

Entwicklung eines mobilen Software-Assistenten zur Unterstützung der vernetzten Pflege

Master-Arbeit
von

Thomas Knapp

An der Fakultät für Wirtschaftswissenschaften
Institut für Informationswirtschaft und -management (IISM)
Information & Market Engineering

Gutachter:	Prof. Dr. rer. pol. Christof Weinhardt
Betreuender Assistent:	Dr. Henner Gimpel
Betreuende Mitarbeiter FZI:	Bruno Rosales Saurer
	Mathias Schmon
	Imanol Bernabeu

31. März 2012

Inhaltsverzeichnis

Abbildungsverzeichnis	iii
Tabellenverzeichnis	iv
Abkürzungsverzeichnis	vi
1. Einleitung	1
2. Ziele der Arbeit	3
3. Analyse der Anforderungen und daraus abgeleitete Entwurfsentscheidungen	6
3.1. Analyse der Anforderungen	6
3.2. Entwurfsentscheidungen des Gesamtsystems auf Basis der erhobenen Anforderungen	10
4. Basisanwendung und Plug-Ins zur Realisierung der identifizierten Anforderungen	15
4.1. Aufgaben und Struktur der Basisanwendung	15
4.2. Einbinden von Plug-Ins in die Basisanwendung	19
4.3. Das Plug-In "Kontakte"	20
4.4. Das Plug-In "Touren"	20
4.5. Das Plug-In "Hilfe"	20
5. Evaluation	21
5.1. Evaluation von Nutzbarkeit, Funktionsumfang und Akzeptanz anhand einer Zielgruppenschulung	21
5.1.1. Vorwissen der Teilnehmer und vorbereitende Maßnahmen	21
5.1.2. Ermittlung des Status Quo - Ablauf und Schlussfolgerungen	21
5.1.3. Anforderungen an CareNet <i>mobile</i> aus Sicht der Zielnutzergruppe	21
5.2. Evaluation der technischen Umsetzung der mobilen Anwendung	21
5.3. Fallstudie Pneumonie	21
6. Fazit und zukünftige Entwicklungs- und Einsatzmöglichkeiten	22
7. Erklärung	23

Anhang	24
A. Kernanforderungen an eine mobile Anwendung zur Dokumentation ärztlich übertragener Tätigkeiten	24
B. Liste grundsätzliche Entwurfsentscheidungen CareNet <i>mobile</i>	25
Literaturverzeichnis	27

Abbildungsverzeichnis

2.1. Zielkomponenten der Arbeit	4
3.1. Verwendete Technologien in der Gesamtarchitektur	11
3.2. Hybride Anwendungsentwicklung mit PhoneGap	13
4.1. Struktur der grafischen Oberfläche von <i>CareNetmobile</i>	16
4.2. Vergleich zweier Beispielimplementierungen von Java und JavaScript	17
4.3. Überprüfung der Benutzerdaten in <i>CareNetmobile</i>	18

Tabellenverzeichnis

Abkürzungsverzeichnis

VERAH	Versorgungsassistentin in der Hausarztpraxis
MFP	Medizinische Fachpflegekraft
DRK	Deutsches Rotes Kreuz
GPS	Global Positioning System
WLAN	Wireless Local Area Network
FZI	Forschungszentrum Informatik
REST	Representational State Transfer
URI	Uniform Resource Identifier
HTTP	Hypertext Transfer Protocol
XML	Extensible Markup Language
JSON	Java Script Object Notation
App	Application
ID	Identifier

1. Einleitung

Nach Schätzungen des Statistischen Bundesamtes steigt der Anteil der über 65jährigen in der Bundesrepublik Deutschland von 19% im Jahr 2005 auf 30% im Jahr 2050 [Pre06] bei gleichzeitig steigender Lebenserwartung. Dieser demographische Wandel birgt große Herausforderungen für das Gesundheitswesen, da eine größere Anzahl an Patienten mit altersbedingten Krankheiten versorgt werden muss, die häufig multimorbide und immobil sind.

Insbesondere Hausarztpraxen als zentrale Anlaufstellen für Beschwerden im Alter stehen vor neuen Aufgaben. Wo früher nur Heilkunde angewandt wurde, werden heute beispielsweise auch koordinative Aufgaben zwischen Fachärzten, Pflegediensten und Apotheken übernommen. Mit einer gestiegenen Anzahl an pflegebedürftigen Personen wird sich dieser Trend weiter verstärken. Hinzu kommt, dass häufigere Immobilität der älteren Patienten von den Hausärzten mehr zeitaufwendige Hausbesuche abverlangen wird. Dem gegenüber steht eine sinkende Anzahl an Hausärzten, da auch deren Altersstruktur vom demographischen Wandel geprägt ist. Nach Schätzungen wird die Anzahl der Hausärzte bis 2020 bundesweit um etwa 7000 zurückgehen, was einem Anteil von 13.3% entspricht ([Kop10, S. 65]). Überdurchschnittlich hoch wird der Rückgang an Hausärzten in den ländlichen Regionen der neuen Bundesländer ausfallen ([Kop10, S. 52 ff.]). Der Rückgang der Hausärzte wird somit bundesweit zu signifikanten Versorgungsengpässen führen, die kurzfristig nicht durch junge Ärzte ausgeglichen werden können.

Um auch in Zukunft eine ausreichende Versorgung von Patienten gewährleisten zu können, wurde vom Gemeinsamen Bundesausschuss eine Richtlinie herausgegeben, die Rahmenbedingungen formuliert, in denen ärztliche Tätigkeiten auf Berufsangehörige der Alten- und Krankenpflege übertragen werden können [ric12]. Diese Maßnahme soll Hausärzte bei Tätigkeiten entlasten, für die nicht unbedingt ärztliche Expertise notwendig ist. Insbesondere die Anzahl der vom Arzt durchzuführenden Hausbesuche könnte sich hierdurch verringern. Ärztliche Tätigkeiten können dabei sowohl an Arzthelfer in einer Praxis, Mitarbeiter eines ambulanten Pflegedienstes, Pfleger in einem Alten- und Pflegeheim sowie Gesundheits-

und Krankenpflegern in einem Krankenhaus delegiert werden. Vor der Herausgabe der Richtlinie im März 2012 wurde bereits in verschiedenen Modellprojekten der Nutzen von Delegation evaluiert. Mit dem Modellprojekt AGnES [vdBMH⁺09] beispielsweise ließ sich belegen, dass durch Delegation positive Effekte bzgl. Effizienz und Qualität der ärztlichen Leistungen erzielt werden können. Um diese Effizienz- und Qualität zu erreichen, benötigen die Berufsangehörigen der Alten- und Krankenpflege allerdings gewisse Zusatzqualifikationen, die im Rahmen verschiedener Weiterbildungsprogramme erworben werden können. Beispiele hierfür sind VERAH ([fhFiDHle08] und [fhFiDHleV08]) oder MFP [QUELLE]. In einem ca. 200stündigen Fortbildungsprogramm werden unter anderem Kenntnisse aus den Bereichen Case Management, Gesundheits-, Praxis- und Wundmanagement sowie IT vermittelt.

Das Gesamtziel des Gesetzgebers ist, eine integrierte Versorgung von Pflegebedürftigen zu erreichen, bei der die Parteien des Gesundheitssystems (Ärzte, Pflegekräfte, Krankenkassen) intensiv zusammenarbeiten sollen. Eine enge Zusammenarbeit bedeutet einen hohen Koordinationsaufwand und damit verbunden ein hohes Maß an Informationsaustausch. Um das volle Potential der Delegation ärztlicher Leistungen ausschöpfen zu können, müssen entsprechende EDV-Systeme unterstützend eingesetzt werden. Zwar werden auch heute schon vielfältige Verwaltungssysteme zur Dokumentation von entweder ärztlichen Tätigkeiten oder Pflegetätigkeiten eingesetzt, jedoch ist das Angebot an Software zur Unterstützung einer Zusammenarbeit zwischen Ärzten und Pflegedienstleistern gering. Gerade wenn es darum geht, vor Ort beim Klienten Vorgänge zu dokumentieren, wird häufig zunächst auf Papier dokumentiert und die Information erst zu einem späteren Zeitpunkt in ein EDV-System übertragen. Diese Medienbrüche kosten Zeit und in der Folge Geld und die Fehlerquellen bei der manuellen Übertragung von Informationen sind vielfältig.

Diese Master-Arbeit setzt an der Schnittstelle zwischen Ärzten und Berufsangehörigen der Alten- und Pflegeberufe an.

[Beschreibung der einzelnen Kapitel.]

2. Ziele der Arbeit

Ziel dieser Master-Arbeit ist es, ein elektronisches System zur Unterstützung der Zusammenarbeit von Ärzten und Berufsangehörigen der Alten- und Pflegeberufe, die eine Zusatzqualifikation zur Ausübung ärztlicher Tätigkeiten haben, zu entwerfen und zu implementieren. Einerseits soll eine Delegation von Aufgaben möglich sein, andererseits sollen die ausgeführten Tätigkeiten elektronisch dokumentiert werden können. Da die ärztlichen Tätigkeiten vornehmlich vor Ort beim Klienten durchgeführt werden, lassen sich im Kern zwei notwendige Komponenten identifizieren.

Abbildung 2.1 zeigt eine stilistische Darstellung der Komponenten der Arbeit. Die Basis bildet ein zentrales System, in welchem die Daten zu Klienten, auszuführenden und ausgeführten Tätigkeiten verwaltet werden. Daneben gibt es eine Software auf einem mobilen Endgerät, die es ermöglicht, Informationen über auszuführende Tätigkeiten abzurufen und nach der Ausführung die Ergebnisse zu dokumentieren. Verbunden werden beide Systeme über eine Middleware, die Anfragen an das Verwaltungssystem über verschiedene Schnittstellen ermöglichen soll. Da die Kommunikation zwischen der mobilen Anwendung und dem Verwaltungssystem in dieser Arbeit aber lediglich über einen Schnittstellentyp läuft, dient die Middleware vor allem als Platzhalter, um zu einem späteren Zeitpunkt weitere Schnittstellen implementieren zu können. Die Implementierung weiterer Schnittstellen ist nicht Teil dieser Arbeit.

Idee der direkten elektronischen Dokumentation beim Klienten ist die Vermeidung von Medienbrüchen. Anstatt zunächst auf Papier zu dokumentieren und diese Informationen zu einem späteren Zeitpunkt zu übertragen, sollen die Informationen über eine internet-gestützte Schnittstelle direkt an das Verwaltungssystem übertragen werden. Ziel ist eine Einsparung der Zeit, die dafür benötigt wird, die schon dokumentierten Informationen noch einmal manuell zu übertragen. Außerdem wird eine Reduzierung möglicher Fehlerquellen angestrebt, um die Qualität der Dokumentation zu erhöhen.

Die Entwicklung der mobilen Anwendung ist Kern der Arbeit. Vorgabe ist, Tablet-PCs zur Dokumentation einzusetzen, da sie den momentan neuesten Stand der Technik darstellen. Welche Plattform (z.B. Android oder iOS) und welche Hardware genau eingesetzt wird,

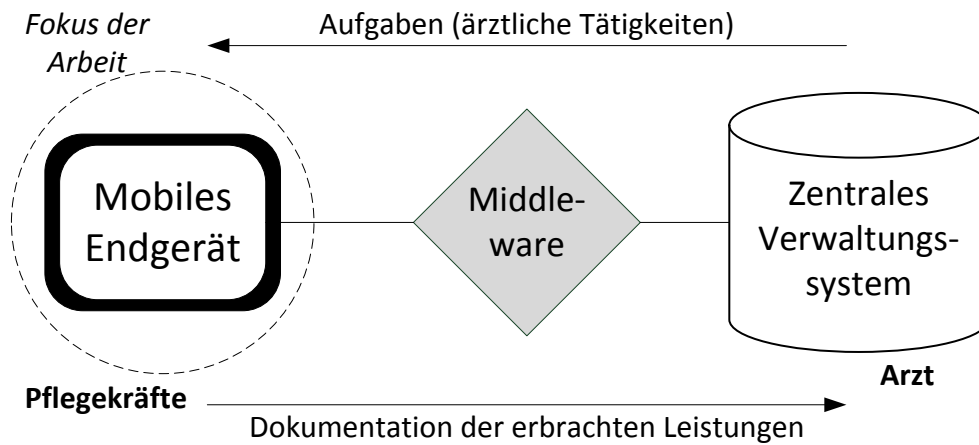


Abbildung 2.1.: Zielkomponenten der Arbeit

ist offen und wird sich aus den zu ermittelnden Anforderungen ableiten.

Der Entwicklungsprozess gliedert sich in eine Anforderungsanalyse, die Implementierung und eine Evaluation der entstandenen Software. Es findet vorerst eine auf Literatur basierende Anforderungsanalyse sowie eine Priorisierung nach technischen und zeitlichen Kriterien statt. Die Implementierung konzentriert sich zunächst auf die Umsetzung der wichtigsten Anforderungen, um einen ausreichend reifen Stand der Software zu erreichen. Die Evaluation gliedert sich **drei (ja?)** Teile. Nach etwa der halben Bearbeitungszeit wird ein erster Prototyp der Software (auf Basis der in der Literatur gefundenen Anforderungen entwickelt) im Rahmen einer Schulung zur Medizinischen Fachpflegekraft (MFP) den Teilnehmern vorgestellt. Hier sollen Akzeptanz, Benutzbarkeit sowie erwarteter Nutzen der Software evaluiert werden. Die hier gesammelte Resonanz wird in die weitere Entwicklung einfließen.

Neben den fachlichen Anforderungen soll die technische Umsetzung der Software evaluiert werden. Hierzu wird die verwendete Architektur nach Kriterien der ISO-25000 (Software Engineering) **QUELLE** geprüft.

Im letzten Teil der Evaluation wird überprüft, ob ein typischer Prozess **[oder eine Leistung]** aus der Richtlinie zur Delegation ärztlicher Leistungen mit der entstandenen Software delegierbar und dokumentierbar ist.

Das zentrale Verwaltungssystem wird in dieser Arbeit nicht von Grund auf entwickelt, sondern basiert auf einer im Projekt VitaBit **QUELLE** entwickelten Software-Lösung zur Dokumentation von Pflegeleistungen. Bereitgestellt wird es von der nubedian GmbH [nG13], die außerdem Unterstützung bei der Anpassung des Systems an die Anforderungen der Dokumentation ärztlicher Leistungen anbietet, da das System momentan lediglich zur Dokumentation von Pflegeleistungen gedacht ist. Im Laufe der Entwicklung der mobilen Anwendung sollen notwendige Anpassungen des zentralen Verwaltungssystem identifiziert und dokumentiert werden.

Neben der nubedian GmbH wird der Entwicklungsprozess dieser Arbeit von Mitarbeitern der DRK Residenz Bad Friedrichshall [Res13] unterstützt. Hier werden Pflegekräfte mit einem an VERAH angelehnten Fortbildungsprogramm zu Medizinischen Fachpflegekräften (MFP) ausgebildet. Um die Terminologie in dieser Arbeit zu vereinfachen, wird künftig anstatt von "Angehörigen der Alten- und Krankenpflege" oder "Medizinischen Fachpflegekraft" stellvertretend für alle Fortbildungsmöglichkeiten zur Erfüllung der Richtlinie nach § 63 Abs. 3c SGB V lediglich die Bezeichnung "Pflegekraft" verwendet.

3. Analyse der Anforderungen und daraus abgeleitete Entwurfsentscheidungen

In diesem Kapitel werden die aus Literatur und vorangegangenen Projekten hervorgegangenen Anforderungen an eine mobile Anwendung zur Dokumentation von ärztlich übertragenen Tätigkeiten vorgestellt. In Abschnitt 3.1 werden funktionelle Anforderungen beschrieben und technische Anforderungen daraus abgeleitet. Abschnitt 3.2 stellt Entwurfsentscheidungen vor, die zur Realisierung der Anwendung getroffen wurden.

3.1. Analyse der Anforderungen

Zu Beginn der Bearbeitungszeit der Master-Arbeit konnte kein Termin mit den eigentlichen Endnutzern der mobilen Anwendung vereinbart werden, um eine direkte Anforderungserhebung durchzuführen. Aufgrund dessen musste zunächst eine Anforderungsanalyse auf Basis von Literatur, aktuellen und vorangegangenen Arbeiten ergänzt um Gespräche mit sachkundigen Mitarbeitern von Kranken- und Pflegeeinrichtungen durchgeführt werden. Da die Grundlage für die Möglichkeit, ärztliche Tätigkeiten an speziell geschulte Pflegekräfte zu übertragen, die Richtlinie des Gemeinsamen Bundesausschusses basierend auf § 63 Abs. 3c SGB V ist, kann die Dokumentation der hier aufgelisteten übertragbaren Leistungen als Basisanforderung gesehen werden. Ein Problem ist aber, dass aus der Auflistung der Tätigkeiten nicht die zu dokumentierenden Daten hervorgehen. Diese konnten in der Literatur jedoch nicht im Detail ermittelt werden, da einige Leistungen medizinisches Fachwissen über deren Erbringung erfordern. Deswegen wurde zunächst angenommen, dass in der Regel eine Bestätigung, ob eine Leistung erbracht wurde, ausreicht, und sonnst nur die Dokumentation von Blutdruck und Puls als wertbasierte Leistungen notwendig ist, sowie das Fotografieren von Wunden. Weitere zu dokumentierende Werte konnten erst bei einem Treffen mit den Endnutzern ermittelt werden. Die Ergebnisse werden in Kapitel 5.1 vorgestellt.

Das Ausbildungsprogramm zu einer Medizinische Fachpflegekraft (MFP) nimmt keine Einschränkung des Arbeitsgebietes vor. Eine Pflegekraft kann somit sowohl stationär in einer Pflegeeinrichtung arbeiten, als auch im ambulanten Dienst tätig sein, bei der die jeweiligen Klienten besucht werden. Die mobile Anwendung ist folglich so zu gestalten, dass beide Einsatzbereiche abgedeckt werden können. Es ist zu erwarten, dass die zu erbringenden Leistungen im stationären und im ambulanten Bereich die gleichen sein werden, jedoch können sich Unterschiede bei den technischen Anforderungen oder der Terminologie ergeben. Wird im Folgenden keine explizite Unterscheidung zwischen ambulanten und stationären Anforderungen genannt, so gilt eine Anforderung für beide Bereiche.

Jede ärztlich übertragene Tätigkeit kann direkt einer Person zugeordnet werden. Um eine Interaktion mit der Person zu ermöglichen, müssen bestimmte Daten über diese verfügbar sein. Hierzu gehören Stammdaten wie Name, Adresse, Anschrift und Alter, medizinische Daten über aktuell verabreichte Medikamente, Vorerkrankungen oder Allergien sowie Angaben zu Angehörigen, dem zuständigen Hausarzt und einer eventuellen Pflegebedürftigkeit. Aus der Notwendigkeit der Verfügbarkeit persönlicher Daten ergibt sich die Anforderung, diese Daten ausreichend zu schützen und nur dem jeweiligen Pflegepersonal zur Verfügung zu stellen. Folglich muss eine Authentifizierungsfunktion in Form einer Anmeldung mit Benutzername und Passwort (Log-In) implementiert werden. Ein Log-In ermöglicht es außerdem, der angemeldeten Pflegekraft nur die Informationen anzuzeigen, die sie für die Ausführung ihrer Tätigkeit benötigt (Autorisierungsfunktion). So kann ausgeschlossen werden, dass Informationen über Klienten verfügbar sind, die gerade nicht behandelt werden.

Es wird angenommen, dass Pflegekräfte in Schichten arbeiten, da eine Versorgung von Klienten Tag und Nacht gewährleistet sein muss. Weiter wird angenommen, dass jede Pflegekraft während einer Schicht für eine bestimmte Anzahl von Klienten zuständig ist, für die eine Menge vordefinierter Leistungen erbracht werden muss. Zu Beginn jeder Schicht bekommt eine Pflegekraft einen Plan ausgehändigt, in dem die abzuarbeitenden Stationen und die entsprechenden Leistungen aufgelistet sind. In diesem Plan muss jede Leistung dokumentiert werden (vgl. [Har10, S. 18 ff.]). Eine Leistung wird entweder durch das Setzen eines Hakens dokumentiert (Bestätigung) oder durch die Eintragung von Messwerten, beispielsweise im Falle der Erfassung von Blutdruck und Puls. Ziel der mobilen Anwendung ist es, dass sowohl die Übermittlung des Tagesplanes pro Pflegekraft als auch die Dokumentation der Leistungen elektronisch erfolgt. In einer Übersicht über alle Klienten/Stationen sollte direkt erkennbar sein, ob bereits alle Leistungen einer Station erbracht wurden, ohne die Einzelleistungen kontrollieren zu müssen.

Konnte eine Leistung nicht oder nicht vollständig erbracht werden, muss es die Möglichkeit geben, dies zu dokumentieren. Neben einer Markierung der Leistung muss ein Feld für eine Begründung (Freitext oder mit vordefinierten Bausteinen) der Nichterbringung der Leistung verfügbar sein. In der Übersicht über die Stationen einer Schicht sollte neben den Zuständen "Erledigt" und "Offen" außerdem der Zustand "Unvollständig abgeschlossen" ersichtlich sein, um zu zeigen, dass zwar alle Leistungen bearbeitet wurden, jedoch nicht alle erbracht werden konnten.

Bei der manuellen Dokumentation kann es immer wieder zu Fehlern kommen. Beispielsweise wenn Werte falsch eingetragen oder Leistungen bestätigt werden, die eigentlich noch nicht vollständig erfüllt sind. Da diese Fehler auch mit einer elektronischen Dokumentation nicht ausgeschlossen werden können, müssen einmal dokumentierte Leistungen stornierbar und korrigierbar sein, ohne dass im zentralen System eingegriffen werden muss. Um einer Verfälschung von Dokumentationsergebnissen durch nachträgliche Veränderungen beispielsweise von Messwerten entgegen zu wirken, sollte protokolliert werden, wer zu welchem Zeitpunkt eine Änderung vornimmt. Diese Protokollfunktion muss zur vollständigen Überwachung auch im zentralen System verfügbar sein und mit der mobilen Anwendung synchron gehalten werden.

Vor allem im ambulanten Bereich kann es immer wieder vorkommen, dass Klienten in Gegenden besucht werden müssen, in denen kein mobiles Internet verfügbar ist. Da der Datenaustausch zwischen mobiler Anwendung und zentralem Verwaltungssystem über eine Internetverbindung geschieht, wäre somit die Arbeit mit dem mobilen System nicht möglich. Darauf zu setzen, dass beim Klienten vor Ort eine lokale Internetverbindung genutzt werden kann, ist auch keine praktikable Lösung, da die Wahrscheinlichkeit für Probleme mit der Einrichtung und Verfügbarkeit hoch ist. Vor Ort beim Klienten anzukommen und festzustellen, dass der Plan der zu erbringenden Leistungen wegen Verbindungsproblemen nicht verfügbar ist, führt zu nicht hinnehmbaren Einschränkungen, da so eventuell wichtige medizinische Tätigkeiten nicht durchgeführt werden. Notwendig ist deshalb, den Plan für eine oder mehrere Schichten sowie alle relevanten Informationen zu Personen (Klienten, Angehörigen, Ärzten) vor dem Beginn einer Schicht lokal auf dem mobilen Endgerät zu speichern und somit auch ohne Internet verfügbar zu haben. Lediglich Aktualisierungen müssten so während einer Schicht nachgeladen werden. Wird das initiale Laden von Informationen über ein lokales Netzwerk durchgeführt, spart dies außerdem Datenvolumen ein, welches bei Verträgen mit Internet über das Mobilfunknetz meist sehr begrenzt ist (*Quelle notwendig?*).

Wichtig bei der Dokumentation von Wunden ist neben einer textuellen Beschreibung eine bildliche Erfassung. Dies könnte mit einer mitgeführten Digitalkamera erfolgen, jedoch besteht dann das Problem, dass die aufgenommenen Bilder nach vollendeter Schicht manuell an das zentrale System übertragen werden müssen und den richtigen Patienten zugeordnet. Dies kostet verhältnismäßig viel Zeit und kann zu Fehlern bei der Zuordnung führen. Wünschenswert ist deswegen eine Funktion in der mobilen Anwendung, die es ermöglicht, direkt über eine interne Kamera Wundfotos aufzunehmen und diese an das zentrale System zu übertragen.

Eine ähnliche Anforderung wurde bei der Erfassung der Vitalwerte wie Puls und Blutdruck identifiziert. Neben der manuellen Messung und Eintragung der ermittelten Werte gäbe es die Möglichkeit, ein automatisches Messgerät via Bluetooth an das mobile Endgerät anzuschließen und damit die Messung und Eintragung vollständig zu automatisieren. Dass dies grundsätzlich möglich ist, wurde im Projekt *Stroke Manager* des Forschungszentrums Informatik (FZI) gezeigt [GMR12]. Die automatische Erfassung würde Fehler bei der manuellen Eintragung von Messwerten würden somit ausgeschlossen. Da die Möglichkeit

besteht, dass das automatische Messgerät ausfällt, muss eine manuelle Eintragung jedoch trotzdem möglich sein.

Das Projekt VitaBit befasste sich, ähnlich wie diese Arbeit, mit der Dokumentation von Pflegeleistungen ambulanter Dienste. Auch hier musste einer Pflegekraft ein Tages-/Tourenplan angezeigt werden. Da die Leistungen ambulanter Dienste ausschließlich vor Ort beim Klienten erbracht werden, die häufig allein leben und nur noch eingeschränkt mobil sind, hat die Pflegekraft in der Regel einen Schlüssel zu den Wohnräumen des Klienten. Die Schlüssel aller Klienten werden meist an einem Bund zusammengefasst und mit Nummern zur Zuordnung versehen. Damit die Pflegekraft nicht erst lang nach dem passenden Schlüssel suchen muss, war eine Anforderung, dass die Schlüsselnummer direkt in der Übersicht der Stationen auftaucht. Da die in dieser Arbeit entwickelte Arbeit ebenfalls für den ambulanten Pflegedienst gedacht ist, wird diese Anforderung auch hier übernommen. Im Falle der stationären Betreuung von Klienten soll anstatt einer Schlüsselnummer die Zimmernummer in der Einrichtung angezeigt werden.

Die Zielgruppe, für die die mobile Anwendung entwickelt wird, sind Angehörige der Alten- und Pflegeberufe. Da es sich bei dieser Gruppe in der Regel um wenig IT-affine Personen handelt, muss die Anwendung übersichtlich strukturiert werden und intuitiv bedienbar sein, damit die Hürden der Benutzung durch die Bedienbarkeit möglichst gering sind. Ziel ist es, dass das grundsätzliche Wissen über die Bedienung von Tablet-PCs sowie das Fachwissen über die Dokumentation der Leistungen ausreicht, um die mobile Anwendung zu verwenden. Der Aufwand für Schulungen soll so minimal gehalten werden. Konzeptionell soll eine Orientierung an den graphischen Benutzerschnittstellen von iOS und Android stattfinden, den Betriebssystemen für mobile Geräte von Apple und Google.

Eine Funktionalität, die in erster Linie für ambulante Pflegedienste interessant ist, ist die Möglichkeit die Adresse eines Klienten auf einer Karte anzeigen zu lassen. Alternativ könnte Global Positioning System (GPS) könnte auch die aktuelle Position relativ zur Adresse des Klienten angezeigt werden, um eine Routenplanung durchzuführen und somit Fahrzeiten besser abschätzen zu können. Hierzu ist allerdings eine aktive Internetverbindung notwendig. Eine vollständige Navigation ist nicht geplant.

Um eine Reduzierung der elektronischen Geräte zu erreichen, die eine Pflegekraft mitführen muss, ist eine Funktion zum Führen von Telefongesprächen hilfreich. Einerseits könnten normale Telefongespräche beispielsweise mit der Zentrale, Angehörigen oder Ärzten geführt werden, andererseits wäre es aber auch möglich, einen Notruf-Knopf in die Anwendung zu integrieren, um in Notsituationen direkt Hilfe anfordern zu können. Beschränkt wird die Funktionalität zwar durch die Notwendigkeit der Verfügbarkeit eines Mobilfunknetzes, jedoch kann diese Einschränkung ebenso beim Mitführen eines separaten Mobiltelefons.

Von der bisher beschriebenen Menge an Funktionen ist lediglich die automatische Erfassung von Blutdruck und Puls mittels eines per Bluetooth angeschlossenen Messgerätes durch manuelle Erfassung substituierbar. Alle anderen Anforderungen müssen auf Basis der in der Literatur gewonnen Erkenntnisse umgesetzt werden, um einen sinnvollen Einsatz der mobilen Anwendung überhaupt erst zu ermöglichen.

Aus den beschriebenen funktionalen Anforderungen an die mobile Anwendung ergeben sich eine Reihe von technischen Anforderungen an das Endgerät. Grundanforderung ist eine Internetverbindung sowohl über ein lokales Wireless Local Area Network (WLAN), als auch über ein Mobilfunknetz, um Touren Daten und Aktualisierungen empfangen und senden zu können. Zur Dokumentation von Wunden per Foto muss eine Kamera mit einer ausreichend hohen Auflösung vorhanden sein, damit das medizinische Personal bei der Beurteilung der Wunden nicht durch schlechte Bildqualität eingeschränkt wird. Zur Unterstützung der intuitiven Bedienung sollte das mobile Gerät einen berührungsempfindlichen Bildschirm haben, was aber bei allen Endgeräten der Fall sein sollte, da dies eines der Hauptmerkmale von Tablet-PCs ist. Zur Bestimmung der Position einer Pflegekraft, um die Route zum nächsten Klienten zu berechnen, muss eine GPS-Verbindung möglich sein. Sollen Messgeräte für Blutdruck und Puls angeschlossen werden, muss eine Bluetooth-Verbindung verfügbar sein.

Da Alten- und Krankenpflege dem sozialen Bereich zuzuordnen ist und ein Ziel die Einsparung von Kosten ist, sollten die Endgeräte möglichst geringe Kosten verursachen. Obwohl grundsätzlich Tablet-PCs sowohl mit dem Android-Betriebssystem als auch mit iOS die technischen Anforderungen erfüllen, können Android Geräte im Schnitt zu einem deutlich niedrigeren Preis erworben werden [QUELLE]. Aus diesem Grund fällt die Entscheidung bei der Wahl der Plattform auf Android.

3.2. Entwurfsentscheidungen des Gesamtsystems auf Basis der erhobenen Anforderungen

Zur Dokumentation von Pflegeleistungen auf Tablet-PC-ähnlichen Endgeräten arbeitete Christina Hardt [Har10] in Ihrer Diplomarbeit mit der im Projekt VitaBIT ([Vit09] und [Kli10]) entwickelten Software. Diese bietet bereits Datenstrukturen für Tourenpläne und die Definition von zu erledigenden Tätigkeiten pro Station. Da die Anforderungen dieser Arbeit auch Touren- bzw. Schichtpläne benötigen, um diese auf einem Tablet-PC abrufen und anzeigen zu können, wird VitaBIT die Basis für das in Kapitel 2 beschriebene zentrale System darstellen. Da es im Kontext dieser Arbeit aber nicht mehr um die Dokumentation von Pflegeleistungen geht, sondern um die Übermittlung und Dokumentation ärztlicher Tätigkeiten durch Pflegepersonal, müssen Änderungen an VitaBIT vorgenommen werden. Um den Änderungen und dem Kontext der vernetzten Pflege auch namentlich gerecht zu werden, wurde das zentrale System in CareNet (Care = Pflege, Net = Netz) umbenannt. Auf die Änderungen an VitaBIT wird in dieser Arbeit nicht eingegangen, da diese von Mitarbeitern der nubedian GmbH durchgeführt werden.

Abbildung 3.1 zeigt die Komponenten des Gesamtsystems erweitert um die verwendeten Technologien, auf denen diese basieren. Die Kommunikation zwischen den Komponenten erfolgt via Internet mittels REST. REST steht für *Representational State Transfer* und

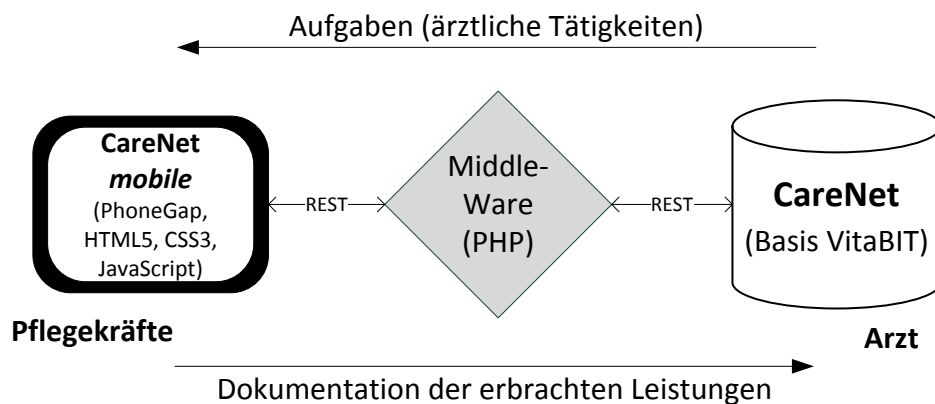


Abbildung 3.1.: Verwendete Technologien in der Gesamtarchitektur

stellt einen Architekturstil zur zustandslosen Interaktion dar. Im Gegensatz zu Sitzungen (Sessions), die von Servern während der Interaktion mit einem Benutzer aufwendig verwaltet werden müssen, werden bei dem REST-Prinzip vorberechnete Ressourcen, beispielsweise eine Webseite, über eine URI (Uniform Resource Identifier) referenziert und geladen. Alle Informationen über den Kontext einer Anfrage sind in der Anfrage selbst enthalten und es werden keine weiteren lokal beim Nutzer oder auf dem Server gespeicherten Informationen benötigt. Eine Kommunikation via REST bedeutet eine lose Verkopplung von vordefinierten Ressourcen, durch die ein Nutzer anhand gegebener URIs navigieren kann. Es werden lediglich die grundlegenden HTTP-Methoden PUT, POST, GET und DELETE verwendet, um Anfragen zu formulieren. Übertragen werden Ressourcen bzw. Datenobjekte in der Regel in Form einer XML- oder JSON-Datei [zMNS05, S. 19].

Die in dieser Arbeit zu entwickelnde mobile Anwendung stellt im Wesentlichen eine mobile Erweiterung von CareNet dar, aufgrund dessen der Name *CareNet mobile* gewählt wurde. Die Verwendung von Tablet-PCs als Hardware ist vorgegeben, da sie den neuesten Stand der Technik repräsentieren. Es wurde eine Neuentwicklung einer Lösung zur mobilen Dokumentation angestoßen, da in den vorangegangenen Projekten Probleme mit der verwendeten Hardware identifiziert werden konnten. Im Projekt VitaBIT wurden zur Dokumentation Smartphones der ersten Generation eingesetzt und in [Har10] wurde ein Panasonic CF-H1 eingesetzt, ein Vorläufer heutiger Tablet-PCs. Dieses Gerät hatte zwar bereits einen berührungsempfindlichen Bildschirm und eine Kamera, jedoch war die Leistungsfähigkeit nicht ausreichend, was zu Performanzproblemen führte. Außerdem konnte die Intuitivität der Bedienung nicht so umgesetzt werden, wie es beispielsweise unter Android oder iOS heute möglich ist. Das in VitaBIT verwendete Smartphone ist zwar sehr handlich und damit leicht mitzuführen, jedoch schränkt der kleine Bildschirm die Möglichkeiten der Darstellung ein, was gegen die Anforderung der Übersichtlichkeit spricht.

Jedes Betriebssystem von Tablet-PCs basiert auf einer eigenen Programmiersprache. Android basiert beispielsweise auf Java, iOS auf Objective-C und Windows Phone auf C#. Anwendungen für die Betriebssysteme können nur in den jeweiligen Sprachen entwickelt werden. Eine so entwickelte Anwendung wird als *nativ* bezeichnet. Dieser Umstand impliziert, dass bei der Entwicklung neuer Anwendungen vorher die Entscheidung getroffen werden muss, auf welcher Plattform diese bereitgestellt werden soll. Sollen mehrere Plattformen bedient werden, müssen mehrere Anwendungen mit der gleichen Funktionalität, aber anderer Quellcodebasis erstellt werden. Dies führt zu hohem zeitlichen Aufwand und ist kostenintensiv.

Eine Alternative ist die Erstellung von *WebApps*. Prinzipiell handelt es sich hierbei um Webseiten, jedoch sind diese für Touch-Oberflächen optimiert und sie sollen optisch an native Anwendungen erinnern. Da WebApps aber in einem Browser auf dem Tablet-PCs aufgerufen werden müssen, können sie nicht das Gefühl einer vollwertigen App (Application = Anwendung) bieten. Ungleich größer sind die Einschränkungen von WebApps bei der bereitstellbaren Funktionalität. Mit reinen Webtechnologien wie HTML5 oder JavaScript ist es nämlich nicht möglich, hardspezifische Funktionen wie die Kamera, den GPS-Sensor oder eine Bluetooth-Verbindung anzusprechen. Dies ist nur über die jeweilige native Programmiersprache möglich. Da in den Anforderungen aber eine Nutzung der Kamera sowie des GPS-Sensors gefordert wird, ist eine Implementierung von *CareNet mobile* als WebApp nicht möglich.

Ein Ansatz zur Lösung dieses Problems bietet das Projekt PhoneGap [Ado13]. Die Idee ist, in dem Gerüst einer nativen Anwendung, einem nativen Container, einen Browser zu platzieren, der Implementierungen in Webtechnologien wie HTML5, CSS3 und JavaScript interpretieren kann. Abbildung 3.2 zeigt den Zusammenhang der verschiedenen Technologien. Um den Zugriff auf endgerätspezifische Funktionen zu ermöglichen, bietet PhoneGap zur Kommunikation mit allen wichtigen Hardware-Funktionen JavaScript-Methoden an, die auf die jeweiligen Befehle in der nativen Programmiersprache abgebildet werden. Anstatt beispielsweise die Kamera unter Android direkt über Java anzusprechen, erfolgt der Aufruf in JavaScript, welcher von einer Java-Klasse verarbeitet und an die Hardware weitergereicht wird. PhoneGap verfolgt somit einen hybriden Ansatz aus nativer Entwicklung und der Nutzung von WebApps.

Die reine Nutzung von Webtechnologien und die Bereitstellung verschiedener nativer Container durch PhoneGap für die jeweils gewünschte Plattform ermöglicht eine Portabilität der Anwendung. Einmal entwickelt kann diese beispielsweise sowohl auf Android, als auch auf PhoneGap oder Windows Phone bereitgestellt werden. Diese Möglichkeit bringt Vorteile in Bezug auf Kosten und Zeit gegenüber der Entwicklung individueller nativer Anwendungen.

Als Anforderung an *CareNet mobile* wurde zwar zunächst nur die Bereitstellung auf Android-Endgeräten gefordert, jedoch ist nicht auszuschließen, dass in Zukunft auch die Nutzbarkeit auf iOS- oder Windows-Geräten gefordert wird. Da es mit der Verwendung von PhoneGap möglich ist, diese Option mit geringem Aufwand offen zu halten, wurde entschieden, *CareNet mobile* in als hybride Anwendung zu entwickeln. [Same Origin Policy reinnehmen?]

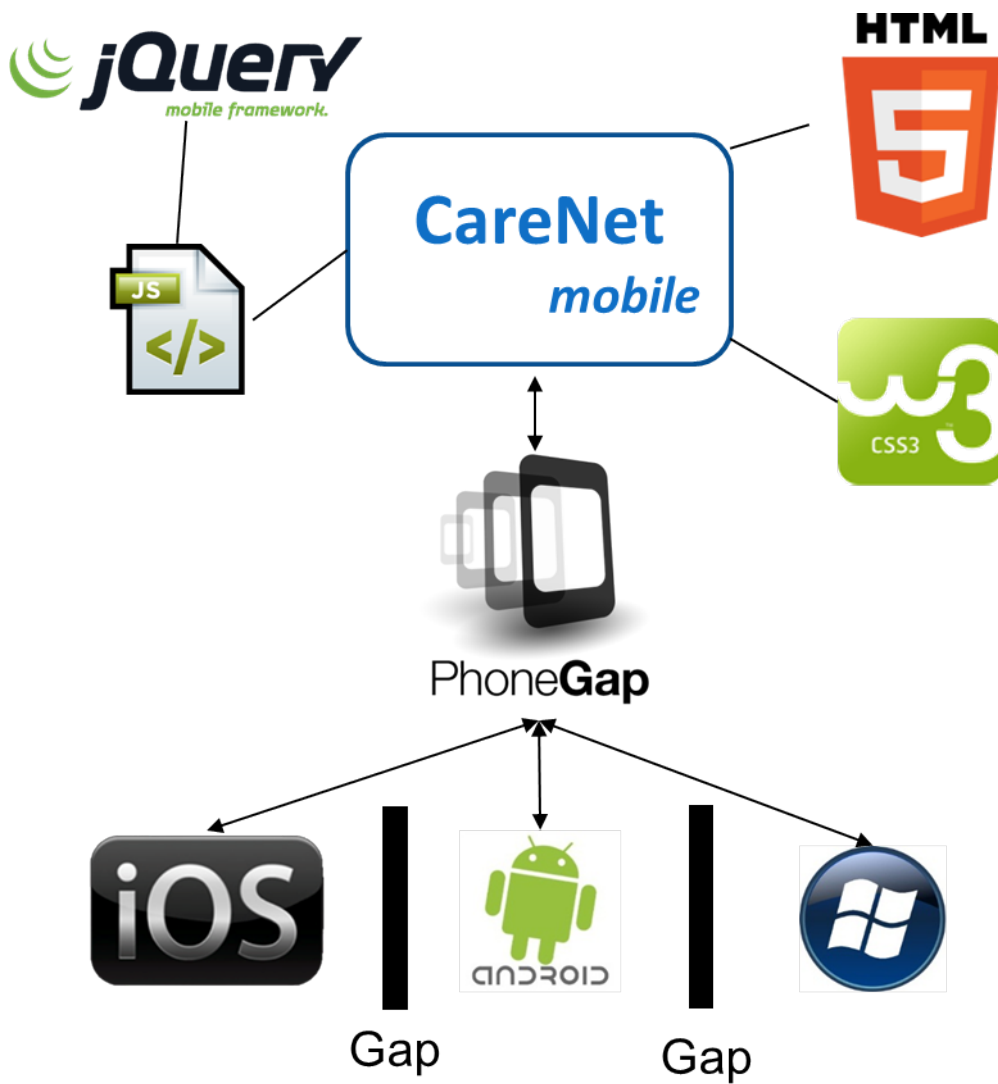


Abbildung 3.2.: Hybride Anwendungsentwicklung mit PhoneGap

Zusammen mit PhoneGap bietet sich die Verwendung bereits existierender JavaScript-Bibliotheken an, die zum Entwurf touch-optimierter Benutzeroberflächen entwickelt wurden. Zwei verbreite Bibliotheken sind jQuery mobile [jQu13] und Sencha Touch [Sen13]. Im Kontext dieser Arbeit wurde jQuery mobile aufgrund der Einfachheit in der Entwicklung und der Mächtigkeit der gebotenen Funktionalität gewählt.

In [Har10] hat sich gezeigt, dass sich einmal gestellte Anforderungen rasch ändern können und dass eine Anwendung so entworfen sein sollte, dass Anpassungen mit vertretbarem Aufwand durchgeführt werden können. Zwar ist momentan die Menge an zu dokumentierenden Tätigkeiten durch eine abgeschlossene Liste in Richtlinie § 63 Abs. 3c SGB V beschränkt, jedoch sind Änderungen in Zukunft möglich. Neben zusätzlichen Tätigkeiten könnten auch andere funktionale Erweiterungen hinzukommen. Um diesen Anforderungen gerecht zu werden, wird die Anwendung auf Basis einer Plug-In-Architektur entwickelt, angelehnt an die bewährte Architektur der Entwicklungsumgebung Eclipse [The13a]. Bei einer Plug-In-Architektur können Funktionalitäten nach dem Steckerprinzip (”plug-In” = einstecken) hinzugefügt und wieder entfernt werden. Im Idealfall sollten Änderungen an den Plug-Ins keine Auswirkungen auf das Gesamtsystem haben. Plug-Ins untereinander können jedoch Abhängigkeiten aufweisen. Wie die Plug-In-Architektur von CareNet *mobile* genau aussehen wird, wird in Kapitel 4 beschrieben.

Wie bereits in Kapitel 2 erwähnt, hat die Middleware in erster Linie eine Platzhalterfunktion, um den Zugang zu CareNet später um weitere Schnittstellen erweitern können. Da im Moment nur eine REST-Schnittstelle angeboten wird, die auch von CareNet *mobile* verwendet wird, wäre die Middleware zur Kommunikation nicht notwendig. Sie nimmt lediglich REST-Aufrufe an und leitet diese unbearbeitet weiter. Es wurde entschieden, die Middleware in PHP [The13b] zu implementieren, da es sich hierbei um eine mächtige und weit verbreite serverseitige Programmiersprache handelt. PHP wird von einem Großteil der Server-Hosting-Anbieter unterstützt und stellt somit keine Limitierung bei der Auswahl des Ortes dar, an dem die Middleware installiert wird.

4. Basisanwendung und Plug-Ins zur Realisierung der identifizierten Anforderungen

[Vorgeplänkel Kapitel]

4.1. Aufgaben und Struktur der Basisanwendung

Um einem Benutzer von *CareNetmobile* die Bedienung der Anwendung zu erleichtern, sollte ein einheitliches Gesamterscheinungsbild eingehalten werden, da ständige Strukturwechsel in der grafischen Oberfläche zu Unübersichtlichkeit führen können [Quelle]. Hierzu wurde die in Abbildung 4.1 gezeigte feste Struktur der grafischen Oberfläche entworfen. Oben links befindet sich das Logo der Anwendung, darunter ist ein Bereich für die Darstellung von Informationen wie das aktuell verwendete Plug-In oder der gerade behandelte Klient reserviert. Am oberen Rand rechts des Logos wurde eine Leiste zum Platzieren von Buttons zur Navigation eingefügt. Der größte Bereich ist zur Darstellung und Bearbeitung des eigentlichen Inhalts gedacht.

Die Strukturierung und Gestaltung der grafischen Darstellung erfolgt mittels HTML5 und CSS3, die Funktionalität wird mit JavaScript entwickelt. Gegenüber der Entwicklung einer nativen Anwendung für Android ergeben sich bei der Entwicklung mit JavaScript gewisse Schwierigkeiten. Während Java eine objektorientierte Sprache ist, bei der Typsicherheit und Kapselung von Funktionalität in Klassen zu den Eigenschaften gehören, ist JavaScript eine Scriptsprache, die keinerlei Typsicherheit und nur bedingte Kapselungsmöglichkeiten bietet. Ein wichtiger Unterschied ist, dass Java vor der Ausführung zunächst kompiliert, d.h. in eine maschinenverständliche Zwischensprache übersetzt werden muss, während JavaScript direkt vom Browser interpretiert werden kann. Dies impliziert, dass JavaScript-Code, selbst wenn er bereits auf einer Webseite ausgeführt wird, dem Nutzer im Klartext zur Verfügung steht, während der Java-Code nur bedingt und durch aufwendige

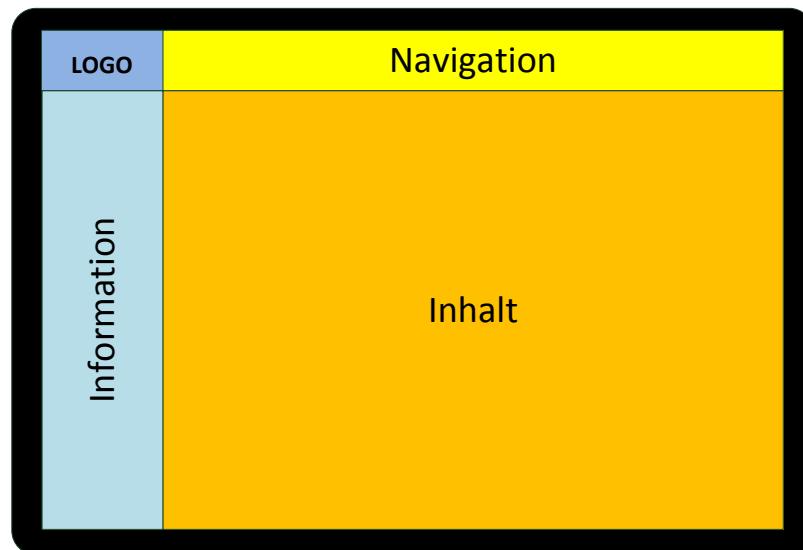


Abbildung 4.1.: Struktur der grafischen Oberfläche von CareNetmobile

Verfahren wieder dekompiert und somit im Klartext verfügbar gemacht werden kann. Dieser Eigenschaft muss sich ein Entwickler besonders im Bezug auf im Quellcode vermerkte sicherheitsrelevante Informationen bewusst sein.

Während der Entwicklung kann vor allem die fehlende Möglichkeit der Kapselung problematisch werden. Abbildung 4.2 zeigt Beispielimplementierungen in Java und JavaScript. Links wurde eine Klasse erstellt, die Methoden und Variablen kapselt. Wird eine Variable außerhalb einer Methode definiert (Abbildung 4.2: *ersteVariableJava*), ist diese in der gesamten Klasse verfügbar. Ist die Variable als *public* gekennzeichnet, kann auf sie über *Klassenname.variablenName* zugegriffen werden. Wird sie als *private* deklariert, ist sie nur innerhalb der Klasse verfügbar. Das gleiche gilt für Methoden. Wird eine Variable innerhalb einer Methode definiert (Abbildung 4.2: *zweiteVariable*), kann auch nur innerhalb dieser Methode auf die Variable zugegriffen werden.

In JavaScript sind grundsätzlich alle Methoden global verfügbar. Die Verfügbarkeit beschränkt sich hier nicht auf eine Klasse oder Datei, sondern auf eine einmal deklarierte Methode kann von allen anderen JavaScript- und HTML-Dateien, die gleichzeitig mit dieser Datei vom Browser geladen wurden, zugegriffen werden. Eine Klassendefinition kann nur in sofern erfolgen, dass einem Methodennamen der Name einer anderen vorher definierten Methode vorangestellt wird (Abbildung 4.2: *beispiel.methodeJS()*). Hiermit wird es möglich, einen Namensraum zu definieren, um Methoden einem bestimmten Kontext (hier: *beispiel*) zuzuordnen. Wird nun aber an einer anderen Stelle im Kontext der geladenen HTML- und JavaScript-Dateien der Name der Methode verwendet, kann der Interpreter, der Browser, keine eindeutige Zuordnung des Namens zu ausführbarem Quellcode mehr herstellen und es wird nichts ausgeführt.

Ebenso verhält es sich mit Variablen. Wird eine Variable mit dem Zusatz *var* außerhalb einer Methode definiert, ist sie global verfügbar. Das gleiche gilt, wenn eine Variable ganz

Java	JavaScript
<pre> Datei: beispiel.java class beispiel{ public string ersteVariableJava = ""; //Zugriff: beispiel.ersteVariableJava public void methodeJava(){ //Zugriff: beispiel.methodeJava(); string zweiteVariableJava = ""; //nur in Methode verfügbar } } </pre>	<pre> Datei: beispiel.js //Konstruktor und „Klasse“ function beispiel(){ ... } var ersteVariableJS = ""; //global beispiel.methodenJS = function(){ //Zugriff: beispiel.methodeJS zweiteVariableJS = ""; //global var dritteVariableJS = ""; //nur in Methode verfügbar } </pre>

Abbildung 4.2.: Vergleich zweier Beispielimplementierungen von Java und JavaScript

ohne Zusatz (Abbildung 4.2: *zweiteVariableJS*) definiert wird. Nur wenn eine Variable innerhalb einer Methode mit dem Zusatz *var* erstellt wird, kann sie nur innerhalb der Methode verwendet werden. Dieser Umstand kann zu massiven Problemen führen. Vergisst ein Entwickler beispielsweise nur ein einziges Mal den Zusatz *var* bei der Definition einer Laufvariable in einer Schleife (häufig mit dem Namen *i*), wird diese nun global geltende Variable alle anderen Schleifen, die ebenfalls die Laufvariable *i* verwenden, beeinflussen. Da dies kein syntaktischer, sondern ein semantischer Fehler ist, kann dieser nicht von automatisierten Fehlerfindungswerkzeugen entdeckt werden und es beginnt die sprichwörtliche Suche nach der Nadel im Heuhaufen. Derartige Umstände können sehr viel Zeit kosten.

Einzelne Plug-Ins sollen die in Abbildung 4.1 gezeigte Struktur nutzen, um Inhalte darzustellen. Damit dies möglich ist, müssen die einzelnen Bereiche, welche als HTML-DIV-Elemente erstellt wurden, eindeutig durch IDs gekennzeichnet werden. Da es in JavaScript nicht, wie in Java, möglich ist, Konstanten zu definieren, wurden in einer separaten JavaScript-Datei *CONSTANTS.js* alle IDs in Form von Variablen hinterlegt, welche den DIV-Elementen zugewiesen wurden. Der Nachteil von Variablen ist, dass diese während der Laufzeit aus den Plug-Ins heraus verändert werden können. Aus diesem Grund muss die Konvention eingehalten werden, zentral definierte Variablen nicht zu verändern, da dies Auswirkungen auf andere Plug-Ins haben könnte. Die Eigenschaft einer Plug-In-Architektur, dass Plug-Ins keinen Einfluss auf die Basisanwendung haben, kann somit nicht vollständig in JavaScript umgesetzt werden. Menschliche Fehler durch Nicht-Einhaltung der Konvention bleiben möglich.

Der Vorteil von zentral definierten Variablen ist, dass an einer zentralen, übersichtlichen Stelle nachgeschaut werden kann, wie eine ID lautet, und dass die wichtigen Strukturele-

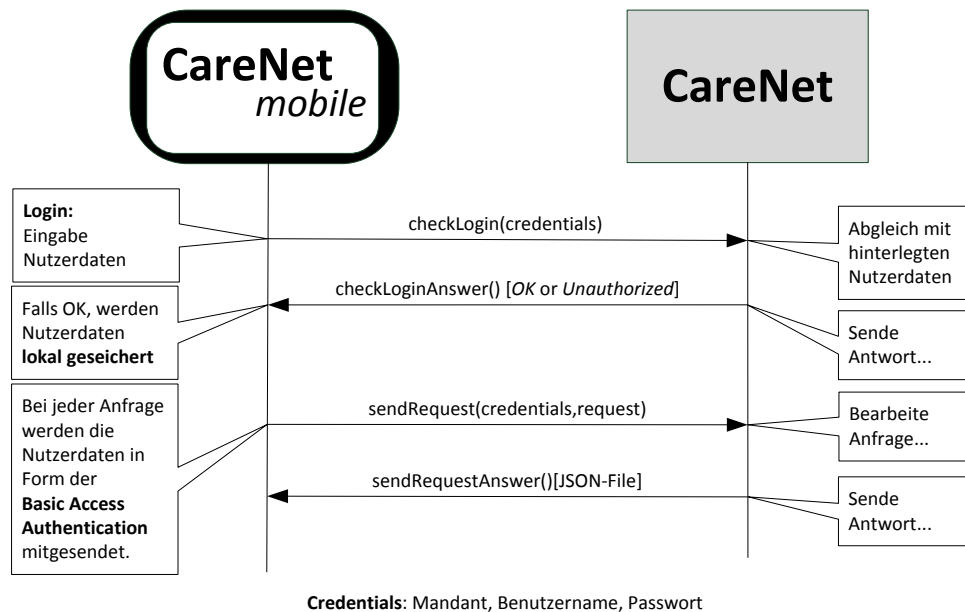


Abbildung 4.3.: Überprüfung der Benutzerdaten in CareNetmobile

mente für Informationen, Navigation und Inhalt zentral referenziert werden können. Die Datei *CONSTANTS.js* wird außerdem dazu genutzt, um alle sonstigen in der Implementierung verwendeten Zeichenketten als Variablen zu hinterlegen. Hierzu zählt beispielsweise die Server-Adresse der Middleware oder alle Arten von Meldungen an einen Nutzer. Dieses Verfahren erhöht die Wartbarkeit der Anwendung, da Änderungen an mehrfach verwendeten Zeichenketten nicht *mehrfach* umgesetzt werden müssen, sondern lediglich *einmal* in der zentralen Datei. Die Fehlerwahrscheinlichkeit durch vergessene Änderungen kann so verringert werden. Dieses Vorgehen ist angelehnt an die zentrale Verwaltung von *String Ressourcen* bei der Entwicklung nativer Android-Anwendungen [Goo13].

Eine Grundfunktion, die die Basisanwendung zur Verfügung stellt, ist ein Log-In, durch welchen eine Sicherung der persönlichen Daten und eine Identifizierung des Nutzers möglich sein soll.

- Grundsätzlich: Plug-In-Architektur angelehnt an Eclipse-Bsp [CHECK]
- Bestandteile: Grundanwendung (Basis), Plug-Ins [CHECK]
- Aufgaben Grundanwendung: Sicherheit (Login, Kommunikation), Verwaltung von Plug-Ins, Bereitstellen einer GUI-Struktur, zentrale Verwaltung lokaler Datenbanken (Methoden) [CHECK]
- Programmiereigenschaften von JavaScript ggü. Java [CHECK]
- Zentrale Verwaltung von IDs und sonstigen Strings [CHECK]
- Plug-Ins sollten keinen Einfluss auf Grundanwendung haben (Kapselung, aber komplette Trennung nicht möglich) [CHECK]

- Grafik Gesamtarchitektur [CHECK]
- Grundstruktur: jQuery mobile (Seitenaufbau, alles mit Divs, CSS, wie Webseite) [MMHHH..., NÜTZLICH?]
- Login
 - als Dialog (vorgeschaltete Seite)
 - Mandant, Nutzernamen, PW
 - Wg. REST-Prinzip: Nur einmalige Überprüfung, ob Daten korrekt sind, kein Aufbau einer dauerhaften Verbindung
 - Speicherung der Daten im lokalen Speicher im Klartext (HTML5/PhoneGap Key-Value-Speicher)
 - Senden der Daten bei jeder Anfrage (Beispielanfrage)
 - Authentifizierungsstring: mandant/nutzername:passwort
 - Middleware verschlüsselt String mit Base64 und sendet den an CareNet (kann einfach entschlüsselt werden)
 - Daten werden so unverschlüsselt gesendet, d.h. Verbindung an sich muss gesichert werden (https)
 - Bei Logout werden Daten einfach aus lokalem Speicher gelöscht
 - Sicherheitsproblem bei eingeschleuster Spyware (JS) über Browser? Nein, wg. Unabhängigkeit von PhoneGap-Browser/WebView und installierten anderen Browsern
- Struktur GUI

...

4.2. Einbinden von Plug-Ins in die Basisanwendung

...

Wie in Kapitel 3.2 beschrieben, wird *CareNetmobile* auf Basis einer Plug-In-Architektur entworfen. Abbildung ?? zeigt das grundsätzliche Konzept. Die Basisanwendung bietet wichtige Grundfunktionalitäten wie einen Log-In, Methoden zur sicheren Kommunikation mit CareNet, Methoden zur Verwaltung lokaler Datenbanken sowie eine Koordination der Plug-Ins. Die eigentliche Funktionalität, um mit *CareNetmobile* arbeiten zu können, wird ausschließlich über Plug-Ins hinzugefügt. Diese werden über eine vordefinierte Schnittstelle, in Abbildung ?? dargestellt als Ring, hinzugefügt.

- Plug-Struktur:
 - Namenskonventionen wg. JS global
 - unbegrenzte Anzahl an Plug-Ins

- Alle zugehörigen Dateien gekapselt in einem Ordner
 - Konfiguration eingebundener Plug-Ins (JSON)
 - Icon
 - JS-Datei (Problem der dynamischen Einbindung)
 - CSS-Datei
 - Versionsnummer, Konfig-Datei (Abhängigkeiten inkl. Versionsnummern)
 - Konstruktor, Methode zur Initialisierung
 - Arbeit mit Containern für GUI
 - Beschreibung der Abhängigkeiten zu Bibliotheken wie jQuery, jQuery mobile, PhoneGap (Redundanzen vermeiden)
 - Grafik zu Plug-In-Struktur
- Namenskonventionen
 - Vorgang des Einbindens
 - Probleme, Erweiterungsmöglichkeiten

4.3. Das Plug-In ”Kontakte”

Für alle Plug-Ins feste Beschreibungsstruktur:

- Begründung für Plug-In (aus welchen Anforderungen geht Plug-In hervor)
- Aufbau/Navigationsstruktur
- Zentrale Frage, die beantwortet werden muss: Welche Funktionalität geht aus welcher Anforderung hervor?

...

4.4. Das Plug-In ”Touren”

...

4.5. Das Plug-In ”Hilfe”

...

5. Evaluation

5.1. Evaluation von Nutzbarkeit, Funktionsumfang und Akzeptanz anhand einer Zielgruppenschulung

Evaluation der antizipierten Anforderungen aufgrund von Literaturrecherche und logischem Denken. Wichtiger Teil: Ergebnisse der Schulung der Alten- und Pflegekräfte vom 19.01.12 . Wie fanden die Teilnehmer die aufgrund der antizipierten Anforderungen entworfene App? Welche Änderungen wurden vorgeschlagen? Welche Erweiterungen sind notwendig? 〈 Nutzbarkeit, Funktionsumfang, Akzeptanz 〉

5.1.1. Vorwissen der Teilnehmer und vorbereitende Maßnahmen

- Eindruck, Vorwissen (inkl. CAS-Bogen)
- CareCM-Schulung (nur erwähnen, kein Mehrwert)
- Grundlagenvermittlung Tablet-PCs

5.1.2. Ermittlung des Status Quo - Ablauf und Schlussfolgerungen

[Orientierung an Fazit]

5.1.3. Anforderungen an CareNet*mobile* aus Sicht der Zielnutzergruppe

5.2. Evaluation der technischen Umsetzung der mobilen Anwendung

Evaluation der Anforderungen dahingehend, welche umgesetzt werden konnten und in welcher Qualität (inkl. der, die noch in der Schulung dazukamen). Vergleich mit Software-Entwicklungs-Standards (ISO xxx), Architekturprinzipien. Evaluation in zwei Teilen:

5.3. Fallstudie Pneumonie

Überprüfung der Anwendbarkeit der mobilen Anwendung auf einen Ablauf in der Praxis.

6. Fazit und zukünftige Entwicklungs- und Einsatzmöglichkeiten

Ausblick auf mögliche Erweiterungen:

- Erweiterbarkeit des Ansatzes ausgrund von Plug-In-Struktur herausstellen
 - Weitere Funktionen, um Komfort zu steigern (Prio C Anforderungen)
 - Schließen eines Falles beim Entfernen vom Einsatzort (damit keine überlangen Verweildauern gespeichert werden)
 - Login zu verschiedenen Instanzen von CareNet (Auswahl beim Login)
 - Unterschiedliche Verfügbarkeit von Plug-ins in Abhängigkeit des Rechten desjenigen, der sich anmeldet
 - Erweiterung um Sprachpakete durch zentrale Definition von Nachrichten
1. Kurzer Vergleich der nativen Entwicklung unter Android mit der hybriden Entwicklung (Wie hat sich die Entwicklung gestaltet?)
 - a) Typsicherheit
 - b) Namensräume (z.B. Gefahr von Überschneidung von Variablennamen und Methodennamen)
 - c) Unterstützung durch native Java-Bibliotheken vs. JavaScript-Bibliotheken
 - d) Einfachheit der Implementierung mit HTML/JavaScript
 - e) Vorteil der Portierbarkeit (Test wäre hier praktisch)

7. Erklärung

Ich versichere hiermit wahrheitsgemäß, die Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt, die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht und die Satzung des Karlsruher Instituts für Technologie (KIT) zur Sicherung guter wissenschaftlicher Praxis in der jeweils gültigen Fassung beachtet zu haben.

Karlsruhe, den 31. 03. 2012

Thomas Knapp

Anhang

A. Kernanforderungen an eine mobile Anwendung zur Dokumentation ärztlich übertragener Tätigkeiten

- Wie wurden die Anforderungen erhoben?
 - Richtlinie
 - Gespräche mit Raymond
 - Gespräche Corinna
 - Gesunder Menschenverstand
 - Wissen über Zielgruppe
 - Artikel Ärzteblatt
 - Diplomarbeit Christina Hardt
 - Projekt VitaBit (Schlüsselnummer, Tourenplan)
 - ...
- Kernanforderungen (Prio A)
 - Funktionalität
 - * Informationen über Klienten müssen verfügbar sein (Patientenakte) [CHECK]
 - * Persönliche Daten → Schutz durch Login-Funktion [CHECK]
 - * Dokumentation von Tätigkeiten aus Richtlinie muss möglich sein [CHECK]
 - * ambulante und stationäre Pflege abdecken! [CHECK]
 - * Dokumentierte Tätigkeiten müssen an zentrales System übertragbar sein [CHECK]
 - * Ablaufplan jeweils für eine MFP pro Tag/Schicht [CHECK]
 - * Übersicht, ob jeweilige Station schon vollständig abgearbeitet [CHECK]
 - * Es muss eine Tätigkeit als *nicht durchgeführt* markiert werden können + Begründung, falls etwas dazwischenkommt [CHECK]
 - * Einmal dokumentierte Werte sollten korrigierbar sein [CHECK]

- * Nachvollziehbarkeit von Änderungen [CHECK]
- * Daten "offline" auf dem Endgerät verfügbar, d.h. keine zwangsläufige Internetverbindung [CHECK]
- * Dokumentation von Wunden (Fotos) [CHECK]
- * Erfassung von Vitalwerten (Blutdruck, Puls), evtl. mittels automatischer Erfassung [CHECK]
- * Schlüsselnummer für Hausbesuche bzw. Zimmernummer bei stationären Behandlungen (aus VitaBIT) [CHECK]
- * Klar und übersichtlich strukturierte Anwendung, intuitive Bedienung (am besten Touch, Endnutzerwissen) [CHECK]
- * Für ambulanten Dienst eventuell Navigation mittels GPS [CHECK]
- * Telefongespräche? [CHECK]
- * ...
- Abgeleitete technische Anforderungen
 - * sozialer Kontext → günstige Hardware [CHECK]
 - * **Hardware:** mobiles Endgerät muss internetfähig sein (am besten über Mobilfunknetze für ambulanten Dienst), eine Kamera haben (Wunddokumentation), am besten Touch-Bedienung, darf trotzdem nicht so teuer sein, evtl. GPS, falls automatische Erfassung evtl. Bluetooth → Android-Tablet [CHECK]
 - * ...
- ...
- Wünschenswerte Eigenschaften, um Nutzungskomfort zu steigern (Prio B) [NEIN]
- Anforderungen, die über die reine Dokumentation oder dessen Unterstützung hinausgehen (Prio C, werden in der Arbeit nicht behandelt) [NEIN]

B. Liste grundsätzliche Entwurfsentscheidungen CareNetmobile

- Zentrales System: CareNet, basierend auf VitaBit [CHECK]
- Kommunikation mit REST; Was ist REST? Prinzipien? [CHECK]
- **CareNet mobile**
 - Name [CHECK]
 - Warum überhaupt Tablet? (Vgl. mit Diplomarbeit der anderen Studentin auf Nokia-Handy) [CHECK]
 - Nativ - WebApp - Hybrid [CHECK]

- PhoneGap, jQuery mobile [CHECK]
- Falls doch jemand Apple will, nach Mglk. suchen, wie das evtl. auch bedient werden kann [CHECK]
- Architektur (Plug-In-Struktur) [CHECK]
- Änderungen an zu dokumentierenden Tätigkeiten sind zu erwarten, also möglichst flexible Architektur [CHECK]
 - Warum Plug-In? Vorteile? [CHECK]
- **Middleware**
 - Schnittstellenfunktion nochmal erwähnen [CHECK]
 - Technologie (PHP) [CHECK]
 - Wo wird Middleware positioniert? Welcher Server? (konzeptionell) [NEIN]

Literaturverzeichnis

- [Ado13] (2013, Januar) Phonegap homepage. Adobe Systems Inc. [Online]. Available: <http://www.phonegap.com/>
- [fhFiDHle08] I. für hausärztliche Fortbildung im Deutschen Hausärzterverband (IhF) e.V., “Fortbildungscurriculum ”Versorgungsassistentin in der Hausarztpraxis-VERAH”,” Januar 2008.
- [fhFiDHleV08] I. für hausärztliche Fortbildung im Deutschen Hausärzterverband (IhF) e.V., “Versorgungsassistentin in der Hausarztpraxis - VERAH® [Informationen für Praxisinhaber],” 2008.
- [GMR12] R. Görlitz, L. Müller, and A. Rashid, “An individual patient-oriented stroke manager architecture,” in *Proceedings of the eHealth2012*, Schreier, Hayn, Hörbst, and Ammenwerth, Eds., Wien, Österreich, Mai 2012.
- [Goo13] (2013, Januar) Verwaltung von String Ressourcen bei der nativen Android-Anwendungsentwicklung. Google Inc. [Online]. Available: <http://developer.android.com/guide/topics/resources/string-resource.html>
- [Har10] C. Hardt, “Entwicklung eines prototypischen Softwaresystems zur Unterstützung der Kommunikation und Dokumentation in der stationären Pflege,” Diplomarbeit, Hochschule Karlsruhe Technik und Wirtschaft; Universität Karlsruhe (TH), April 2010.
- [jQu13] (2013) jquery mobile homepage. The jQuery Foundation. [Online]. Available: <http://jquerymobile.com/>
- [Kli10] D. Klingbeil, “Mehr Zeit für die Pflege der Kunden,” *Häusliche Pflege*, vol. 11, pp. 32–34, November 2010.
- [Kop10] D. T. Kopetsch, “Dem deutschen Gesundheitswesen gehen die Ärzte aus! Studie zur Altersstruktur-und Arztlzahlentwicklung,” Bundesärztekammer und Kassenärztliche Bundesvereinigung, Studie 5, August 2010.
- [nG13] nubedian GmbH. (2013, Januar). [Online]. Available: <http://www.nubedian.de/>
- [Pre06] S. B. Pressestelle, “Bevölkerung Deutschlands bis 2050 - 11. koordinierte Bevölkerungsvorausberechnung,” November 2006.

- [Res13] (2013, 01) Residenz Bad Friedrichshall. [Online]. Available: <http://www.pflegedienste-hn.drk.de/>
- [ric12] “Richtlinie des Gemeinsamen Bundesausschusses über die Festlegung ärztlicher Tätigkeiten zur Übertragung auf Berufsangehörige der Alten- und Krankenpflege zur selbständigen Ausübung von Heilkunde im Rahmen von Modellvorhaben nach § 63 Abs. 3c SGB V.” p. 1128, März 2012.
- [Sen13] (2013, Januar) Sencha touch homepage. Sencha Inc. [Online]. Available: <http://www.sencha.com/products/touch>
- [The13a] (2013, Januar) Architektur Eclipse. The Eclipse Foundation. [Online]. Available: <http://help.eclipse.org/juno/index.jsp?topic=%2Forg.eclipse.platform.doc.isv%2Fguide%2Farch.htm>
- [The13b] (2013, Januar) PHP Homepage. The PHP Group. [Online]. Available: <http://www.php.net>
- [vdBMH⁺09] N. van den Berg, C. Meinke, R. Heymann, T. Fiß, E. Suckert, C. Pöller, A. Dreier, H. Rogalski, T. Karopka, R. Oppermann, and W. Hoffmann, “AGnES: Hausarztunterstützung durch qualifizierte Praxismitarbeiter; Evaluation der Modellprojekte: Qualität und Akzeptanz,” *Deutsches Ärzteblatt*, vol. Heft 1-2, no. 106, pp. 3–9, Januar 2009.
- [Vit09] F. VitaBIT, “Pflegeservice von morgen,” *Deutsches Ärzteblatt*, vol. 106, no. 50, pp. 3,4, 2009.
- [zMNS05] M. zur Muehlen, J. V. Nickerson, and K. D. Swenson, “Developing web services choreography standards - the case REST vs. SOAP,” *Decision Support Systems*, vol. 40, no. 1, pp. 9 – 29, 2005, <ce:title>Web services and process management</ce:title>. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0167923604000612>