

Fragen nach dem Versuch:

Sonstiges:

Einverständniserklärung zur Verwendung der Filmaufnahmen

Hiermit gebe ich mein Einverständnis, dass meine Filmaufnahmen im Rahmen der Diplomarbeit von Dominik Oepen erhoben und verwendet werden. Die Filmaufnahmen werden nicht mit den weiteren erhobenen Daten zusammengeführt. Ich stimme der Verwendung für folgende Zwecke zu:

1. Verwendung zur Versuchsanalyse: [ ]
2. Verwendung für Präsentationszwecke im Rahmen der Diplomarbeit: [ ]
3. Veröffentlichung der Aufnahmen zusammen mit der Diplomarbeit auf DVD: [ ]

Ich wurde darüber in Kenntnis gesetzt, dass ich meine Einverständniserklärung durch ein Schreiben an

Dominik Oepen
Heidenfeldstraße 2
10249 Berlin

widerrufen kann.

Name: _____

Vorname: _____

_____ _____

Ort, Datum **Unterschrift**

C. Anleitung zum Kompilieren

Um die im Rahmen dieser Arbeit entwickelte Software auf dem OpenMoko zu verwenden, muss ein Image des verwendeten Betriebssystems SHR inklusive dieser Software erstellt werden. Im Folgenden wird beschrieben, wie ein solches Image inklusive der für die eID-Authentisierung benötigten Software erstellt werden kann. Ein fertiges Image ist außerdem auf der dieser Diplomarbeit beiliegenden CD im Ordner `image` enthalten.

Zunächst muss die für das Übersetzen der Software benötigte Crosscompile-Toolchain vorbereitet werden. SHR verwendet dafür OpenEmbedded. Zur Automatisierung der Einrichtung der Toolchain wird das SHR-Makefile verwendet (siehe Listing C.1). Dieses lädt alle benötigten Komponenten aus dem Internet und richtet die Build-Umgebung ein. Im Folgenden wird davon ausgegangen, dass sich die Build-Umgebung im Ordner `/home/do/shr/` befindet. Wird ein anderer Ordner verwendet, so müssen die Pfade in den Befehlen entsprechend angepasst werden.

```
1 cd /home/do/shr/
2 wget http://shr.bearstech.com/Makefile
3 make setup
```

Listing C.1: Vorbereitung der Build-Umgebung für SHR

Um das Image zu erstellen, müssen die folgenden Programme installiert sein: python-2.6, git, svn, cvs, make, gcc, g++, patch, help2man, diffstat, texi2html, bzip2, gawk, tar, md5sum, texinfo und chrpath.

Als Nächstes gilt es, die im Rahmen dieser Arbeit angefertigten bitbake-Rezepte in die Build-Umgebung zu kopieren. Diese befinden sich auf der beiliegenden CD im Ordner `src/bitbake`. Anschließend kann der Übersetzungsvorgang angestoßen werden.

```
1 cd /media/cdrom/
2 cp -r src/bitbake/* /home/do/shr/shr-testing/openembedded/
    recipes/
3 cd /home/do/shr/shr-testing/
4 echo 0 > /proc/sys/vm/mmap_min_addr
5 make image
```

Listing C.2: Kompilieren des Images

Der Befehl in Zeile 3 von Listing C.2 ist auf Ubuntu-Systemen notwendig, um die Bibliothek eglibc kompilieren zu können. Der anschließende Übersetzungsvorgang benötigt mehrere Stunden. Danach befindet sich das fertige Image im Un-

terordner `tmp/deploy/image/om-gta02`. Das Image besteht dabei aus dem Kernel (`uImage-om-gta02-latest.bin`) und dem Root-Filesystem (`lite-om-gta02.jffs2`).

Um das Image auf das OpenMoko aufzuspielen, kommt das Script neotool zum Einsatz, welches sich auf der beiliegenden CD im Ordner `Scripte` befindet. Das Script muss mit `root` Rechten gestartet werden und kann anschließend über eine grafische Oberfläche bedient werden (siehe Abbildung C.1). Über den Menüpunkt „Flash your Neo“ kann nun das Image auf das OpenMoko installiert werden.

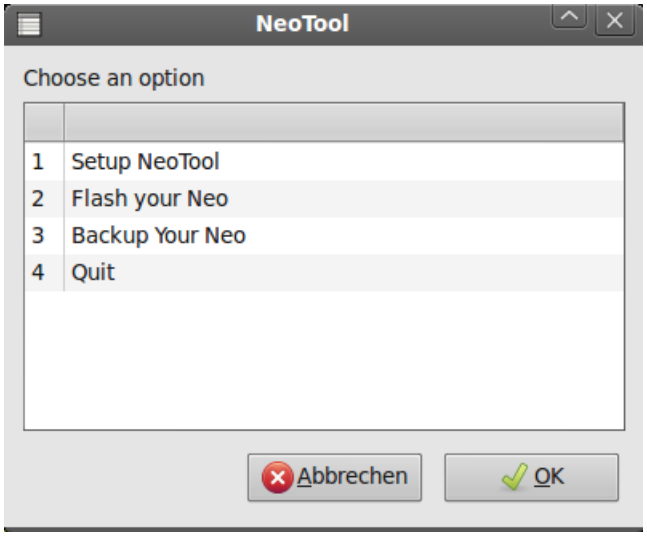

Abbildung C.1.: Grafische Oberfläche des neotool

Um das Image auf das OpenMoko aufspielen zu können, muss dieses zunächst in den NOR-Flash gebootet werden. Dafür muss der AUX-Knopf auf der oberen linken Seite des Telefons gedrückt gehalten werden und anschließend der Power Knopf auf der rechten unteren Seite gedrückt werden. Daraufhin erscheint ein Boot-Menü auf dem Display des OpenMoko. Nun kann das Telefon per USB mit dem Computer verbunden und der Flash-Vorgang via neotool gestartet werden.

Bevor die eID-Applikation auf dem OpenMoko verwendet werden kann, muss dieses in den USB-Host Modus versetzt werden (vergleiche Listing 5.1). Anschließend aktiviert der Befehl `ifdnfc-activate yes` den IFD-Handler von ifdnfc, sodass der Touchatag verwendet werden kann.

D. Materialien der heuristischen Evaluation

1 Nielsens Heuristiken

H1 **Sichtbarkeit des Systemzustands:** Das System sollte den Nutzer immer informieren, was gerade vorgeht, durch geeignetes Feedback innerhalb angemessener Zeit.

H2 **Übereinstimmung zwischen System und realer Welt:** Das System soll die Sprache des Nutzers sprechen, mit Wörtern, Formulierungen und Konzepten, die dem Nutzer vertraut sind, nicht mit systemorientierten Begriffen. Folge Konventionen aus der Realität, sodass Informationen in natürlicher und logischer Reihenfolge erscheinen.

H3 **Benutzerkontrolle und Freiheit:** Nutzer wählen Softwarefunktionen oft versehentlich. Sie brauchen einen klar markierten Notausgang, um einen unabsichtlich erreichten Zustand verlassen zu können, ohne durch einen ausgedehnten Dialog hindurch zu müssen. Unterstütze Undo und Redo.

H4 **Konsistenz und Standards:** Die Nutzer sollen nicht überlegen müssen, ob verschiedene Begriffe, Situationen und Aktionen dasselbe bedeuten. Folge Plattform-Konventionen.

H5 **Fehler vermeiden:** Besser noch als gute Fehlermeldungen ist ein sorgfältiges Design, das verhindert, dass es überhaupt erst Probleme auftreten.

H6 **Erkennen vor Erinnern:** Mache Objekte, Aktionen und Optionen sichtbar. Der Nutzer sollte nicht gezwungen werden, sich Informationen aus einem Teil eines Dialogs für einen anderen Teil zu merken. Anleitungen zur Benutzung des Systems sollten sichtbar oder leicht auffindbar sein, wann immer angemessen.

H7 **Flexibilität und effiziente Nutzung:** Beschleunigungsmöglichkeiten die der ungeübte Nutzer nicht sieht, können für erfahrene Nutzer die Interaktionsgeschwindigkeit erhöhen, sodass das System sowohl für Neulinge, als auch für Experten geeignet ist. Erlaube Nutzern, häufige Aktionen auf ihre Bedürfnisse zuzuschneiden.

H8 **Ästhetisches und minimalistisches Design:** Dialoge sollten keine Information enthalten, die irrelevant ist, oder selten benötigt wird. Jedes Extra an Information in einem Dialog konkurriert mit relevanten Informationen und vermindert deren relative Sichtbarkeit.

H9 **Unterstützung beim Erkennen, Verstehen und Bearbeiten von Fehlern:** Fehlermeldungen sollten in klarer Sprache (keine Kodierung) gegeben werden. Sie sollen das Problem genau beschreiben und konstruktiv eine Lösung vorschlagen.

H10 **Hilfe und Dokumentation:** Obwohl es besser ist, wenn das System ohne Dokumentation benutzt werden kann, kann es nötig sein, Hilfe und Information mitzugeben. Jede solche Information sollte leicht zu durchsuchen sein, die Aufgaben des Nutzers in den Mittelpunkt stellen und konkrete Schritte zur Ausführung nennen. Die Dokumentation sollte nicht ausführlich sein.

2 Bewertungskriterien

Bei der Bewertung der Schwere eines Usability-Problems gilt es vier Faktoren zu berücksichtigen:

1. *Frequenz* des Auftretens (einmalig bis laufend)
2. *Einfluss* auf die Arbeitsabläufe (Nichtigkeit bis schwere Störung)
3. *Persistenz* des Auftretens (zufälliges bis regelmäßiges Auftreten)
4. *Markteinfluss* (ein insgesamt ungünstiger Eindruck sollte vermieden werden)

Zur Bewertung der einzelnen Verstöße gegen die Heuristiken soll die folgende Skala verwendet werden:

0 = Ich stimme nicht zu, dass dies überhaupt ein Usability Problem ist.

1 = Kosmetisches Problem – Braucht nicht behoben zu werden, außer der Projektrahmen stellt genug Zeit dafür zur Verfügung.

2 = Geringfügiges Usability-Problem – Der Behebung sollte untergeordnete Priorität gegeben werden.

3 = Bedeutendes Usability-Problem – Es ist wichtig, es zu beheben, und sollte daher eine hohe Priorität haben.

4 = Usability Katastrophe – Es ist ein Muss, dieses Problem zu beheben, bevor das Produkt ausgeliefert wird.

Bitte nutze diese Tabelle, um die von dir gefundenen Verstöße gegen die einzelnen Heuristiken zu dokumentieren. Notiere bitte jedes Problem einzeln. Wenn du in einem Dialog drei Probleme findest, so erstelle dafür drei Einträge. Falls du einem Problem keine Heuristik zuordnen kannst, so lasse die Spalte einfach leer.

Fundort	Beschreibung des Problems	Heuristik	Bewertung

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

ANOVA Analysis of Variance

APDU Application Protocol Data Unit

ATR Answer To Reset

BAC Basic Access Control

BSI Bundesamt für Sicherheit in der Informationstechnik

CAN Card Access Number

CAR Certificate Authority Reference

CA Chip Authentication

CHAT Certificate Holder Authorisation Template

CHR Certificate Holder Reference

CSCA Country Signing Certificate Authority

CVCA Country Validating Certificate Authority

DS Document Signer

DV Document Verifier

ePA elektronischer Personalausweis

FAR False Acceptance Rate

FBZ Fehlbedienungszähler

FRR False Rejection Rate

LWUIT Lightweight User Interface Toolkit

MRZ Machine Readable Zone

NFC Near Field Communication

nPA neuer Personalausweis

PACE Password Authenticated Connection Establishment

PCD Proximity Coupling Device

PICC Proximity Integrated Circuit Card

PKI Public Key Infrastructure

PUK PIN Unblocking Key

QES Qualifizierte Elektronische Signatur

RFID Radio Frequency Identification

RI Restricted Identification

SAM Secure Access Module

SSEE Sichere Signaturerstellungseinheit

SWIG Simplified Wrapper and Interface Generator

TA Terminal Authentication

UID Unique Identifier

ZDA Zertifizierungsdiensteanbieter

Literaturverzeichnis

[And08] Anderson, Ross: *Security Engineering – A Guide To Building Dependable Distributed Systems*. Wiley Publishing, Indianapolis, 2. Auflage, 2008.

[AS99] Adams, Anne und Martina Angela Sasse: *Users are not the enemy*. Communications of the ACM, 42(12):40–46, 1999.

[AS09] Aichholzer, Georg und Stefan Strauss: *The Citizen's Role in National Electronic Identity Management - A Case-study on Austria*. Advances in Human-oriented and Personalized Mechanisms, Technologies, and Services, International Conference on, 0:45–50, 2009.

[Bei10] Beilke, Kristian: *Mobile eCard-API*. Diplomarbeit, Humboldt Universität zu Berlin, 2010. Zur Veröffentlichung eingereicht.

[BFK09] Bender, Jens, Marc Fischlin und Dennis Kügler: *Security Analysis of the PACE Key-Agreement Protocol*. In: *ISC '09: Proceedings of the 12th International Conference on Information Security*, Seiten 33–48, Berlin, Heidelberg, 2009. Springer-Verlag.

[BS03] Brostoff, Sacha und M. Angela Sasse: *"Ten strikes and you're out": Increasing the number of login attempts can improve password usability*. In: *Proceedings of CHI 2003 Workshop on HCI and Security Systems*. John Wiley, 2003.

[BSI04] BSI: *Risiken und Chancen des Einsatzes von RFID-Systemen*. Technischer Bericht, Bundesamt für Sicherheit in der Informationstechnik, Bonn, Oktober 2004.

[BSI09a] BSI: *Anforderungen an Chipkartenleser mit ePA Unterstützung*. Technische Richtlinie 03119, Bonn, Dezember 2009. Version 1.1.

[BSI09b] BSI: *Architektur Elektronischer Personalausweis*. Technische Richtlinie 03127, Bonn, Oktober 2009. Version 1.01.

[BSI09c] BSI: *eCard-API-Framework - Protocols*. Technical Guideline 03112-7, Bonn, Juli 2009. Version 1.1.

[BSI09d] BSI: *eCards mit kontaktloser Schnittstelle als sichere Signaturerstellungseinheit*. Technische Richtlinie 03117, Bonn, Juli 2009. Version 1.0.

[BSI10a] BSI: *Advanced Security Mechanisms for Machine Readable Travel Documents*. Technical Guideline 03110, Bonn, März 2010. Version 2.03.

[BSI10b] BSI: *EAC-PKI'n für den elektronischen Personalausweis*. Technische Richtlinie 03128, Bonn, Juli 2010. Version 1.02.

[BSSW10] Borges, Prof. Dr. Georg, Prof. Dr. Jörg Schwenk, Prof. Dr. Carl-Friedrich Stuckenberg und Dr. Christoph Wegener: *Identitätsdiebstahl und Identitätsmissbrauch im Internet*. 2010.

[DR93] Dumas, Joseph F. und Janice C. Redish: *A Practical Guide to Usability Testing*. Greenwood Publishing Group Inc., Westport, CT, USA, 1993.

[FB04] Folmer, Eelke und Jan Bosch: *Architecting for usability: a survey*. Journal of Systems and Software, 70(1-2):61 – 78, 2004.

[Fin08] Finkenzeller, Klaus: *RFID-Handbuch: Grundlagen und praktische Anwendungen induktiver Funkanlagen, Transponder, kontaktlosen Chipkarten und NFC*. Hanser, München, 5. Auflage, 2008.

[FK04] Finke, Thomas und Harald Kelte: *Radio Frequency Identification – Abhörmöglichkeiten der Kommunikation zwischen Lesegerät und Transponder am Beispiel eines ISO14443-Systems*. Technischer Bericht, Bundesamt für Sicherheit in der Informationstechnik, 2004.

[FvGB03] Folmer, Eelke, Jilles van Gurp und Jan Bosch: *A Framework for capturing the Relationship between Usability And Software*. In: *Software Process: Improvement and Practice*, Seiten 67–87, 2003.

[GP96] Gerhardt-Powals, Jill: *Cognitive engineering principles for enhancing human-computer performance*. In: *International Journal of Human-Computer Interaction*, Band 8, Seiten 189–211. Taylor & Francis Group, April 1996.

[Hor09] Horsch, Moritz: *MobilePACE*. Diplomarbeit, TU Darmstadt, September 2009.

[ICA08] ICAO: *Doc 9303, Machine Readable Travel Documents, Part 3, Specifications for Electronically Enabled MRtds with Biometric Identification Capability*, Band 2. International Civil Aviation Organisation, 3 Auflage, Mai 2008.

[Ica09] Icart, Thomas: *How to Hash into Elliptic Curves*. In: *CRYPTO '09: Proceedings of the 29th Annual International Cryptology Conference on Advances in Cryptology*, Seiten 303–316, Berlin, 2009. Springer-Verlag.

[IO84] Ives, Blake und Margrethe H Olson: *User Involvement and MIS Success: A Review of Research*. INFORMS: Institute for Operations Research, 30:586–603, 1984.

[ISO98] ISO 9241-11: *ISO 9241-11: Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability*. ISO/IEC, März 1998.

[ISO00] ISO 14443-4: *Identification cards — Contactless integrated circuit(s) cards — Proximity cards — Part 4: Transmission protocol*. ISO/IEC, März 2000.

[ISO05] ISO 7816-4: *Identification cards — Integrated circuit cards — Part 4: Organization, security and commands for interchange*. ISO/IEC, 2. Auflage, Januar 2005.

[Kam10] Kampe, Ingo: *NFC-Telefon als PACE-fähiges Lesegerät für elektronische Ausweisdokumente*. Diplomarbeit, Humboldt Universität zu Berlin, 2010. Zur Veröffentlichung eingereicht.

[Kuj03] Kujala, Sari: *User involvement: a review of the benefits and challenges*. Behaviour & Information Technology, 22(1):1–16, January 2003.

[LLG+05] Lim, Daihyun, Jae W. Lee, Blaise Gassend, G. Edward Suh, Marten Van Dijk und Srinivas Devadas: *Extracting Secret Keys From Integrated Circuits*. IEEE Transactions on very large scale integration (VLSI) systems, 13(10):1200–1205, October 2005.

[LYC09] Loo, W.H., Paul H.P. Yeow und S.C. Chong: *User acceptance of Malaysian government multipurpose smartcard applications*. Government Information Quarterly, 26(2):358 – 367, 2009. Building the Next-Generation Digital Government Infrastructures.

[Mor81] Moran, Thomas P.: *The Command Language Grammar: a representation for the user interface of interactive computer systems*. International Journal of Man-Machine Studies, 15(1):3–50, 1981.

[Nie94] Nielsen, Jakob: *Usability Engineering*. Morgan Kaufmann Publishers, Oktober 1994.

[Nie05] Nielsen, Jakob: *How to Conduct a Heuristic Evaluation*. http://www.useit.com/papers/heuristic/heuristic_evaluation.html, 2005. Zuletzt besucht am 30.07.2010.

[NM90] Nielsen, Jakob und Rolf Molich: *Heuristic evaluation of user interfaces*. In: *CHI '90: Proceedings of the SIGCHI conference on Human factors in computing systems*, Seiten 249–256. ACM Press, 1990.

[Nor02] Norman, Donald A.: *The design of everyday things.* Basic Books, New York, 2002.

[PHVS04] Po, Shirlina, Steve Howard, Frank Vetere und Mikael B. Skov: *Heuristic Evaluation and Mobile Usability: Bridging the Realism Gap.* In: *Proceedings of Mobile HCI 2004.* Springer-Verlag, 2004.

[Plö08] Plötz, Henryk: *Mifare Classic – Eine Analyse der Implementierung.* Diplomarbeit, Humboldt–Universität zu Berlin, Berlin, Oktober 2008.

[SB06] Sarodnick, Florian und Henning Brau: *Methoden der Usability Evaluation – Wissenschaftliche Grundlagen und praktische Anwendungen.* Verlag Hans Huber, Bern, 2006.

[SBW01] Sasse, M. A., S. Brostoff und D. Weirich: *Transforming the 'weakest link' — a human/computer interaction approach to usable and effective security.* BT Technology Journal, 19(3):122–131, 2001.

[Sch09] Schmeh, Klaus: *Elektronische Ausweisdokumente - Grundlagen und Praxisbeispiele.* Hanser, München, 2009.

[SP09] Shneiderman, Ben und Catherine Plaisant: *Designing the User Interface: Strategies for Effective Human-Computer Interaction.* Pearson Addison Wesley, Boston, 5 Auflage, April 2009.

[UKN+08] Ullmann, Markus, Dennis Kügler, H. Neumann, S. Stappert und Vögeler M.: *Password Authenticated Key Agreement for Contactless Smart Cards.* In: *Workshop on RFID Security*, Budapest, 2008.

[vdV90] Veer, Gerrit van der: *Human Computer Interaction, Learning, Individual Differences and Design Recommandation.* Alblasserdam: Offsetdrukkerij, 1990.

[VM02] Viega, John und Gary McGraw: *Building Secure Software.* Addison-Wesley, Boston, MA, 2002.