Mark Niklas Held, Bakk.techn.

Anforderungen an eine Web-GIS Lösung zur Unterstützung der technischen Außendienstprozesse für die öffentliche Beleuchtung

Masterarbeit

Technische Universität Graz

Institut für Geoinformation

Betreuer: Univ.-Prof. Dr.phil. Norbert Bartelme

Graz, November 2013

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Graz, _____ _____

Date Signature

Eidesstattliche Erklärung[1]

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am _____ _____

Datum Unterschrift

[1] Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008; Genehmigung des Senates am 1.12.2008

Danksagung

Mein erster Dank gilt Günther Gleixner, Geschäftsführer der GRINTEC GmbH, welcher mir das sehr interessante Thema zur Verfügung gestellt hat und mir ermöglichte diese Masterarbeit im Umfeld seines Unternehmens durchzuführen. An dieser Stelle möchte ich auch jenen danken, welche mich innerhalb der GRINTEC GmbH unterstützt haben. Hierbei gilt mein Dank vor allem Heike Miketta, welche mich umfassend betreut hat und mir immer mit Rat und Tat zur Seite stand, Gerhard Pachler und Klaus Matuschek, welche mich während meiner Entwicklungsarbeit tatkräftig unterstützten und Elmar Kranjec für die zahlreichen Tipps und interessanten Fachgespräche. Des Weiteren möchte ich mich bei den unterstützenden Mitarbeitern der Energie Graz GmbH & Co KG für die Zusammenarbeit bedanken.

Natürlich möchte ich mich auch bei meinem Betreuer Univ.-Prof. Dr.phil. Norbert Bartelme bedanken, welcher stets Zeit und ein offenes Ohr für mich hatte und für die äußerst interessante und prägende Lehre in den letzten Jahren. Eva Bauer danke ich für das aufmerksame Korrekturlesen meiner Arbeit.

Mein größter Dank gilt meinen Eltern, meiner Schwester und meiner Familie, welche mir durch ihre moralische und finanzielle Unterstützung die Durchführung und den erfolgreichen Abschluss meines Studiums ermöglichten. Abschließend möchte ich mich bei all meinen Studienkollegen und Freunden, auch jene welche ich während meiner Auslandssemester kennengelernt habe, bedanken, dass sie die letzten Jahre zu einer so schönen und unvergesslichen Zeit für mich machten.

Abstract

This master thesis contains the requirements on a mobile and map-based application, which should support the field work of the department for public illumination of an electricity distribution company. For the purpose of identification of the public illumination-objects of the city Graz, the luminaires, the masts and the distributors, were tagged with RFID-transponders. The theoretical part of this master thesis is about the actual- and desired-condition-analysis of the field-works of the before mentioned department. These analyses show the requirements on the mobile application. Furthermore the different automatically identification-systems are annotated, the state of the art of development of mobile application is analyzed and the used network information system is described. The practical part is about the execution of the requirements in a mobile application, which can communicate with RFID-transponders. As a last point there is a reference on unsolved problems and a view to future development opportunities.

Kurzfassung

Die vorliegende Masterarbeit beinhaltet die Anforderungen und Möglichkeiten einer mobilen und Karten basierenden Applikation, welche Außendiensttätigkeiten der Abteilung für öffentlich Beleuchtung eines EVU unterstützen und optimieren soll. Für diesen Zweck sollen auf den Beleuchtungsobjekten, Leuchte, Masten und Beleuchtungsschaltstellen, angebrachte RFID-Transponder zur Unterstützung der Identifikation genutzt werden. Der theoretische Teil der Arbeit beschäftigt sich mit der Ist- und Soll-Zustandsanalyse von Außendiensttätigkeiten der oben genannten Abteilung. Diese Zustandsanalysen zeigen die Anforderungen seitens der EVU an die Applikation. Des Weiteren werden die verschiedenen automatischen Identifikationssysteme erläutert, auf den derzeitigen Stand der Entwicklung von mobilen Applikationen eingegangen und das zum Einsatz kommende Netzinformationssystem beschrieben. Der praktische Teil beschäftigt sich mit der Umsetzung der zuvor analysierten Prozessen und Integration der Anforderungen in eine mobile Applikation, welche mit RFID-Transpondern kommunizieren kann. Abschließend werden ungelöste Problemstellungen erläutert und es wird ein Ausblick auf zukünftige Weiterentwicklungsmöglichkeiten gegeben.

Inhaltsverzeichnis

Danksagung iii

Abstract/Zusammenfassung iv

1 Aufgabenstellung 1
- 1.1 Einleitung . . . 1
- 1.2 Zielsetzung . . . 1

2 Theoretische Grundlagen 3
- 2.1 Ist-Zustandsanalyse . . . 3
  - 2.1.1 Prozessanalyse . . . 3
  - 2.1.2 GIS-Analyse . . . 8
- 2.2 Soll-Zustandsanalyse . . . 10
  - 2.2.1 Hardware . . . 10
  - 2.2.2 Systeminfrastruktur . . . 10
  - 2.2.3 Prozessanalyse . . . 13
  - 2.2.4 GIS-Analyse . . . 17
- 2.3 Automatische Identifikationssysteme . . . 18
  - 2.3.1 Allgemein . . . 18
  - 2.3.2 Vorteile und Nachteile von RFID zu anderen Systemen . . . 19
  - 2.3.3 Bestandteile eines RFID-Systems . . . 21
  - 2.3.4 Near Field Communication . . . 22
  - 2.3.5 Anwendungsbeispiele . . . 23
- 2.4 Mobile Applikationen . . . 25
  - 2.4.1 Entwicklungsarten von mobilen Applikationen . . . 25
  - 2.4.2 Designprinzipien von mobilen Anwendungen . . . 28
- 2.5 Geo- und Netzinformationssysteme . . . 29
  - 2.5.1 Geoinformationssysteme . . . 29
  - 2.5.2 Netzinformationssysteme . . . 30

3 Praktische Umsetzung 33
- 3.1 Allgemeine Entwicklungsumgebung . . . 33
  - 3.1.1 SWebApp-Server . . . 33
  - 3.1.2 SWebApp-Client . . . 35
- 3.2 NFC-Integration in hybride Anwendung . . . 36
  - 3.2.1 Konfiguration von Grails . . . 36
  - 3.2.2 Klasse nfcIO und Methoden . . . 36

3.3 Kopplungsprozess . . . 39
3.3.1 Konfiguration von Grails . . . 39
3.3.2 Kopplung Feature Action . . . 40
3.4 Instandhaltungsprozess . . . 41
3.4.1 Konfiguration von Grails . . . 41
3.4.2 Instandhaltungs-Klassen . . . 44
3.5 Visualisierung der Betriebsmittel in der Karte . . . 47

4 Ergebnisse und Problematiken **49**
4.1 Grundfunktionalitäten SWebApp . . . 49
4.2 Kopplungsprozess . . . 51
4.3 Instandhaltungsprozess . . . 53
4.3.1 Grundelemente . . . 53
4.3.2 Wartungstätigkeiten . . . 54
4.3.3 Störung . . . 55
4.4 Visualisierung der Betriebsmittel in der Karte . . . 60

5 Zusammenfassung und Ausblicke **62**
5.1 Rückblick auf Aufgabenstellung . . . 62
5.1.1 Projektplanung . . . 62
5.1.2 Prototyp-Entwicklung . . . 63
5.1.3 Usability-Test und Fehler- beziehungsweise Problembehebung . 63
5.2 Ausblicke über weitere Schritte zur Umsetzung der Ergebnisse . . . 63
5.3 Ausblicke über weiteren Forschungs- bzw. Handlungsbedarf für Probleme 64
5.3.1 Offene Probleme . . . 64
5.3.2 Erweiterungsaussichten . . . 65

Literatur **67**

Abkürzungsverzeichnis **68**

Abbildungsverzeichnis **69**

Tabellenverzeichnis **70**

Listings **71**

Appendix **72**

1 Aufgabenstellung

1.1 Einleitung

In Kooperation mit der Firma GRINTEC GmbH soll für die Abteilung für öffentliche Beleuchtung der Energie Graz GmbH & Co KG eine mobile Version eines Geoinformationssystems für die Verwaltung der Betriebsmittel erstellt werden.

Diese mobile Anwendung soll Außendienstprozesse im Bereich der Instandhaltung unterstützen beziehungsweise dokumentieren. Für diesen Zweck sollen die Betriebsmittel der Abteilung für öffentliche Beleuchtung mit RFID-Transpondern ausgestattet werden. Diese Transponder dienen erstrangig zur Identifikation der jeweiligen Objekte im GIS und zweitrangig zur zusätzlichen Datensicherung. Aufgrund dieser vereinfachten Identifikation der Betriebsmittel und der Konnektivität zum GIS soll die Durchführung der verschiedenen Instandhaltungsarbeiten vereinfacht werden.

Bei der Applikation handelt es sich um einen Prototypen, der größtenteils auf die Anforderungen der Abteilung für öffentliche Beleuchtung der Energie Graz GmbH & Co KG eingeht. Aus diesem Grund beziehen sich die nachfolgenden Zustandsanalysen, siehe Kapitel Ist-Zustandsanalyse und Soll-Zustandsanalyse, auf die Prozesse dieser Abteilung. Prinzipiell soll die Anwendung so entwickelt werden, dass diese frei erweiterbar und auch für Prozesse anderer Unternehmen einsetzbar ist.

Alle Abkürzungen dieser Masterarbeit können den Abkürzungsverzeichnis auf Seite 68 entnommen werden.

1.2 Zielsetzung

Diese Arbeit lässt sich in drei Hauptziele gliedern:

1. Projektplanung
    - Ist-Zustandsanalyse
    - Soll-Zustandsanalyse

2. Prototyp-Entwicklung
    - Erst Kopplung von auf Betriebsmittel befindenden RFID-Transponder mit dazugehörigem GIS-Objekt
    - Abwicklung von Instandhaltungstätigkeit mittels mobilem Gerät
    - Visualisierung der Betriebsmittel abhängig von Instandhaltungsaufgaben
3. Usability-Test und Fehler- beziehungsweise Problembehandlungen

Das erste Hauptziel beschäftigt sich mit der Projektplanung. Hierbei soll eine ausführliche Ist-Soll-Zustandsanalyse durchgeführt werden, um die genauen Anforderungen an die Anwendung zu erhalten. Es ist von großer Bedeutung zu erkennen, wie bestimmte Arbeitsprozesse derzeit durchgeführt werden um anschließend herauszufinden, wie man diese durch den Einsatz neuer Technologien und eines neuen Systemes optimieren kann. Des Weiteren soll die derzeitige Systeminfrastruktur, vor allem im Hinblick auf relevante GIS-Objekte und deren Attribute, analysiert und gegebenenfalls erweitert werden.

Bei dem zweiten Hauptziel handelt es sich um die Entwicklung der Applikation beziehungsweise eines Prototypen einer Applikation, basierend auf vorherigen Analysen. Es sollen die drei Prozesse der obenstehenden Aufzählung in einer Anwendungssoftware umgesetzt werden. Die Abwicklung der ersten zwei Prozesse, "Kopplung von Betriebsmittel mit GIS" und "Abwicklung einer Instandhaltungstätigkeit", sollen mittels eines mobilen Gerätes, Smartphone oder Tablet, durchgeführt werden können. Bei dem dritten Teil der Anwendung handelt es sich um die Visualisierung der Betriebsmittel abhängig von Instandhaltungsaufgaben und der Benutzer soll somit einen schnelleren Überblick über den Status der verschiedenen Tätigkeiten erhalten.

Abschließend beschäftigt sich das dritte Kernziel mit Usability-Tests und mit der Behebung von auftretenden Problemen der Applikation. Offene Schwierigkeiten beziehungsweise ungelöste Probleme sollen aufgezeigt und analysiert werden.

2 Theoretische Grundlagen

2.1 Ist-Zustandsanalyse

Die Analyse des Ist-Zustandes ist ein wesentlicher Bestandteil der Projektplanung und muss vorab durchgeführt werden. Aufgrund der Ist-Zustandsanalyse kann festgestellt werden inwiefern man durch Einsatz von neuen Technologien beziehungsweise neuer Anwendungssoftware das vorhandene System vereinfachen und verbessern kann. Dieses Kapitel gliedert sich in zwei wesentliche Punkte, die Prozessanalyse und GIS-Analyse. Im Kapitel Soll-Zustandsanalyse wird zusätzlich die Hardware und Systeminfrastruktur analysiert. Aufgrund des Fehlens dieser Komponenten im Ist-Zustand werden diese Kapitel folgend nicht aufgeführt beziehungsweise behandelt.

2.1.1 Prozessanalyse

Folgende Betriebsmittel der Abteilung für öffentliche Beleuchtung der Energie Graz GmbH & Co KG sind für die Prozessanalysen relevant.

- Leuchte
- Mast
- Beleuchtungsschaltstelle

Prozess - Kopplung von Beleuchtungsobjekt (RFID-Transponder) mit GIS

Dieser Prozess beschäftigt sich mit der Verbindung zwischen den Beleuchtungsbetriebsmitteln beziehungsweise mit den RFID-Transpondern auf diesen physischen Objekten und den dazugehörigen GIS-Objekten in der GIS-Datenbank. Da diese Betriebsmittel noch nicht mit RFID-Transpondern bestückt sind, handelt es sich um die Erstkopplung und deshalb gibt es diesen Prozess im Ist-Zustand nicht und wird nur der Vollständigkeit halber hier erwähnt.

Prozess - Instandhaltung

Folgende Instandhaltungsarbeiten, den Betriebsmitteln zugeordnet, müssen in verschiedenen Zyklen beziehungsweise nach Bedarf durchgeführt werden. Anschließend werden diese Instandhaltungsarbeiten einzeln erläutert.

- Leuchte
  - Lampentausch
  - Leuchtenreinigung
- Mast
  - Holzmastprüfung
  - Standsicherheitsprüfung
  - Anstrich
- Beleuchtungsschaltstelle
  - Anlagenprüfung
- nicht zuordenbar
  - Baumschnitt

Lampentausch findet Aufgrund der Lebensdauer der Lampen alle 5 Jahre statt. Im Zuge des Lampentausches sollte des Weiteren eine Reinigung der Leuchten stattfinden, siehe Leuchtenreinigung. Außerdem werden Lampen in Unterführungen und auf Brücken alle 2 bis 3 Jahre gereinigt. Folgende Punkte müssen auf Sicht geprüft werden:

- Holzmasten
- Stahlmasten und Ausleger
- Dachständer
- Schaltstellen
- Leuchten und Aufhänger
- Seilanlagen und Seile
- Mauerhacken und Mastabspannungen

Die Planung dieser Wartungstätigkeit findet derzeit in einer Excel-Liste statt. In dieser Liste wird eingetragen in welchem Jahr und in welchen Bezirken die Lampen getauscht werden sollen. Die Außendienstmitarbeiter erhalten dann für den jeweiligen Bezirk eine Karte und markieren auf dieser die Objekte, bei welchen die Arbeit durchgeführt wurde. Durchgeführte Tätigkeiten werden von den Planungs-Zuständigen vermerkt und abhängig davon wird das nächste Tauschdatum geplant.

Leuchtenreinigung kann als eigener Prozess oder zusammen mit dem Lampentausch stattfinden, siehe Lampentausch. Es müssen die selben Sichtkontrollen wie bei dem Lampentausch durchgeführt werden und die Planung erfolgt auch gleich.

Holzmastprüfung sollte nach der Erstaufstellung nach 15 Jahren und anschließend in der Regel alle 5 Jahre durchgeführt werden. Die Mastbeurteilung wird bei der Prüfung mit den Noten 1 bis 4 in ein Holzmastprotokoll eingetragen. Jedes dieser Holzmastprotokolle wird anschließend in eine Excel-Liste übertragen. Alle mit der Note 4 beurteilten Masten, das heißt der Holzmast muss erneuert werden, müssen von einer Fremdfirma auf Standsicherheit geprüft werden, siehe Standsicherheitsprüfung. In Abbildung 2.1 wird der Ablauf einer Holzmastprüfung anhand eines UML-Aktivtätsdiagrammes gezeigt.

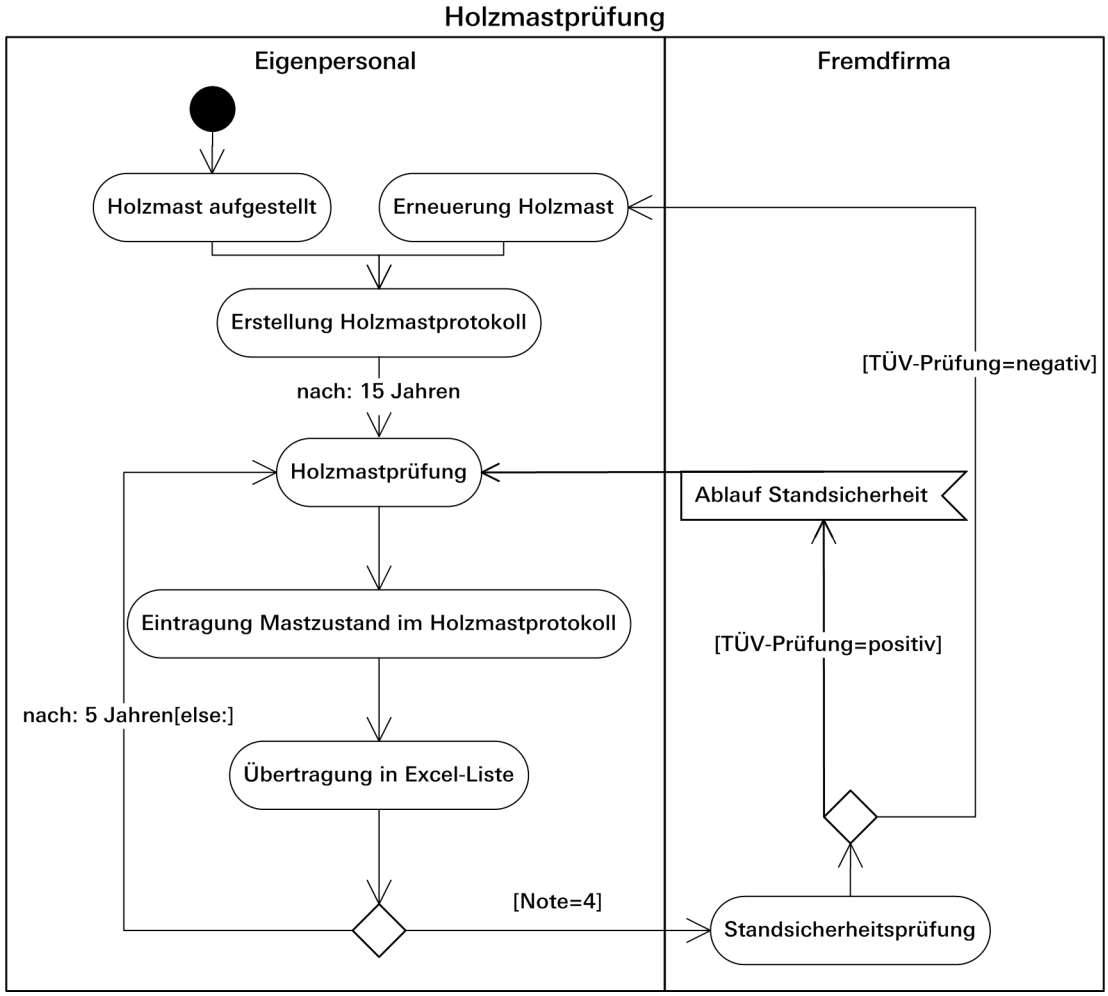


Abbildung 2.1: UML-Aktivitätsdiagram: Holzmastprüfung (Ist-Zustand)

Standsicherheitsprüfung ist eine TÜV-Prüfung für Masten. Stahlmasten über 4 Meter Höhe werden erstmalig nach 20 Jahren und anschließend je nach Bedarf geprüft. Betonmasten werden je nach Bedarf geprüft. Ausgenommen von dieser Prüfung sind Holzmasten, welche bei einer Holzmastprüfung mit dem Prüfergebnis besser als 4

beurteilt wurden. Prinzipiell wird die Standsicherheitsprüfung durch eine Fremdfirma durchgeführt. Abschließend werden Masten-Zertifikate für die Energie Graz GmbH & Co KG erstellt. Eine genauere Erläuterung der Prüfung kann dem UML-Aktivtätsdiagram in Abbildung 2.2 entnommen werden. Stahlmasten unter 4 Meter Höhe, sowie Kunststoffmasten und Poller werden vom Eigenpersonal im Zuge der Anlagenprüfung geprüft.

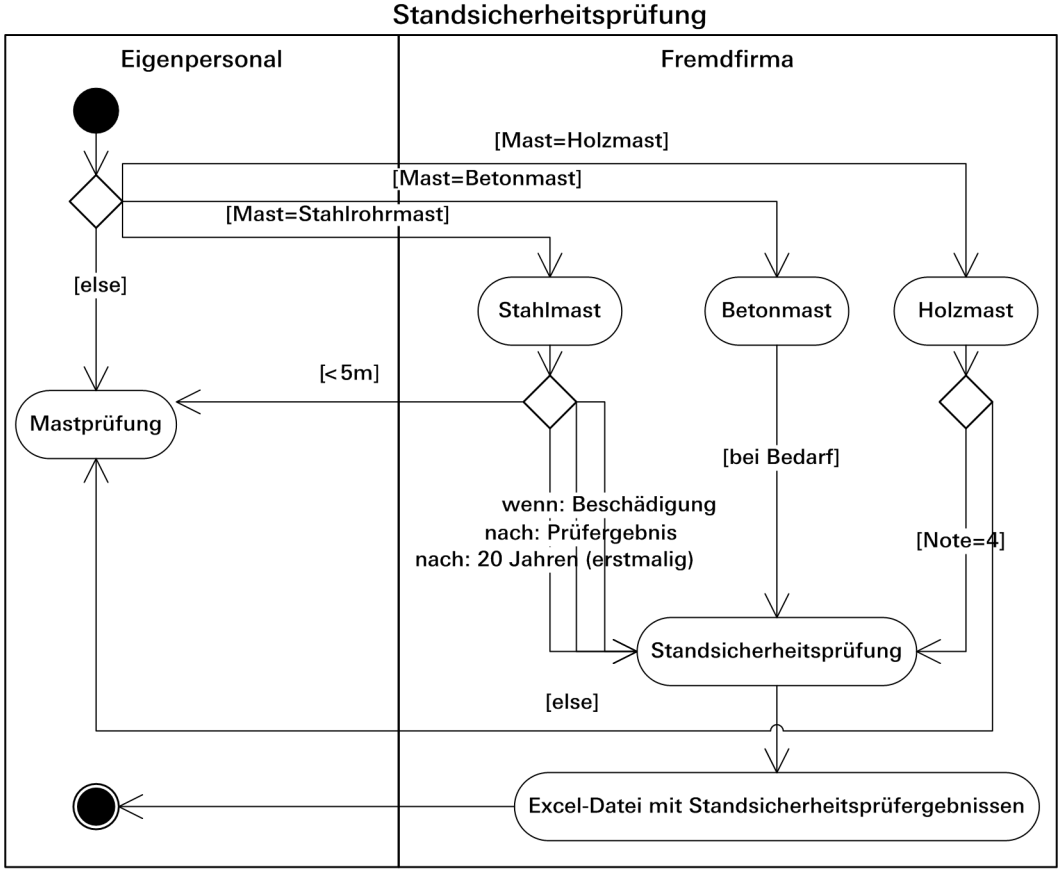


Abbildung 2.2: UML-Aktivitätsdiagram: Standsicherheitsprüfung (Ist-Zustand)

Anstrich soll je nach Bedarf durchgeführt werden. Altmasten, neue Masten, Sondermasten und Masten mit Beschädigung unterliegen dieser Tätigkeit.

Anlagenprüfung soll Bezirksweise im Fünfjahresrhythmus durchgeführt werden. Ziel der Inspektion und Zustandsanalyse ist es, die Sicherheit des Betriebes aller Anlagen zu gewährleisten und den Zustand der Anlagen vertragsmäßig zu dokumentieren. Im UML-Aktivitätsdiagramm in Abbildung 2.3 wird der genaue Ablauf der Anlagenprüfung gezeigt. Diese Prüfung wird einerseits in eine Sichtkontrolle von Verteiler, Mast und Leuchte und andererseits in eine "Elektrotechnische Analgenprüfung" aufgeteilt.

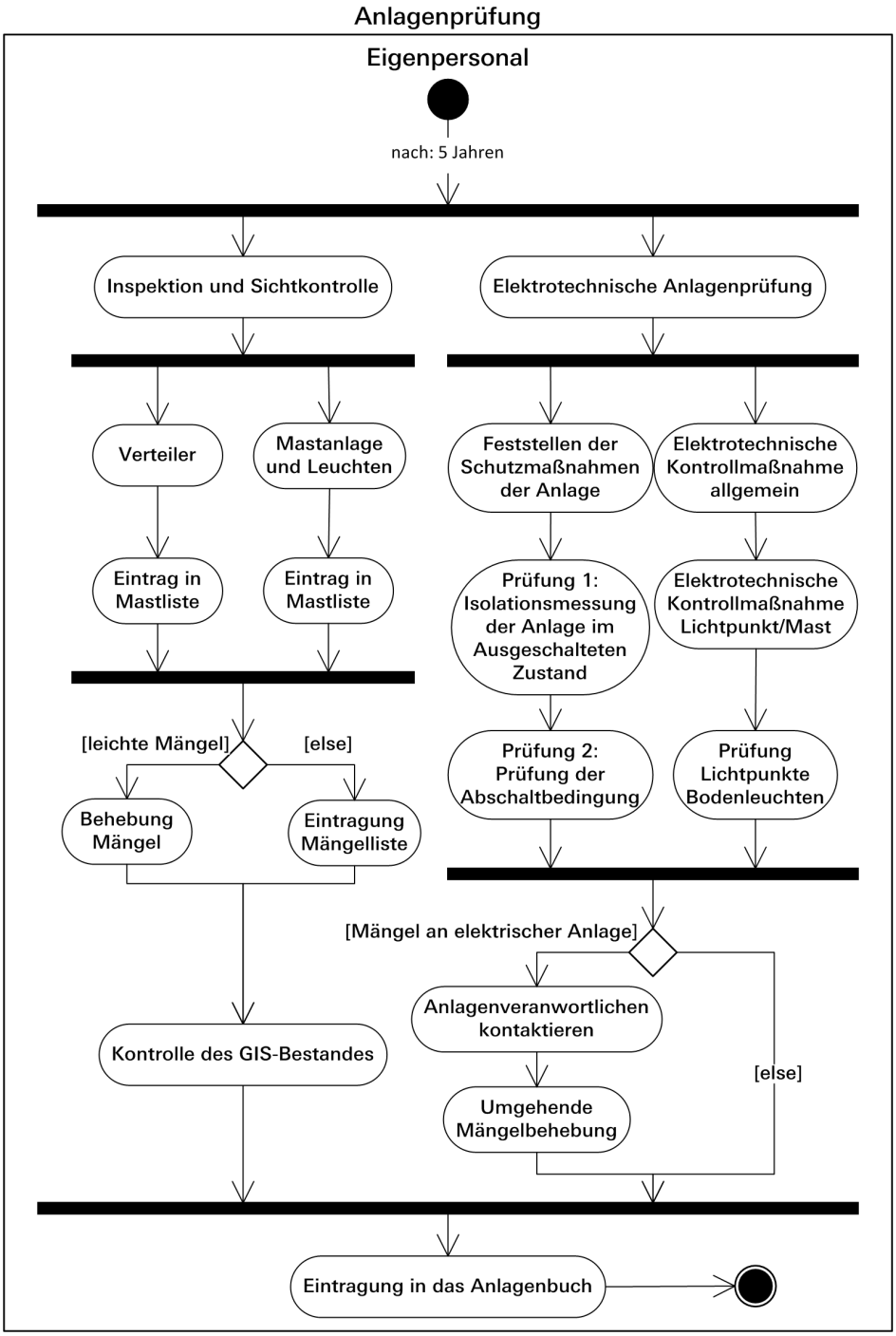


Abbildung 2.3: UML-Aktivitätsdiagram: Anlagenprüfung (Ist-Zustand)

Baumschnitt wird durch eine Fremdfirma durchgeführt. Hierbei werden Leitungen, Leuchten und Seilabspannungen freigeschnitten.

Prozess - Störungsmeldung und -behebung

Neue Störungen werden derzeit zuerst vom Außendienstmitarbeiter vermerkt und dann im Büro in das Omni-Tracker-Ticketing-System der Energie Graz GmbH & Co KG gespeichert. Zur Behebung der Störung kann der Außendienstmitarbeiter eingetragene Störmeldungen auswählen, ansehen, und nach Behebung der Störung aus dem System löschen. Konnte die Störung nicht behoben werden, weil zum Beispiel Reparaturmittel fehlten, kann dies im Omni-Tracker-Ticketing-System eintragen werden.

2.1.2 GIS-Analyse

In diesem Abschnitt werden für die Anwendung relevante GIS-Objekte erläutert. Diese müssen jedoch erweitert werden. Die Erweiterungen sind im Abschnitt Soll-Zustandsanalyse zu finden. In Tabelle 2.1 sind die relevanten GIS-Objekte mit jeweiliger Anzahl der Objekte im Datenbestand der Energie Graz GmbH & Co KG.

Tabelle 2.1: Relevante GIS-Objekte und Anzahl

GIS-Objekt	Anzahl
e_leuchte	25526
e_mast	13900
e_beleuchtungsschaltstelle	1330

Die Attribute dieser Objekte sind großteils unterschiedlich. Nur die Attribute für Objekt-Info, welche Informationen enthält wann und von wem das GIS-Objekt erfasst beziehungsweise geändert wurde, und für Lokation, welche die genauere Adresse des Beleuchtungsobjektes enthält, scheinen in allen drei GIS-Objekten auf. In den folgenden Tabellen sind Attribute von Objekt-Info und Lokation, sowie die Attribute der einzelnen Beleuchtungs-GIS-Objekte zu sehen.

Tabelle 2.2: Atrribute von Objekt-Info (Ist-Zustand)

Objekt-Info
Entstehungsform
erfaßt von
erfaßt am
geändert von
geändert am

Tabelle 2.3: Attribute von Lokation (Ist-Zustand)

Lokation
Gemeindeschlüssel
Gemeinde
Ortschlüssel
Ortsteil
Straßenschlüssel
Straßenname
Hausnummer

Tabelle 2.4: Attribute von E Mast (Ist-Zustand)

E Mast
-System-Id
-Barcode
-Mastnummer Id
-Mastnummer
-Mastzusatz
-Mastart
-Masttyp
-Isolator
-Montagedatum
-Ausbaudatum
-Stilllegungsdatum
-Eigentümer
-Hersteller
-aktueller Status
-zukünftiger Status
-Bemerkungen
-Anzahl der Hausleitungen
-Anzahl der Hausanschlusstlg.
-E Leuchte
-E Tragewerk
-Position
-E Beschriftung
-E Zusatz Text
Objekt-Info[a]
Lokation[b]
Zusatzinfo
-Aufgenommen am
-Aufgenommen von
-Baujahr
-Nennhöhe
-Fundament
-Mastlänge [m]
-Aufstellort
-Verankerung Material
-Verankerung Ziel
-bewertet am
-bewertet von
-Bewertungsart
-Klassifikation Fremd [Monate]
-Klassifikation EGG
-nächste Bewertung am
-Lebensdauer [Monate]
-Lebensdauer ende
-Bewertungsbemerkung
-Blitzschutz
-Penleiterverbindung
-gemessen am
-gemessen von
-Messung offen[Ohm]
-Messung geschlossen [Ohm]
-IK Kurzschlusstram [kA]
-Beleuchtung
-sonstige Leitungen
-sonstige Aufbauten
-fertig
-Anstrich
-Anstrich von
-RAL Farbe
Komponenten
-E NSP-Kabelverteilerschrank
-E Sammelschiene
-E Schaltfeld
-E Leitung allgemein

Tabelle 2.5: Attribute von E Leuchte (Ist-Zustand)

E Leuchte
-System-Id
-Bezeichnung
-Leuchtennummer
-Netzart
-Bauart
-Leuchtentyp
-Vorschaltgeräte
-Schutzklassen
-Lichtpunkthöhe
-Anlagennummer
-aktueller Status
-Bemerkung
-Betriebsart gesamt
-Gezählt?
-Verrechenbar?
-E Mast
-E Beleuchtungsmast
-E Tragewerk
-E Lampe
-Position
-Beschriftung
Objekt-Info[a]
Lokation[b]
Zusatz-Info
-Eigentümer
-Betreiber
-Hersteller
-Montagedatum
-Monteur
-Reinigungsfirma
-Reinigung am
-Lampentausch am
-nächster Lampentausch am
-Stromkosten
-Wartungskosten

Tabelle 2.6: Attribute von E Beleuchtungsschaltstelle (Ist-Zustand)

E Beleuchtungsschaltstelle
-System-Id
-Schaltstellennummer
-Bauart
-Typ
-Fabrikat
-Schrankmaterial
-Sockelmaterial
-Revisionsdatum
-aktueller Status
-Stilllegungsdatum
-Ausbaudatum
-zukünftiger Status
-Hersteller
-Montagedatum
-Monteur
-techn. Überprüfung am
-techn. Überprüfung von
-nächste Überprüfung am
-Bemerkung
-E Schaltfeld
-E Sammelschiene
-E Beleuchtungsmast
-E Schaltbrücke
-E Leitung anlagenintern
-Position
-Beschriftung
Objekt-Info[a]
Lokation[b]

[a] Attributbeschreibung in Tabelle 2.2

[b] Attributbeschreibung in Tabelle 2.3

2.2 Soll-Zustandsanalyse

Nach einer vollständigen Ist-Zustandsanalyse kann der Soll-Zustand, das heißt die möglichen Verbesserungen durch den Einsatz neuer Anwendungen und dadurch entstehenden optimierten Prozesse, analysiert werden. Zusätzlich zu den bereits in Abschnitt 2.1 beschriebenen Punkten "Prozessanalyse" und "GIS-Analyse" werden anschließend die zukünftig verwendete Hardware und Systeminfrastruktur analysiert.

2.2.1 Hardware

Die zur Verfügen stehende Hardware ist ein Android-Smartphone des Fabrikates HTC One SV[1]. Der wesentliche Bestandteil dieses mobilen Gerätes ist der integrierte NFC-Receiver, welcher dazu dient RFID-Transponder zu lesen und zu beschreiben. Genauere Informationen zu dieser Technologie sind in Unterabschnitt 2.3.4 erläutert.

2.2.2 Systeminfrastruktur

In diesem Kapitel wird die im Soll-Zustand verwendete Systeminfrastruktur beziehungsweise -architektur beschrieben. Grundsätzlich ist zu erwähnen, dass die Systeminfrastruktur auf einem Smallworld-GIS von General Electric basiert. Eine Abbildung dieser Struktur unterstützend zur textlichen Beschreibung befindet sich auf Seite 12.

In der Intranet-Zone der Systeminfrastruktur, siehe Abbildung 2.4, also im sogenannten LAN oder auch firmeneigenen Netzwerk befindet sich als grundlegendes Element der GIS-Datenbank-Server, welcher als GIS-Datenspeicher dient. Er kann zum Beispiel keinerlei Analysen durchführen und auch nicht mit dem Client kommunizieren. Davor stehen ein oder mehrere GIS-Applikations-Server, welche Anfragen des GSS-Servers erhalten und sich die dazugehörigen Informationen vom GIS-Datenbank-Server holen beziehungsweise erhalten, verarbeiten und an den GSS-Server zurück geben. Der GSS-Server erhält Serviceanfragen vom Client über ein HTT-Protokoll, verarbeitet diese und leitet diese über TCP/IP an einen der GIS-Applikations-Server weiter. Nach Erhalt von Ergebnissen, übermittelt er diese wieder in eine für den Client verständliche Sprache via HTTP an diesen zurück.

Da man sich mit dem mobilen Client nicht im Firmennetzwerk befindet, werden die Information über das Internet an diesen weitergeleitet. Um eine Datensicherheit zu gewährleisten befindet sich der dafür benötigte Web-Server in der sogenannten Demilitarized Zone, kurz DMZ. Folgend befindet sich die aus de.wikipedia.org (2013a) stammende Definition von DMZ:

[1] genauer Spezifikationen unter http://www.htc.com/at/smartphones/htc-one-sv/

> *Eine Demilitarized Zone (DMZ, auch ent- oder demilitarisierte Zone) bezeichnet ein Computernetz mit sicherheitstechnisch kontrollierten Zugriffsmöglichkeiten auf die daran angeschlossenen Server.*
>
> *Die in der DMZ aufgestellten Systeme werden durch eine oder mehrere Firewalls gegen andere Netze (z. B. Internet, LAN) abgeschirmt. Durch diese Trennung kann der Zugriff auf öffentlich erreichbare Dienste (Bastion Hosts mit z. B. E-Mail, WWW o. ä.) gestattet und gleichzeitig das interne Netz (LAN) vor unberechtigten Zugriffen von außen geschützt werden.*
>
> *Der Sinn besteht darin, auf möglichst sicherer Basis Dienste des Rechnerverbundes sowohl dem WAN (Internet) als auch dem LAN (Intranet) zur Verfügung zu stellen.*
>
> *Ihre Schutzwirkung entfaltet eine DMZ durch die Isolation eines Systems gegenüber zwei oder mehr Netzen.*

In der Internet-Zone kann sich genauso ein normaler GIS Web-Client, hier genannt SWeb-Client, wie auch ein mobiler GIS Web-Client, hier genannt SWebApp-Client, befinden. Der SWebApp-Client beziehungsweise das Smartphone auf dem dieser Client installiert ist, hat Zugang zu den RFID-Transponder am Beleuchtungsobjekt und erhält dadurch die Identifikationsnummer des RFID-Transponders. Diese wird über den Web-Server an den GSS-Server weitergeleitet, welche die Anfrage, wie bereits oben beschrieben, verarbeitet und gewünschte Informationen über den gleichen Weg zurück sendet.

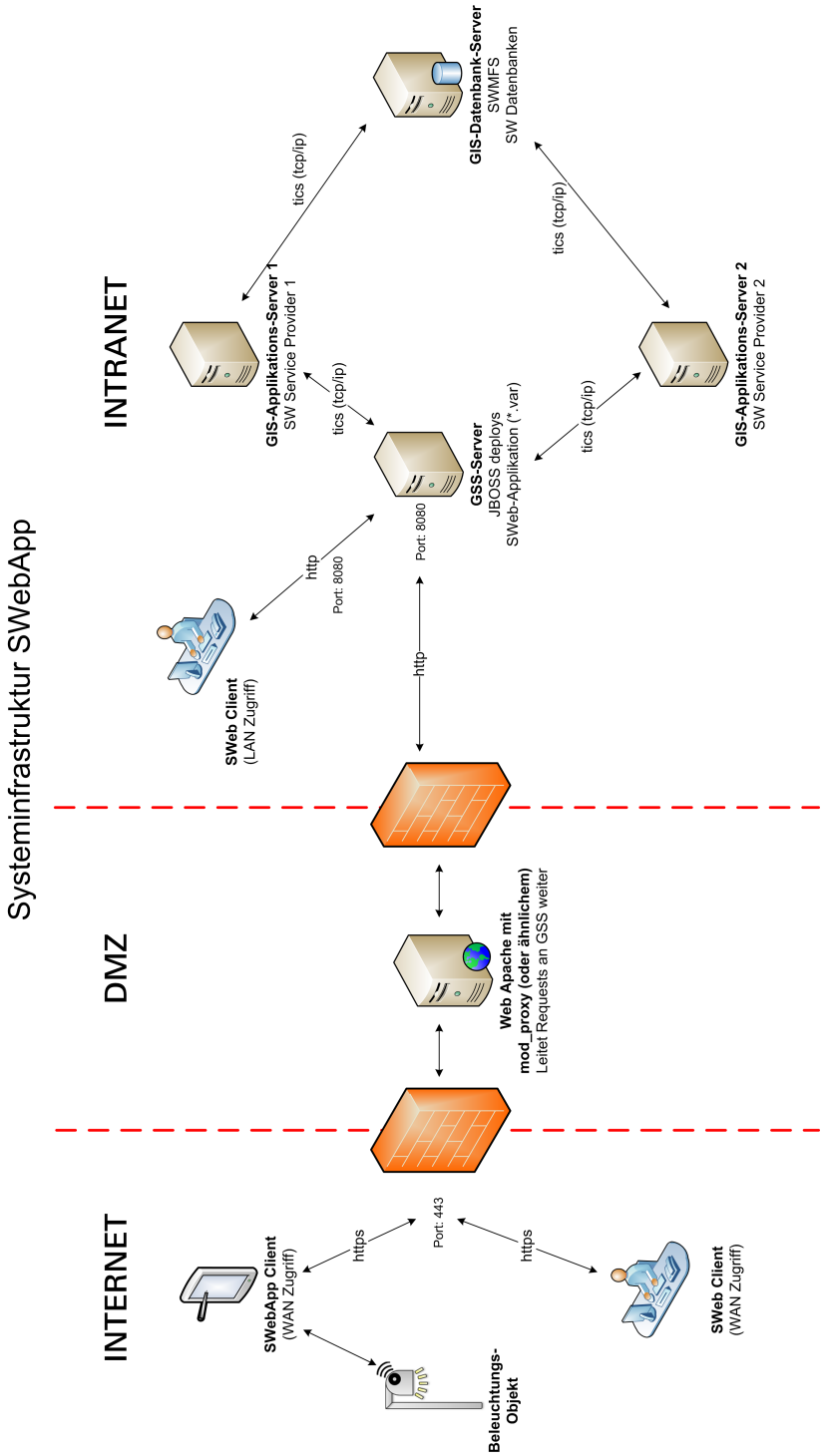


Abbildung 2.4: Systeminfrastruktur (Soll-Zustand)

2.2.3 Prozessanalyse

Wie bereits in Unterabschnitt 2.1.1 erwähnt, sind die Betriebsmittel Leuchte, Mast und Beleuchtungsschaltstelle für die Prozessanalyse relevant.

Prozess - Kopplung von Beleuchtungsobjekt (RFID-Transponder) mit GIS

Es handelt sich um einen komplett neuen Arbeitsschritt, da dieser Prozess erst durch die Einführung von Smartphone und RFID-Transponder für die Instandhaltungstätigkeiten entstanden ist. Wesentlicher Bestandteil dieses Prozesses ist es eine dauerhafte Verbindung oder auch Kopplung zwischen dem GIS-Objekt und den RFID-Transponder am reelen Beleuchtungsobjekt herzustellen. Als Verbindungsmedium wird ein Smartphone mit integriertem NFC-Receiver, siehe Unterabschnitt 2.2.1, verwendet.

Prinzipiell geht es darum, die gescannte RFID-Transponder-Identifikationsnummer am reellen Beleuchtungsobjekt in das Attribut-Feld „RFID“ des zugehörigen und zuvor ausgewählten GIS-Objektes zu schreiben. Der Prozessablauf ist in Abbildung 2.5 dargestellt.

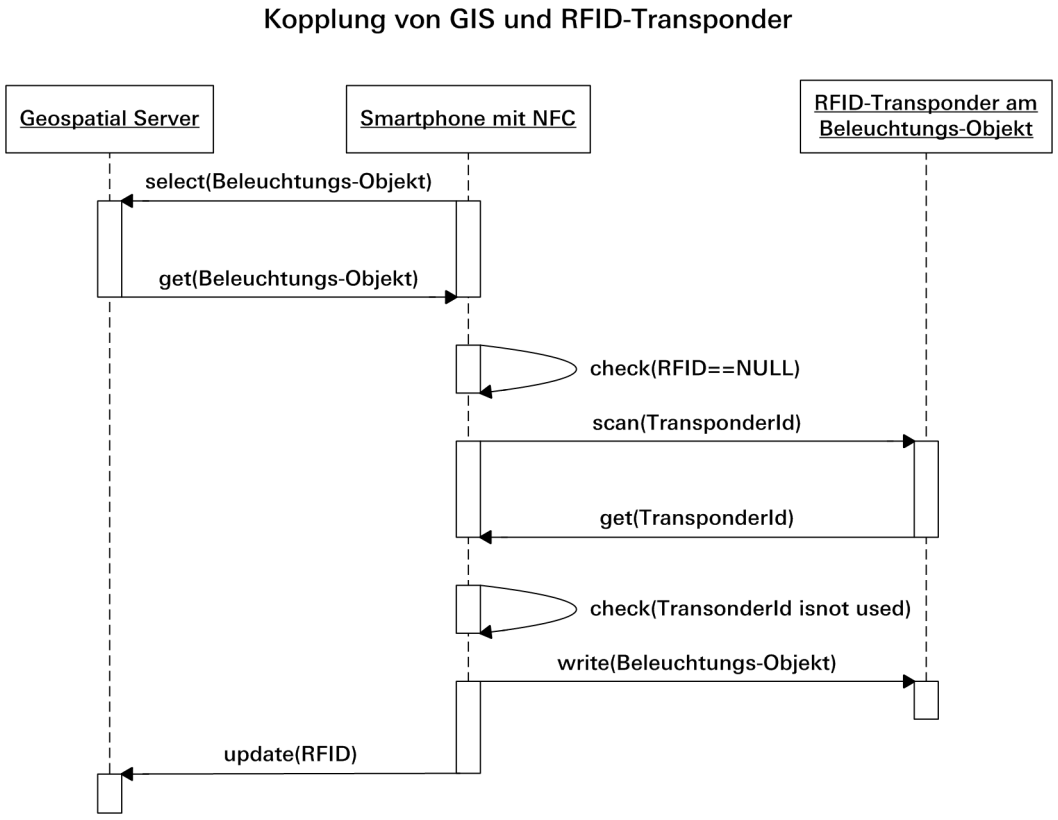


Abbildung 2.5: UML-Sequenzdiagram: Kopplung (Soll-Zustand)

Als erstes muss der Nutzer das zu koppelnde Beleuchtungsobjekt in der Karte auf seinem Smartphone auswählen. Zur groben Vorauswahl beziehungsweise zur Verortung des Smartphones kann die GPS-Position des Smartphones genutzt werden. Nach der Auswahl werden alle benötigten Informationen des selektierten GIS-Objektes vom Geospatial Server auf dem Display des mobilen Gerätes angezeigt. Falls das Attribut-Feld "RFID" bereits einen Wert beinhaltet, wird der Nutzer gewarnt und kann den Kopplungsprozess abbrechen oder diesen Wert überschreiben. Anschließend soll der RFID-Transponder mittels des NFC-Receivers des Smartphones gescannt werden. Das Ergebnis dieses Vorganges ist die einmalige RFID-Transponder-Identifikationsnummer. Nun wird überprüft, ob diese Nummer bereits auf ein Beleuchtungs-Objekt am GSS gespeichert ist. Falls die ID bereits gespeichert ist, wird der Kopplungsvorgang abgebrochen. Eine doppelt vorkommende Transponder-ID würde zu einer Inkonsistenz der Datenbank führen und sollte verhindert werden. Falls die ID noch nicht vergeben wurde, wird diese auf das ausgewählte GIS-Objekt in das Attribut-Feld "RFID" geschrieben. Abschließend werden Datum und Uhrzeit, der eingeloggte Benutzername und bestimmte Kerninformationen des Beleuchtungsobjektes, siehe nachfolgende Aufzählung, auf den RFID-Transponder geschrieben.

- E Leuchte
  - Bezeichnung
  - Bauart
  - Leuchtentyp
- E Mast
  - Mastart
  - Masttyp
  - Material
- E Beleuchtungsschaltstelle
  - Bauart
  - Hersteller
  - Schrankmaterial

Diese Kerninformationen sollen eine Hilfestellung für den Instandhaltungs-Prozess bieten, falls keine Verbindung zum GSS hergestellt werden kann, eventuell aufgrund fehlender Internetverbindung. Nach erfolgreichem Schreiben ist der Kopplungsprozess abgeschlossen.

Prozess - Instandhaltung

Durch die zuvor erläuterte Kopplung wird beim nächsten Scannen des RFID-Transponders am realen Beleuchtungsobjekt im Zuge einer Instandhaltungstätigkeit das richtige GIS-Objekt gefunden und angezeigt. Diese Informationen stammen direkt aus dem GIS

und sollen die Durchführung und die Dokumentation der Außendiensttätigkeit erleichtern. Eine grafische Darstellung dieses Ablaufes kann der Abbildung 2.6 entnommen werden.

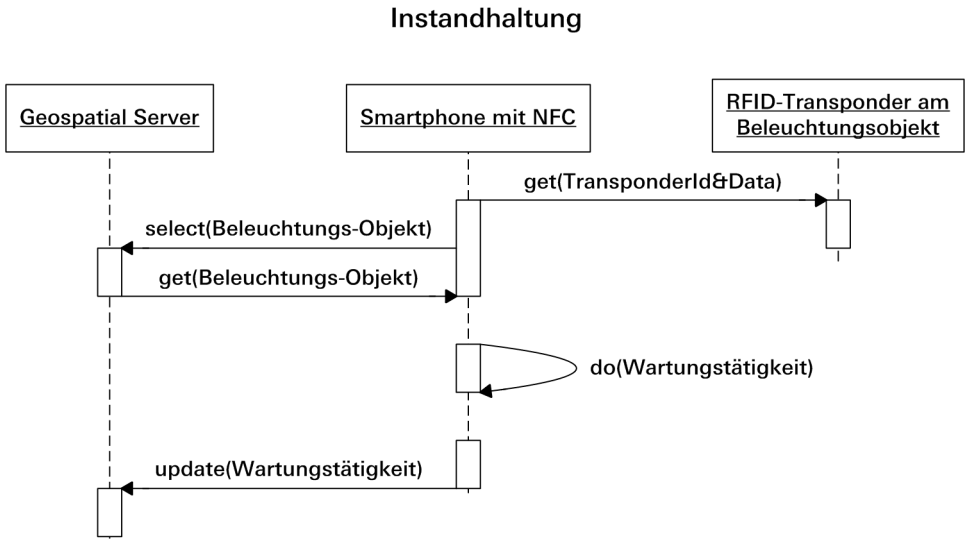


Abbildung 2.6: UML-Sequenzdiagram: Instandhaltung (Soll-Zustand)

Der erste Schritt ist es, mittels Scannen des am Beleuchtungsobjekt befindlichen RFID-Transponders, die eindeutige Transponder-ID zu ermitteln. Anschließend werden vom GSS die wichtigsten Informationen des GIS-Objektes, welches mit der zuvor gescannten Transponder-ID gekoppelt ist, angezeigt. Je nach Objekttyp werden den Mitarbeitern Wartungstätigkeiten angezeigt. Diese Tätigkeiten sind im nächsten Abschnitt beschrieben. Nach erfolgreicher Durchführung der gewählten Tätigkeit werden diese abschließend im GIS gespeichert. Konnte die Wartungstätigkeit nicht erfolgreich Durchgeführt werden, zum Beispiel durch fehlende Reparatur- oder Wartungsmaterialien, wird dies im Feld „Bemerkungen“gespeichert. Dieser Speicherort wird derzeit als Übergangslösung verwendet, da es aufgrund von einer fehlenden Schnittstelle keinen Zugang zum Omni-Tracker-Ticketing-System gibt.

Leuchtenreinigung hat auch nach der Einführung der neuen Applikation den gleichen Zyklus wie im Ist-Zustand beschrieben, siehe Abschnitt 2.1.1. Auch bei der Planung dieser Tätigkeit wird sich nichts ändern, jedoch kann diese durch die Speicherung des Datums der letzten Durchführung dieser Tätigkeit am GIS-Objekt, vereinfacht werden. Der Mitarbeiter kann sich die zu reinigenden Lampen auf der Karte in seiner Smartphone-Applikation anzeigen lassen. Hierdurch soll eine schnellere und übersichtlichere Durchführung dieser Tätigkeit gewährleistet werden. Nach erfolgreichen Leuchtenreinigung und diverser Sichtkontrollen, diese wurden bereits in Abschnitt 2.1.1 beschrieben, wird in das Attribut „Gereinigt am“des zuvor identifizierten Beleuchtungsobjekt das Datum dieser Tätigkeit geschrieben. Außerdem wird

automatisiert in das Feld „Nächste Reinigung am“das aktuelle Datum plus 5 Jahre eingetragen. Konnte die Tätigkeit nicht vollständig durchgeführt werden, besteht die Möglichkeit eine automatisiert erstellte Störmeldung in das Feld „Bemerkungen“und am RFID-Transponder zu speichern.

Leuchtentausch erfolgt sehr ähnlich wie die zuvor erwähnte Leuchtenreinigung. Meist wird bei einem Leuchtenreinigung auch die Leuchte getauscht, jedoch kann diese Tätigkeit auch eigenständig durchgeführt werden. Genauso wie zuvor beschrieben müssen hier ein paar Sichtkontrollen durchgeführt und bestätigt werden. Nach erfolgreichem Abschluss wird das aktuelle Datum in das Feld „Getauscht am“und ein automatisch berechnetes Datum in das Feld „Nächster Tausch am“des Beleuchtungsobjektes gespeichert. Bei nicht erfolgreich abgeschlossenem Tausch wird diese, wie auch in Leuchtenreinigung erwähnt, als Störung gemeldet.

Holzmastprüfung Wie schon zuvor bei Leuchtentausch und Leuchtenreinigung ändert sich nichts beim Zyklus dieser Tätigkeit durch die Einführung einer neuen Applikation. Der generelle Ablauf kann der Abbildung 2.1 entnommen werden, jedoch wird der Mastzustand nicht mehr in eine Excel-Liste sondern direkt auf das GIS-Objekt „E Mast“gespeichert. Somit können zum Beispiel alle Masten mit dem Zustand 4 abgefragt werden. Konnte eine Mastprüfung nicht vollständig durchgeführt werden, wird eine Störmeldung erstellt.

Standsicherheitsprüfung wird von einer externen Firma durchgeführt und da diese keinen Zugang zu der Applikation hat, wird diese Tätigkeit in der Applikation nicht behandelt. Um diese Tätigkeit in die Anwendung zu integrieren, müsste man aus Datenschutzgründen einen eigenen beschränkten Zugang für externe Bearbeiter einrichten.

Anstrich Da es keinen genaue Planung dieser Tätigkeit gibt, wird nur die Durchführung am GIS-Objekt „E Mast“dokumentiert. Hierbei wird in das Attributfeld „Anstrich am“das Datum der Durchführung gespeichert. Konnte diese Tätigkeit nicht vollständig durchgeführt werden, kann eine Störmeldung erstellt werden.

Anlagenprüfung wird auch weiterhin im Fünfjahresrythmus durchgeführt. Die in Abbildung 2.3 gezeigten Punkte müssen hierbei bestätigt werden. Bei erfolgreicher Prüfung, das heißt alle Punkte konnten abgehackt werden, wird das aktuelle Datum in das Attribut-Feld „EGG Technische Überprüfung am“und ein automatisiert berechnetes Datum in das Feld „Nächste Überprüfung am“des GIS-Objektes eingetragen. Bei nicht erfolgreicher Durchführung kann eine Störmeldung erstellt werden.

Baumschnitt ist keinem eindeutigen Beleuchtungsobjekt zuordenbar und wird zusätzlich von einer Fremdfirma durchgeführt und wird deshalb nicht in der Applikation behandelt.

Prozess - Störungsmeldung und -behebung

Der Benutzer hat die Möglichkeit eine Störmeldung zu tätigen. Diese kann aufgrund einer nicht durchführbaren Wartungstätigkeit oder aufgrund einer Auffälligkeit, zum Beispiel Leuchte funktioniert nicht mehr, entstehen. Wie schon zuvor erwähnt, gibt es derzeit keine Verbindung zum Omni-Tracker-Ticketing-System der Energie Graz GmbH & Co KG und deshalb wird eine Störmeldung in das Feld „Bemerkungen“ und auf dem RFID-Transponder gespeichert. Diese Störmeldung wird abhängig vom Grund der Störung automatisiert erstellt und es kann ein individueller Störtext hinzugefügt werden.

2.2.4 GIS-Analyse

Es werden die gleichen GIS-Objekte, wie im Unterabschnitt 2.1.2 beschrieben, verwendet. Diese wurden jedoch um das Attribut „RFID“ erweitert. In dieses wird bei der Kopplung, siehe Abschnitt 2.2.3, die eindeutige Transponder-ID gespeichert. Für die Dokumentation der verschiedenen Wartungstätigkeiten im GIS werden folgende vorhandene Attribut-Felder der verschiedenen GIS-Objekte verwendet.

- E Leuchte
  - Lampentausch
    - „Lampentausch am“: Datum des Tausches
    - „nächster Lampentausch am“: automatisch berechnet
    - „Bemerkungen“: Bemerkungen zum Lampentausch
  - Leuchtenreinigung
    - „Reinigung am“: Datum der Reinigung
    - „nächster Reinigung am“: automatisch berechnet
    - „Bemerkungen“: Bemerkungen zur Leuchtenreinigung
- E Mast
  - Holzmastprüfung
    - „bewertet am“: Datum der Bewertung
    - „Klassifikation“: Note der Klassifikation
    - „Bewertungsbemerkung“: Bemerkung zur Bewertung
  - Mastanstrich
    - „Anstrich am“: Datum des Mastanstriches
    - „Bemerkungen“: Bemerkungen zum Mastanstrich

- E Beleuchtungsschaltstelle
  - Anlagenprüfung
    - „techn. Überprüfung am“: Datum der Prüfung
    - „nächste Überprüfung am“: automatisch berechnet
    - „Bemerkungen“: Bemerkung zur Überprüfung

2.3 Automatische Identifikationssysteme

2.3.1 Allgemein

Automatische Identifikationssysteme dienen zur schnellen und teils kontaktlosen Identifikation von Waren, Personen und Tieren. Man kann zwischen folgenden Systemen unterscheiden:

- Barcode
- Quick Response Code
- Optical Character Recognition
- Biometrische Verfahren
  - Fingerabdruckverfahren
  - Sprachidentifikation
- Chipkarten
- Radio Frequency Identification

Anschließend werden diese Identifikationssysteme kurz beschrieben und es wird ausführlich auf die Vorteile von RFID eingegangen.

Barcode ist ein Verfahren, welches sich mit Binärcode aus einem Feld von parallel angeordneten Strichen und Trennstücken befasst. Diese Kombinationen aus Strichen und Trennstücken kann numerisch oder alphanumerisch mittels optischer Laserabtastung interpretiert werden. Der meist verbreitete Barcode ist der EAN-Code (European Article Number), welcher für Lebensmittel entwickelt und genutzt wird.

Quick Response Code ist eine quadratische Matrix aus schwarzen und weißen Punkten, welche mittels eines Scanners, was auch eine Mobiltelefonkamera sein kann, gelesen und dekodiert werden kann. Anwendungsbereiche dieser Technologie, neben der ursprünglichen Bestimmung wie der Produktionslogistik, ist weiteres die Fahrplanauskunft, auf Visitenkarten, in der Werbung und auch in der Indoor-Navigation. Wesentlicher Vorteil gegenüber des

Abbildung 2.7: Bsp. QR-Code

Barcodes ist, dass der QR-Code für die gleiche Information wesentlich weniger Platz benötigt.

Optical Character Recognition sind speziell entwickelte Schrifttypen, welche maschinell wie auch menschlich gelesen werden können. Großer Vorteil ist die hohe Informationsdichte und die zweifache Lesbarkeit, falls das Lesegerät ausfällt oder Informationen von Mensch und Maschine schnell verarbeitet werden müssen. Die Einsatzgebiete und wohl bekanntesten Beispiele sind in Kontoführungsbüchern, Schecks und auf Briefen beziehungsweise Postkarten und Paketen.

Fingerabdruckverfahren, auch Daktyloskopie, befasst sich mit dem Vergleich der Papillaren und Hautleisten der Fingerspitzen beziehungsweise Fingerkuppen. Anwendungsgebiet dieses Verfahren findet man vorwiegend in der Zutrittskontrolle. Hierbei wird die Fingerkuppe auf ein Lesegerät gedrückt und die gescannten Informationen werden mit Referenzmustern in einer Datenbank verglichen.

Sprachidentifikation basiert auf einem Sprachvergleich einer vom Benutzer in ein Mikrofon gesprochene Sprachprobe mit ihrem Referenzmuster. Haupteinsatzgebiet ist die Zugriffskontrolle.

Chipkarte ist ein elektronischer Datenspeicher, welcher in eine Plastikkarte integriert ist. Informationen daraus können mittels Lesegerät, welches nur auf Kontakt funktioniert, ausgelesen und interpretiert werden. Der wesentliche Unterschied zu RFID ist, dass keine kontaktlose Datenübertragung gewährleistet wird. Einsatzgebiete für Chipkarten sind Telefonchipkarten, Sim-Karten von GSM-Handys oder auch Bankomat- und Kreditkarten.

Radio Frequency Identifkation stammt aus der Funk- und Radartechnologie und befasst sich mit der Interpretation von Radiowellen. Daten werden genau wie bei Chipkarten auf einem elektronischem Datenträger gespeichert, jedoch erfolgt die Energieversorgung und der Datenaustausch nicht durch galvanischen Kontakt mit dem Lesegerät sondern Kontaktlos durch Aufbau eines magnetischen oder elektromagnetischen Feldes zwischen Transponder und Sender beziehungsweise Empfänger. Bei Sender und Empfänger handelt es sich um ein Gerät, welches folgend als Receiver bezeichnet wird. Transponder ist der sogenannte Datenträger.

2.3.2 Vorteile und Nachteile von RFID zu anderen Systemen

Die folgende Tabelle stammt aus Finkenzeller (2012) und zeigt die wesentlichen Unterschiede der verschiedenen Identifikationssysteme.

Tabelle 2.7: Vergleich der verschiedenen Identifikationssystemen

Parameter	Barcode/ QR-Code	OCR	Sprach-erkennung	Biometrie	Chipkarte	RFID-Transponder
Typische Datenmenge/ Byte	1 ∼ 100	1 ∼ 100	-	-	16 ∼ 64k	16 ∼ 64k
Datendichte	gering	gering	hoch	hoch	sehr hoch	sehr hoch
Maschinenlesbarkeit	gut	gut	aufwändig	aufwändig	gut	gut
Lesbarkeit durch Personen	bedingt	einfach	einfach	schwer	unmöglich	unmöglich
Einfluss von Schmutz/Nässe	sehr stark	sehr stark	-	-	möglich (Kontakte)	kein Einfluss
Einfluss von Abdeckung	totaler Ausfall	totaler Ausfall	-	möglich	-	kein Einfluss
Einfluss von Richtung und Lage	gering	gering	-	-	eine Steckrichtung	kein Einfluss
Abnutzung, Verschleiß	bedingt	bedingt	-	-	Kontakt	kein Einfluss
Anschaffungskosten Elektronik	sehr gering	mittel	sehr hoch	sehr hoch	gering	mittel
Betriebskosten (z.B. Drucker)	gering	gering	keine	keine	mittel	keine
unbefugtes Kopieren/ Ändern	leicht	leicht	möglich (Tonband)	unmöglich	unmöglich	unmöglich
Lesegeschwindigkeit	gering ∼ 4 s	gering ∼ 3 s	sehr gering >5 s	sehr gering >5 ... 10 s	gering ∼ 4 s	sehr schnell ∼ 0,5 s
Maximale Entfernung	0 ... 50 cm	<1 cm (Scanner)	0 ... 50 cm	direkter Kontakt	direkter Kontakt	0 ... 5 m

Zusammengefasst hat RFID folgende Vorteile gegenüber den anderen Systemen:

- Speicherung hoher Datenmenge und -dichte
- Ausschließliche Maschinenlesbarkeit, kann als Vorteil und Nachteil betrachtet werden
- Kein Einfluss der Nutzbarkeit durch Schmutz, Nässe, optische Abdeckung
- Nutzbar in jeder Lage und Richtung
- Keine Abnutzung beziehungsweise Verschleiß; die Lebensdauer liegt bei über 20 Jahren
- Hohe Datensicherheit, da Kopieren und Ändern durch Verschlüsslung verhindert werden kann
- Sehr hohe Lesegeschwindigkeit (∼ 0,5 s) bei Leseentfernung von bis zu 5 m, jedoch bei der hier verwendeten NFC-Technologie, siehe Unterabschnitt 2.3.4, maximale Entfernung von 10 cm

Es muss auch auf folgende Nachteile hingewiesen werden:

- Ausschließliche Maschinenlesbarkeit, kann als Vorteil und Nachteil betrachtet werden

- Mittlere Anschaffungskosten für NFC-Transponder zwischen 1€und 3€, je nach Speicherkapazität und Technologie
- Mittlere Anschaffungskosten für Receiver-Hardware, für NFC wird zum Beispiel ein NFC-fähiges Smartphone benötigt

Abschließend ist zu erwähnen, dass für die zu entwickelnde Anwendung Barcode- und QR-Code-Systeme ebenfalls verwendbar und vergleichbar wären. Jedoch aufgrund von zu geringer Datenmenge und -dichte nicht praktikabel sind. Außerdem kann hierbei nur eine einmalige Speicherung durchgeführt werden und keine neuen Daten hinzugefügt oder Datenänderungen durchgeführt werden. Die sehr geringen Anschaffungskosten der erwähnten Systeme, können gegebenenfalls selbst gedruckt werden, wäre ein wesentlicher Vorteil.

2.3.3 Bestandteile eines RFID-Systems

RFID-Systeme unterscheiden sich in den verschiedenen Frequenzbereichen, von 125 kHz bis 5,8 GHz. Ein Grund der verschiedenen Frequenzen ist die Sende- beziehungsweise Empfangsreichweite. Genauso ändern sich je nach Frequenz die Preise für die Receiver, je höher die Frequenz desto teurer die Receiver. Die Sendefrequenzen werden in vier Bereiche unterteilt:

Tabelle 2.8: Verschiedene RFID-Sendefrequenzbereiche

Bezeichnung	Frequenzbereich	Reichweite
low frequency (LF)	30 kHz bis 300 kHz	bis zu 10 cm
high frequency (HF)	3 MHz bis 300 MHz	bis zu 1 m
ultra high frequency (UHF)	300 MHz bis 3 GHz	bis zu 100 m
Mikrowellen	größer 3 GHz	bis zu 200 m

Im wesentlichen bestehen RFID-Systeme immer aus mindestens zwei Komponenten:

- **Transponder** ist an den zu identifizierenden Objekt angebracht und dient als Datenspeicher. Es kann zwischen passivem Transponder, ohne eigener Energie Versorgung, und aktivem Transponder, mit eigener Energieversorgung, zum Beispiel Batterie und dadurch höherer Reichweite, unterschieden werden. Prinzipiell besteht jeder Transponder aus einem Koppelelement (Spule, Antenne) und einem elektronisches Mikrochip (Datenspeicher).
- **Receiver** ist gleichzeitig Sende- und Empfangsgerät und kann Informationen vom Transponder abfragen sowie auf diesen schreiben. Des weiteren dient der Receiver als Energieversorger für passive Transponder. Die Energieversorgung durch den Receiver erfolgt indem ein elektromagnetisches Feld zum Transponder aufgebaut wird.

Abbildung 2.8: Bestandteile eines RFID-Systems

2.3.4 Near Field Communication

Near Field Communication ist ein internationaler Standard zur kontaktlosen Übertragung von Daten über kurze Distanz von maximal 10 cm. Für diese Spezifikation wird der Frequenzbereich 13,56 MHz genutzt und hat eine Datenübertragungsrate von 424 kBit/s. NFC ist durch ISO/IEC 14443A, ISO/IEC 14443B und ISO 18092 genormt, welche vom NFC-Forum entwickelt wurden. 2004 wurde das NFC-Forum von Nokia, Philips und Sony gegründet und hat mittlerweile mehr als 170 Mitglieder. Aufgrund, dass NFC in einigen Smartphones integriert ist, wird diese Technologie immer mehr in mobilen Anwendungen verwendet. Prinzipiell kann man zwischen zwei Betriebsarten, dem active Mode und passive Mode, unterscheiden. Abbildung 2.9 und Abbildung 2.10 stammen aus Finkenzeller (2012) und zeigen die Abläufe von NFC in beiden Betriebsarten.

Active Mode

Im Active Mode gibt es keinen reinen Transponder und keine reinen Receiver. Beide Bestandteile der Systeminfrastruktur sind RFID-Receiver, jedoch haben diese abwechselnd die Aufgabe von Transponder und Receiver. Abwechselnd wird ein magnetische Feld aufgebaut und Daten von einem Gerät zum anderen gesendet beziehungsweise abgefragt.

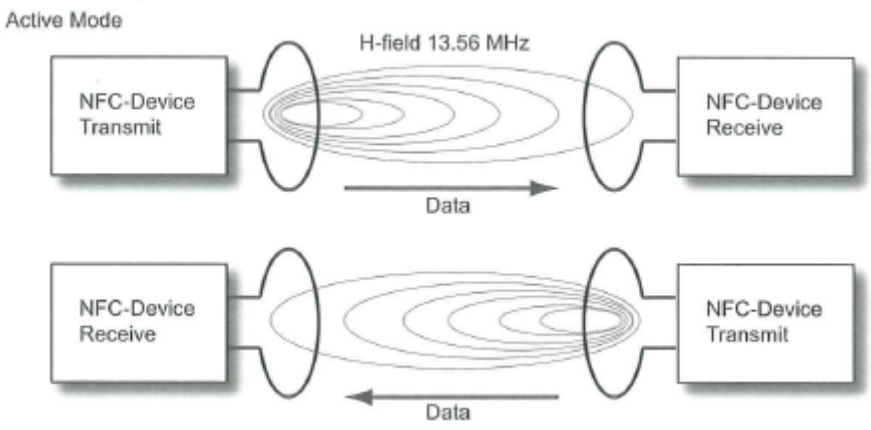


Abbildung 2.9: NFC - Active Mode

Passive Mode

In diesem Mode gibt es einen klaren Receiver und Transponder. Es ähnelt mehr dem üblichen RFID-System. Folgende Definition stammt aus Finkenzeller (2012).

> *Auch beim pasiven Mode erzeugt der NFC-Initiator zur Datenübertragung an das NFC-Target ein magnetisches Wechselfeld, dessen Amplitude im Takt der zu übertragenden Daten moduliert wird (ASK-Modulation). Nach der Übertragung eines Datenblocks wird das Feld jedoch nicht abgeschaltet, sondern unmoduliert weiterhin abgestrahlt. Das NFC-Target kann nun durch das Erzeugen einer Lastmodulation Daten an den NFC-Initiator übertragen. Auch das Verfahren der Lastmodulation kennen wir bereits von RFID-Systemen.*

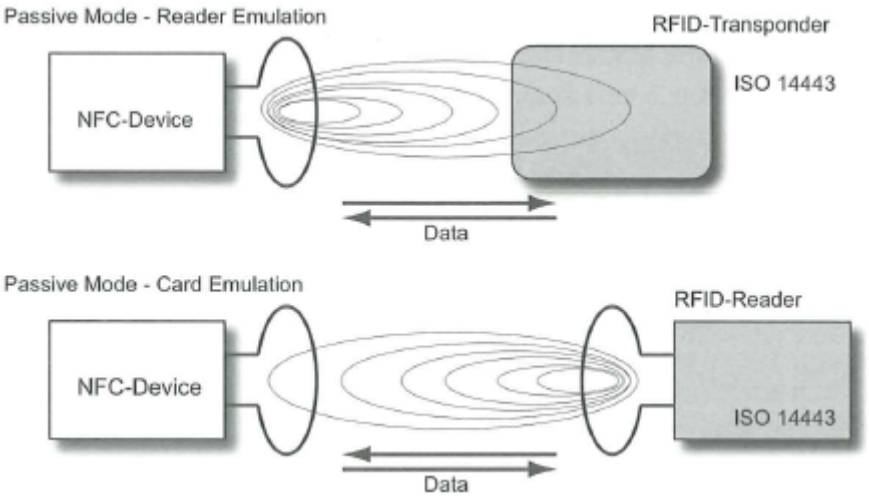


Abbildung 2.10: NFC - Passive Mode

2.3.5 Anwendungsbeispiele

Nachfolgend sind einige Anwendungsbeispiele verschiedener RFID-Systeme, zusammengefasst aus Finkenzeller (2012). Es sind Beispiele von Anwendungen aktiver beziehungsweise passiver Systeme und Systeme mit verschiedenen Reichweiten angeführt.

Öffentlicher Nahverkehr nutzt RFID-Transponder als kontaktlose Fahrscheine. Die wesentlichen Anforderungen an elektronisches Fahrgeld-Management sind Witterungs- und Verschleißempfindlichkeit, hohe Schreib- und Lesegeschwindigkeit und einfache Handhabung. Diese können von dem RFID-System erfüllt werden. Es kommen vor allem kontaktlose Chipkarten aus Plastik wie auch Papier, Armbanduhren und auch Mobiltelefone beziehungsweise Smartphones mit NFC, siehe Unterabschnitt 2.3.4,

zum Einsatz. Die wesentlichen Vorteile von kontaktlosen Chipkarten im öffentlichen Nahverkehr sind:

- Vorteile für den Fahrgast:
  - Bargeldlose Bezahlung, elektronische Fahrscheine können vorab mit einen höheren Geldbetrag aufgeladen werden
  - Monatsfahrkarten können an einem beliebigen Tag im Monat beginnen
  - Vorbezahlte kontaktlose Chipkarten behalten auch bei Umstellung des Tarifes ihre Gültigkeit
- Vorteile für die Fahrer:
  - Wegfall des Fahrkartenverkaufs im Transportmittel
  - kein Bargeld im Fahrzeug vorhanden
  - Wegfall der täglichen Einnahmeabrechnung
- Vorteile für Verkehrsunternehmen und -verbund
  - Kostenreduktion bei Betriebs- und Wartungskosten von Verkaufsautomaten und Fahrscheinentwertern
  - Leistungsorientierte Verbundabrechnung der einzelnen Verbundpartner
  - Gewinnung aussagekräftiger statistischer Daten

Kontakloser Zahlungsverkehr ist ein wesentliches Anwendungsgebiet von RFID-Systemen. Es wird zwischen geschlossenen und offenen Zahlungsverkehrssystemen unterschieden. Geschlossene Systeme sind nur innerhalb des Wirkungsbereiches eines einzelnen Anbieters gültig, wie zum Beispiel die kontaklose Chipkarte der Mensa der Universität. In den meisten Fällen handelt es sich hierbei um ein Prepaid-System. Bei offenen Zahlungsverkehrssystemen dient die kontaklose Chipkarte als Bargeldersatz. Diese basieren meist auf nationalen oder globalen Standards und sind deshalb auch landes- oder sogar weltweit verfügbar. Im wesentlichen funktionieren diese gleich wie eine herkömmliche EC- oder Kreditkarte, nur dass diese auch kontaktlos nutzbar sind. Aufgrund des aufkommen von NFC-Receivern in Mobiltelefonen beziehungsweise Smartphones, siehe Unterabschnitt 2.3.4, werden diese neuerdings vermehrt als Bezahlsysteme verwendet.

Elektronischer Reisepass ist ein mit RFID-Transponder ausgestatteter Reisepass, auch ePass genannt. Dieser wurde bis August 2006 in den damaligen 25 Mitgliedstaaten der EU eingeführt. Auch Japan, Singapur, USA und weitere Länder wollen diesen Pass einführen. Durch den in den Reisepass laminierten Transponder soll die Fälschungssicherheit erhöht werden.

Ski-Ticketing ist ein weiteres Anwendungsgebiet von RFID-Systemen. Hierbei werden Papiertickets durch kontaklose Chipkarten ersetzt. Mit speziellen Lesegeräten vom Liftbetreiber können diese Transponder gelesen werden. Die Lesereichweite wurde

hierbei so ausgelegt, dass das Ticket nicht aus der Jackentasche genommen werden muss.

Zutrittskontrollen befasst sich mit der Zutrittsberechtigung einzelner Personen zu Gebäuden, Geländen oder einzelnen Räumen. Prinzipiell kann man zwischen Online- und Offline-Systemen unterscheiden. Wenn eine große Anzahl von Personen an nur wenigen Eingängen auf eine Zutrittsberechtigung hin überprüft werden muss, werden vorwiegend Online-Systeme verwendet. Alle Zutrittsterminals sind hierbei mit einem zentralen Rechner verbunden, welcher eine Datenbank mit Zutrittsrechten enthält. Somit können möglichst schnell und einfach mehr Zugriffsrechte vergeben werden. Offline-Systeme werden meist eingesetzt bei vielen Räumen in denen wenige Personen Zutritt haben. Zugriffsrechte werden hierbei nur auf den RFID-Receiver beziehungsweise Terminal gespeichert. Dieses System wird großteils in Hotels verwendet.

Weiter Anwendungsbeispiele werden nur aufgezählt jedoch nicht weiter erläutert. Diese können in Finkenzeller (2012) nachgelesen werden.

- Internationaler Containerverkehr
- Tieridentifikation
- Brieftrauben-Preisflug
- Elektronische Wegfahrsperre
- Behälteridentifikation
- Sportliche Veranstaltungen
- Industrieautomation
- Medizinische Anwendungen

2.4 Mobile Applikationen

In diesem Kapitel wird auf die Entwicklungsarten von mobilen Anwendungen eingegangen. Des weiteren werden Designprinzipien von mobilen Anwendungen erläutert.

2.4.1 Entwicklungsarten von mobilen Applikationen

Anwendungen für mobile Geräte, also Smartphones und Tablets, können Nativ, Webbasiert oder Hybrid entwickelt werden. Im folgenden Kapitel werden die drei Entwicklungsarten, deren Unterschiede, sowie Vor- und Nachteile beschrieben. Folgend wird nur mehr die Bezeichnung Smartphone verwendet, hierbei wird jedoch genauso ein Tablet gemeint.

Native Anwendungen

Native Anwendungen sind Anwendungen, welche für ein bestimmtes Betriebssystem, zum Beispiel iOS oder Android, entwickelt wurden und direkt auf das jeweilige mobile Gerät installiert werden. Es handelt sich also um ein eigenständiges Programm auf dem Smartphone. Somit kann bestmöglich die Rechenleistung des Smartphones verwendet werden und es können die Systemfunktionen, wie Kamera, GPS, Bewegungssensoren und so weiter, angesprochen werden. Diese Anwendungen können meist nur über die verschiedenen Verkaufsplattformen der verschiedenen Hersteller von mobilen Betriebssystemen erhalten werden. Jedoch muss eine Applikation bevor sie zum Download angeboten wird, vom Betreiber der Plattform genehmigt werden. Dieser Genehmigungsprozess ist selbstverständlich nicht kostenlos und meist zeitaufwendig. iOS-Applikationen können nur über deren AppStore, auf das iPhone installiert werden. Auch nach Genehmigung von Anwendungen von Apple, müssen Anwendungsupdates neu geprüft werden. Android-Applikationen können genauso direkt auf das Smartphone installiert werden, scheinen jedoch nur nach Genehmigung von Google im PlayStore auf.

Zusammenfassend die Vor- und Nachteile von nativen Anwendungen.

Vorteile:

- Bessere Rechen-Performance
- Zugriff auf Systemfunktionen
- Einfacherer Vertriebsweg über die verschieden Verkaufsplattformen
- Programm befindet sich am Smartphone und somit am Home-Bildschirm

Nachteile:

- Genehmigung durch Betriebssystemhersteller, um in den eigenen Store zu aufgenommen werden
- Kosten, um in den Anwendungsstore aufgenommen zu werden
- Zeitaufwendig durch Prüfung
- Mehr Aufwand, um für jedes Betriebssystem eine Applikation zu entwickeln
- Generell höherer Entwicklungsaufwand als für Webanwendungen

Webanwendungen

Eine Webanwendung kann nur über den Browser des Smartphones aufgerufen werden und besitzt deshalb auch nur den Funktionsumfang des Browsers. Jedoch kann sie mit jedem Browser auf jedem beliebigen mobilen Betriebssystem aufgerufen werden und ist somit Plattform unabhängig. Der Browser des Smartphones dient nur zur Darstellung der webbasierten Anwendung, dieser kann jedoch nicht auf die Systemfunktionen, wie Kamera oder Bewegungssensoren und so weiter, zugreifen. Der Zugriff auf den GPS-Empfänger des Smartphones ist seit HTML5 möglich. Generell ist zu erwähnen,

dass durch Aufkommen von neuen Standards wie, HTML5 und CSS3, die Entwicklung und die Performance von Webanwendungen um einiges verbessert wurde. Eine WebApp kann und muss jedoch nicht in den Vertriebsplattformen der verschiedenen mobilen Betriebssystemhersteller aufscheinen. Dies kann als Vor- und Nachteil erachtet werden. Prinzipiell sind Webanwendungen schneller und kostengünstiger entwickelt als native Anwendungen und laufen ohne große Anpassungen auf jedem mobilen Gerät. Auch jedes Update der App ist wesentlich schneller verfügbar als bei nativen Anwendungen.

Folgend werden die wesentlichen Vor- und Nachteile von Webanwendungen aufgelistet.

Vorteile:

- Plattform unabhängig
- Verkauf ohne eigenen betriebssystemabhängigen Vertriebsstore
  - Keine Extrakosten
  - Schneller Release, da keine Genehmigung nötig
  - Sofortige Update-Möglichkeit
  - Keine Einschränkungen durch Betriebssystemhersteller
- schneller und einfacher zu entwickeln

Nachteile:

- Kein Zugriff auf Systemfunktionen, außer GPS
- Verkauf ohne eigenen betriebssystemabhängigen Vertriebsstore
  - Schneller Vertriebsweg
- Schlechtere Performance als native Anwendungen

Hybride Anwendungen

Eine hybride Applikation kann man als Kombination von Web- und Native-Apps sehen. Prinzipiell wird eine browserbasierte Anwendung in HTML5, CSS3 und JavaScript entwickelt. Diese mobile Website wird in einen nativen Container eingebettet und hat somit den Anschein einer nativen Anwendung. Hierbei kann es sich um einen Container jedes beliebigen mobilen Betriebssystems handeln. Die Integration kann mittels JavaScript Framework wie zum Beispiel Apache Cordova stattfinden. Diese bieten auch die Möglichkeit auf die Hardwarefunktionen des Smartphones zu zugreifen. Somit können die Vorteile von Nativen- und Web-Anwendungen genutzt werden. In einer Studie des Gartner Inc., weltführendes Forschungsinstitut für Informationstechnologien, werden 2016 voraussichtlich mehr als 50% mobiler Anwendungen Hybrid-Apps sein, siehe Gartner (2013).

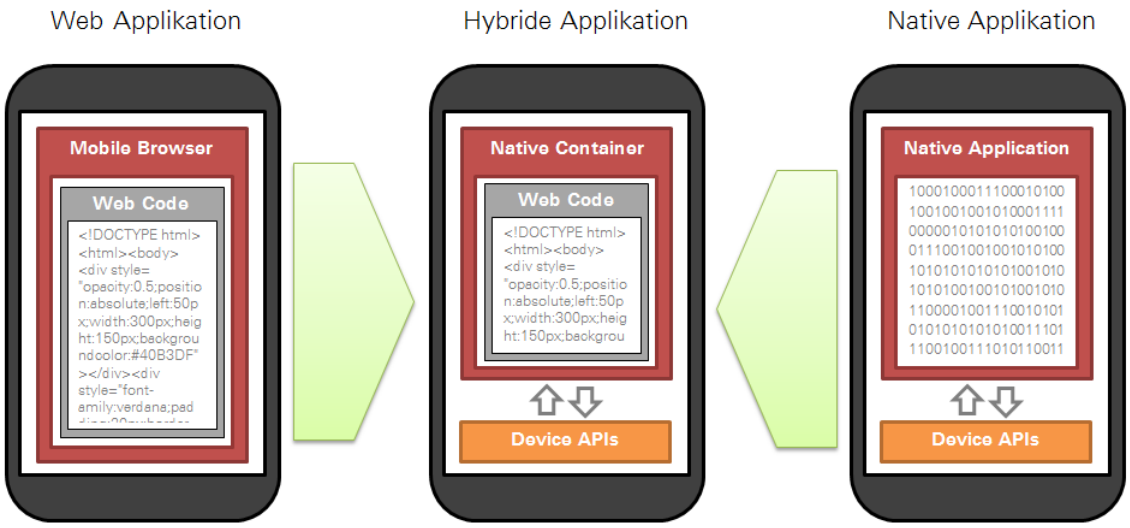


Abbildung 2.11: Web-, Hybrid- und Native-App

Folgend eine Liste der Vor- und Nachteile.

Vorteile:

- Schnellere und einfachere Entwicklung, da nur mit HTML5, CSS3 und JavaScript
- Quasi Plattform unabhängig, nur Integration in den jeweiligen Container der verschiedenen mobilen Betriebssysteme
- Vertrieb via Internet, wenn nur als Web-Applikation angeboten, jedoch dann ohne Hardwarezugriff
- Vertrieb genauso über die Plattformen der verschieden Betriebssysteme nach Genehmigung

Nachteile:

- Schlechtere Performance als native Anwendung

2.4.2 Designprinzipien von mobilen Anwendungen

Bei der Entwicklung von mobilen Anwendungen müssen einige Details beachtet und richtig umgesetzt werden. Folgende Herausforderungen sind zu berücksichtigen, zitiert aus Spiering (2011).

- *Der Bildschirm ist wesentlich kleiner.*
- *Der Benutzer bedient die Anwendungen mit dem Finger, nicht mit der Maus.*
- *Die Texteingabe ist selbst bei High-End-Telefonen immer noch keine Freude.*
- *Der Benutzer hat unterwegs andere Anforderungen an eine Anwendung oder Webseite als zu Hause oder bei der Arbeit am PC.*

- *Trotz UMTS und anderen Technologien ist das mobile Internet nicht beständig schnell, ein Nutzer kann sogar in Gebiete ohne Empfang kommen.*

Die zuvor aufgezählten Herausforderungen können mit folgenden Vorschlägen gelöst werden.

- Einheitliches Konzept und Use-Cases für die Vorteile durch die Nutzung der Anwendung
- Große Grafiken und lange Texte eignen sich nicht für mobile Anwendungen, da diese lange Ladezeiten verursachen und platz mäßig nicht gut darstellbar auf den kleinen Smartphone-Displays sind
- Lange Ladezeiten vermeiden und versuchen zu umgehen
- Möglichst wenige Texteingaben durch den Benutzer
- Komplexe Formulare vermeiden
- Design möglichst einfach und übersichtlich halten

Design von Web- bzw. Hybrid-Apps

Abbildung 2.12: Logo Sencha

Um eine Web- beziehungsweise Hybrid-App ähnlich wie eine native Anwendung aussehen zulassen, können verschiedene CSS- und JavaScript-Frameworks verwendet werden. Durch diese Frameworks wird das Gestalten dieser Anwendungen um einiges vereinfacht. Typische Elemente wie Schieberegler, Buttons, Formulare und so weiter müssen nicht erst selbst gestaltet werden, sondern können direkt eingebunden werden. Das für diese Anwendung verwendete Framework ist Sencha Touch. Weitere bekannte Frameworks sind iWebKit 5 oder jQTouch. All diese Frameworks werden kostenlos zur Verfügung gestellt.

2.5 Geo- und Netzinformationssysteme

In diesem Abschnitt wird der Begriff GIS und NIS erläutert.

2.5.1 Geoinformationssysteme

Folgende Definition stammt aus (Günther, 2009).

> *Ein Geographisches Informationssystem oder Geoinformationssystem (GIS) ist ein Informationssystem, das aus den gleichwertigen Komponenten Hardware, Software, Daten und Anwender bzw. die von ihm erstellte Anwendung (HSDA) besteht. Mit dessen Hilfe können Geodaten digital erfasst, bearbeitet, verwaltet, umfangreich analysiert, modelliert und alphanumerisch und graphisch präsentiert (EVAP)*

> *werden. Durch den Raumbezug der Daten unterscheidet sich ein GIS in seinen Bearbeitungsmethoden wesentlich von anderen Informationssystemen. Durch seine umfangreichen Analysemethoden hebt es sich grundsätzlich von reinen Zeichen- und Präsentationsprogrammen ab.*

Um das weite Anwendungsfeld von GIS weiter zu unterteilen und spezifizieren sind in den letzten Jahren folgende untergeordnete Systeme entstanden.

- Landinformationssystem (LIS)
- Rauminformationssystem (RIS)
- Umweltinformationssystem (UIS)
- Netzinformationssystem (NIS)
- Fachinformationssystem (FIS)

Jedes dieser Systeme hat verschiedene Anforderungen an Datenart, Geometrie, Topologie, Thematik, Maßstab, zeitliche Auflösungen und Genauigkeit der Daten. Im folgenden Abschnitt wird genauer auf das Netzinformationssystem eingegangen. Erläuterungen und Spezifikation der anderen Systeme können in Bill (2010) nachgelesen werden.

2.5.2 Netzinformationssysteme

Anwendungsbereiche

Typische Anwendungen eines NIS sind im Ver- und Entsorgungsbereich. Das System dient hierbei der Verwaltung und Dokumentation der Betriebsmittel von Netzbetreibern. Unter Netzbetreiber versteht man Energieversorgungsunternehmen (EVU), Kommunen, Post, Wasser- und Bodenverbände, Militär und viele andere. Bei den EVUs kann man weiter in die Sparten Strom, Gas, Wasser, Fernwärme und die Bereiche Abwasser, Kommunikation (Telefon, TV, Internet), Stadtbeleuchtung, Roh- und Fertigprodukte wie Öl und Müllentsorgung unterscheiden.

Anforderungen

Folgend ist eine Liste einiger Anforderungen von EVUs an ein NIS. Hierbei muss bedacht werden, dass es sich um mehrere 1000 km von Leitungen beziehungsweise tausende Betriebsmittel handelt, welche erfasst, bearbeitet, verwaltet, analysiert und präsentiert werden müssen und dies meist auf sehr beschränkten Raum. Des weiteren sollte erwähnt werden, dass die Leitungen als Betriebsvermögen angesehen werden können und deshalb die Dokumentation einen hohen Stellenwert hat.

Die wichtigsten Anforderungen an ein NIS:

- Leitungsdokumentation
- Planauskunft für Bauvorhaben

- Verwaltung von Kunden- und Auftragsbestand
- Netzberechnungen beziehungsweise -analyse zum Beispiel für Störungssuche und -behebung
- Wartungsübersicht und -planung
- Verbrauchsabrechnung
- Jahresstatistiken beziehungsweise generell Statistiken

Aufgrund der vielen Anforderungen müssen sowohl interne wie auch externe Schnittstellen zu verschieden Systemen und Datenlieferanten geschaffen werden.

Aus den obigen Anforderungen wird deutlich, dass sowohl die kartographische Lage, aber vor allem die Topologie von Daten von großer Bedeutung sind. Die geometrische Exaktheit ist oft nicht so wichtig, da diese oft nur für Darstellungszwecke verwendet wird.

Beispiel: Smallworld

Anschließend wird das NIS Smallworld von General Electric beschrieben. Dieses NIS basiert auf den 10 Hausforderungen an ein GIS von Richard Newell aus den Paper Newell (1989).

- *The topology capture problem*
- *Large data volumes*
- *Continuous mapping - the seamless database*
- *Accommodating existing database*
- *Version management - the problem of the long transaction*
- *Hybrid vector raster database*
- *Overlay analysis - a problem of robustness*
- *Very large polygons*
- *The front end language - the seamless programming environment*
- *The query language*

Schlussendlich wurde Smallworld GIS aufgrund der obenstehenden Anforderungen mithilfe objektorientierter Methoden und Sprachen entwickelt. Um diese Hausforderungen zu meistern entstand 1990 die objektorientierte Programmiersprache *Magik*.

Smallworld ist modular aufgebaut und kann je nach Anforderung durch Komponenten erweitert werden, wie in Abbildung 2.13 zusehen. Das System besteht aus einer *GE Smallworld Basistechnologie*, auch *Smallworld Core Spatial Technology* genannt, welches für die allgemeine GIS-Funktionalität, wie zum Beispiel allgemein topologische Abfragen und für das eigene topologische Konzept zuständig ist. Die *Fachschalenbasis* bietet die Funktionalitäten und das Datenmodel für die *Fachschalen*, welche der Core nicht zur Verfügung stellt. Diese ist speziell für die *Fachschalen* des deutschsprachigen Raumes zuständig und beinhaltet zum Beispiel speziell benötigte Bemaßungskomponenten. *Fachschalen* sind spartenspezifische Datenmodelle und Funktionalitäten. Beispielsweise

braucht ein Stromnetz andere Objekte als ein Wassernetz und eine Netzwerkanalyse im Stromnetz arbeitet anderes wie die gleiche Analyse im Wassernetz.

Durch die Fachschalenstruktur ist dem System in Funktionalität und Datenmodellstruktur keine Grenzen gesetzt, denn falls bestimmte Anforderungen benötigt werden, können diese in die bestehenden *Fachschalen* integriert werden oder es kann eine neue *Fachschale* erstellt werden.

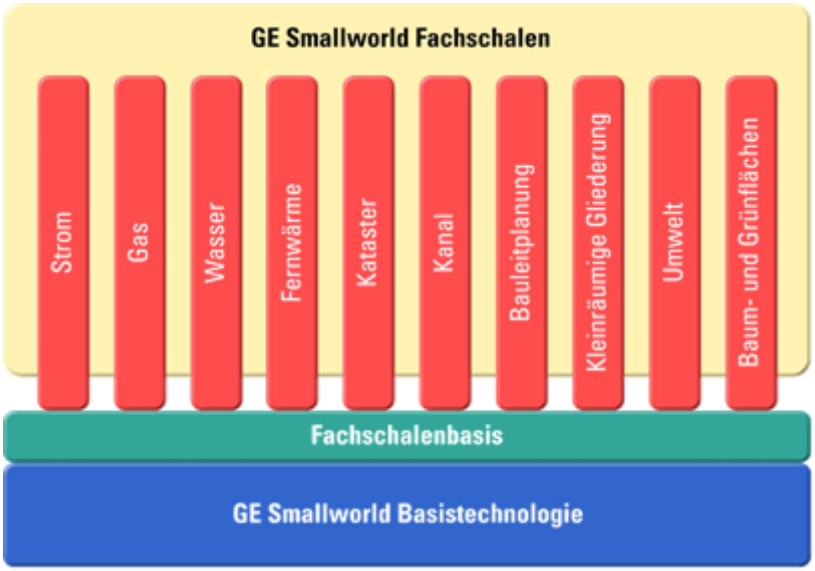


Abbildung 2.13: Smallworld Fachschalen (Abbildung von GE)

Um nicht nur den üblichen Desktop-Zugang zu dem NIS zu haben, bietet der sogenannte *Smallworld GeoSpatial Server* zahlreiche *Business Services* und *Web Services*, welche ermöglicht die NIS-Daten auf eine Webbrowser oder auch einen mobilen Webbrowser darzustellen und zu nutzen. Bei den *Business Services* stehen unter anderem folgende zur Verfügung:

- Karte
- Abfrage
- Update
- Netzwerkanalyse
- Objektinfo

Folgende beispielhafte *Web Services* bietet der *Smallworld GSS Server*:

- Map Service
- Message Webservice
- Reply Webservice

3 Praktische Umsetzung

3.1 Allgemeine Entwicklungsumgebung

In diesem Kapitel wird beschrieben, wie die Entwicklungsumgebung, die Softwarestruktur und die Client-Server-Kommunikation aufgebaut ist. Es wird eine mobile und browserbasierende Version eines Smallworld NIS, welche von der GRINTEC GmbH entwickelt wurde, verwendet. Dieses Produkt nennt sich SWebApp und kann umfangreich erweitert und konfiguriert werden.

Die Systeminfrastruktur von SWebApp besteht aus 5 Komponenten:

- SWebApp-Client
- SWebApp-Server
- Smallworld-GSS-Server
- Smallworld-Applikations-Server
- Smallworld-GIS-Datenbank-Server

Diese 5 Komponenten wurden bereits im Unterabschnitt 2.2.2 beschrieben. Für die Erläuterung der Softwarestruktur sind die Komponenten SWebClient und SWeb-Server von größter Bedeutung und werden anschließend auf ihre softwaretechnischen Aspekte hin untersucht. Abbildung 3.1 ähnelt der Abbildung 2.4, behandelt jedoch nicht den netzwerktechnischen sondern den softwaretechnischen Aspekt.

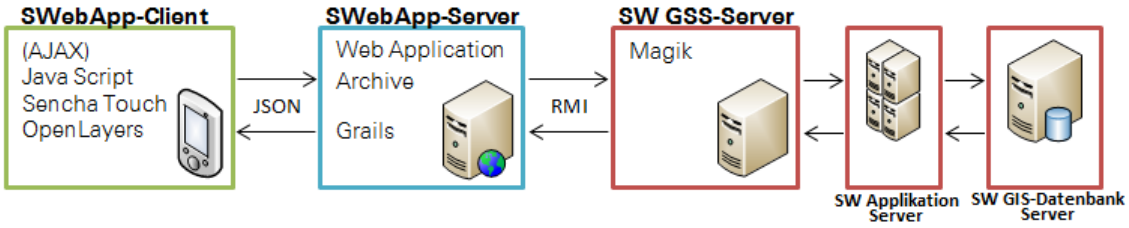


Abbildung 3.1: SWebApps - Softwarestruktur

3.1.1 SWebApp-Server

Bei dem SWebApp-Server handelt es sich um einen Apache Tomcat Webserver, also um einen Open Source Webserver, welcher auf Java Servlets und JavaServer Pages spezifiziert ist. Auf diesem Server wird ein Web Application Framework von Grails

verwendet. Die folgende Definition stammt aus (de.wikipedia.org, 2013b) und soll das Konzept von Grails erläutern.

> *Grails ist ein Web Application Framework für die Programmiersprache Groovy. Grails bietet Konzepte wie Scaffolding, automatische Validatoren und Internationalisierung. Grails ist an Ruby on Rails angelehnt und baut auf mehreren etablierten Frameworks wie Spring, Hibernate und SiteMesh auf und verbindet diese mit der Skriptsprache Groovy.*
>
> *Grundlegend ist das Prinzip von Konvention vor Konfiguration. Artefakte eines bestimmten Typs finden sich zum Beispiel immer in dem gleichen Verzeichnis der Projektstruktur oder bestimmte Elemente haben immer den gleichen Namen. Dies spart Konfigurationsaufwand und erleichtert den Einblick in ein fremdes Projekt. Eine in Grails erstellte Webanwendung lässt sich als Web Archive (WAR-Datei) exportieren und kann so auf jedem Servlet-Container wie zum Beispiel dem Apache Tomcat installiert werden.*

Wie schon in der Definition erwähnt, kann eine in Grails erstellte Webanwendung in eine WAR-Datei exportiert und somit auf dem Apache Tomcat Webserver verwendet werden. Um die Webanwendung möglichst konfigurierbar zu halten, muss als erstes eine *sbwebapp-config.groovy*-Datei erstellt werden. Diese Grundkonfiguration der Webanwendung basiert auf einer Baumstruktur, in welcher bestimmte Regeln beachtet werden müssen. Es können Elemente der Struktur hinzugefügt, entfernt oder auch konfiguriert werden. Diese gibt somit an, welche Komponenten für den Client zur Verfügung stehen.

Der SWebApp-Server kommuniziert einerseits mit dem SWebApp-Client in JSON und anderseits über RMI mit dem Smallworld GSS-Server. Er bekommt also vom Client Anfragen in JSON, auch Requests genannt. Diese werden verarbeitet, übersetzt und an den GSS weiter geleitet. Anschließend sendet der GSS die Antwort auf den Request an den Web-Server. Dieser verarbeitet und übersetzt diese wieder in JSON und übergibt diese an den Client. Bei den Anfragen durch den Client handelt es sich um Services welche über einen Grails-Controller weitergeleitet werden. Die Netzkarte beziehungsweise die visuelle Darstellung des NIS ist ein spezielles Service. Die weitergeleitete Map-Request-Antwort, liefert dabei eine Bilddatei vom GSS. Diese Datei beziehungsweise deren Speicherort wird anschließend vom Webserver an den Client weitergegeben und hier mittels der Javascript-Bibliothek OpenLayers als WMS in die Grundkarte eingebunden.

Der gesamte SWebApp-Server, inklusive erweiterten Grails-Framework, wie Controllers und Services, und die Verbindungen in Richtung Client und GSS wurden von der GRINTEC GmbH im Vorfeld bereits entwickelt und zur Verfügung gestellt. Lediglich die Grails-Konfiguration der verwendeten Komponenten für den Client musste mittels der *sbwebapp-config.groovy*-Datei serverseitig erstellt werden.

3.1.2 SWebApp-Client

Die Client-Software ist eine Webanwendung beziehungsweise hybride Anwendung, siehe Abschnitt 2.4, für mobile Geräte, und benötigt einen auf WebKit basierenden Browser. Beim Start der Client-Anwendung wird auf den SWebApp-Server die Groovy Server Pages-Datei aufgerufen. Diese GSP-Datei erstellt und übermittelt anschließend die, für den Client benötigten, HTML- und JavaScript-Dateien. Somit wird die gesamte Webanwendung für den Client auf dem Server erstellt und anschließend übermittelt. Für die bessere Verständlichkeit von GSP folgt eine Definition aus (en.wikipedia.org, 2013a).

> *Groovy Server Pages (GSP) is a presentation language for web applications, similar to JSP. GSP allows static and dynamic content to be mixed in the same document. The result is a dynamically generated HTML, XML or other type of document in response to a Web client request.*

Die gesamte Webanwendung läuft nun auf dem Client und alle weiteren Requests zum Webserver würden in JSON, ausgenommen Karten-, Down- und Upload-Services, asynchron gesendet und empfangen. Für die Darstellung und Gestensteuerung wurde die User-Interface JavaScript Bibliothek von Sencha Touch integriert, siehe Abschnitt 2.4.2, für die Kartendarstellung und -steuerung die JavaScript Bibliothek OpenLayers.

Prinzipiell gibt es drei Arten von Client-Anwendungen, die vom Webserver an den Client übermittelt werden können, siehe Abbildung 3.2.

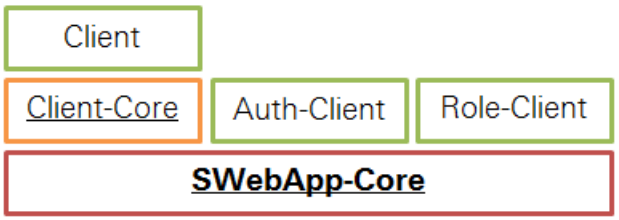


Abbildung 3.2: SWebApps - Clientstruktur

Alle drei Clients bauen auf dem *SWebApp-Core* auf, welcher die Grundfunktionalitäten für diese zur Verfügung stellt. Hierbei zählen zum Beispiel die Grundelemente für die Erstellung eines Formulars, Erstellung und Steuerung von Wizards, die Darstellung der aktuellen Geolocation und weitere. Direkt auf dem Core liegen die zwei Clients *Auth-Client* für die Anwendungsanmeldung und *Role-Client* für die NIS-Rollenauswahl. Des Weiteren gibt es noch den *Client-Core*, welcher unter dem Standardclient *Client* liegt und spezielle Funktionalitäten, wie zum Beispiel die Darstellung der Karte, den Editor zur Anzeige von Attributen von NIS-Objekten, den Down- und Upload von Dateien vom und in das NIS und weitere, für diesen liefert. Der Standardclient *Client* nutzt die Funktionalitäten des *Core* und *Client-Core* und ist somit deren Ausführung. Aus Gründen der modularen Konfigurierbarkeit wurde ein solcher Aufbau gewählt. Es

könnte jetzt zum Beispiel ein weiter Client erstellt werden, welcher auch auf die zwei unteren Ebenen aufbaut, jedoch nicht alle Funktionen beinhaltet.

Die drei Clients und die darunterliegenden Ebenen wurden von der GRINTEC GmbH im Vorfeld entwickelt und für den Prototypen verwendet. Um die speziellen Ansprüche der Applikation zu erfüllen, wie zum Beispiel die Nutzung von NFC oder auch ein speziell konfigurierter Wizard für die Instandhaltungsprozesse, wurden im Zuge der Masterarbeit noch weitere JavaScript-Funktionen entwickelt. Auf diese Funktionen, sowie die spezielle Konfiguration seitens des Servers, wird in den folgenden Abschnitten eingegangen.

3.2 NFC-Integration in hybride Anwendung

Für den Zugriff beziehungsweise die Steuerung der NFC-Systemkomponente des Android-Smartphones wurde die JavaScript-Klasse *nfcIO* erstellt. Die Funktionalität dieser Klasse besteht vor allem aus dem Lesen und Beschreiben von RFID-Transpondern. Diese Funktionen werden sowohl für die Kopplung als auch für die Instandhaltung benötigt. Es ist zu erwähnen, dass für den Prototypen keinerlei Verschlüsslungen von Daten am RFID-Transponder verwendet wurden. Somit sind die auf dem Transponder gespeicherten Daten frei zugänglich. In diesem Abschnitt wird der Aufbau und die Methoden dieser neu entwickelten Klasse beschrieben.

3.2.1 Konfiguration von Grails

Wie bereits in Unterabschnitt 3.1.1 beschrieben, muss Grails serverseitig über die *swebapp-config.groovy*-Datei konfiguriert werden. Hierzu zählt auch das Einbinden von externen JavaScript-Dateien. Die *nfcIO.js* wurde mit folgendem groovy-Befehl in Grails integriert:

Listing 3.1: swebapp-config.groovy: Integration von nfcIO.js

```
1 client.customCode = [
2   javascript: ["download/js/nfcIO.js"]
3 ]
```

3.2.2 Klasse nfcIO und Methoden

Wie bereits in Abschnitt 2.4.1 erklärt wird mittels der JavaScript Bibliothek von Apache Cordova eine hybride Anwendung kompiliert um somit einen Zugriff auf die Hardwarekomponenten des Smartphones, also in diesem Fall auf die NFC-Komponente, zur Verfügung zustellen. Da jedoch Cordova noch keine Funktionalität zum Zugriff auf die NFC-Komponente bietet, wurde zusätzlich ein externes Plugin von

Chariot Solutions eingebunden. Eine Beschreibung dieses Plugins findet man unter **`https://github.com/chariotsolutions/phonegap-nfc`**. Viele Methoden im Listing 3.3 stammen aus dem Cordova Framework beziehungsweise aus dem hinzugefügten Plugin. Diese dienen zum Einschalten der NFC-Komponente am Smartphone und zum anschließenden Lesen und Schreiben auf den RFID-Transponder.

Methode: Read

Folgend wird die RFID-Transponder Lese-Methode gezeigt und beschrieben.

Listing 3.2: nfcIO.js: Read-Methode

```
1  Read: function() {
2    var me = this;
3    var execute =  true;
4    document.addEventListener('deviceready',
5      function(){deviceready();}, false);
6
7    function deviceready(){
8      /** Waiting for a NDEF-Tag */
9      nfc.addNdefListener(nfcRead,
10       function(){success('addNdefListener');},
11       function(){fail('addNdefListener');});
12
13     /**  Waiting for a MimeType */
14     nfc.addMimeTypeListener('text/pg', nfcRead,
15       function(){success('addMimeTypeListener');},
16       function(){fail('addMimeTypeListener');});
17
18     /** success and fail message for nfcListeners */
19     function success(arg){console.log('Started ' +arg);}
20     function fail(arg){Gt.Msg.info('FAILED: Starting ' +arg, false,this);}
21
22     /** fireEvent - Variables tagId and records of NFC Transponder */
23     function nfcRead(nfcEvent){
24       var tag = nfcEvent.tag;
25       var tagId = nfc.bytesToHexString(tag.id);
26       var records = tag.ndefMessage;
27       if (execute == true){me.fireEvent('TagRead',tagId,records);}
28       execute=false;
29 }  }  }
```

Diese Methode wird eine Ebene höher von einem JavaScript *Listener* gestartet, welcher auf den in Zeile 27 befindlichen *fireEvent* wartet. In der Methode werden explizit die zwei von Chariot Solutions zu Verfügung gestellten *addNdefListener* und *addMimeTypeListener*, welche der NFC-Hardwarekomponente des Android-Smartphones mitteilen, dass diese auf die zwei Transponder-Speicherformate Ndef und MimeType hören soll, initialisiert. Bei erfolgreichem Empfangen beziehungsweise Auffinden eines RFID-Transponders im zuvor bestimmten Format, wird die in Zeile 23-28 stehende Funktion aufgerufen. Diese Funktion leitet die gelesene Transponder-ID *tagId* und den Transponder-Inhalt *records* an den übergeordneten Methodenaufruf weiter. Die Variablenübergabe in Zeile 28 steht für die Ausführung der Methode und beendet

den Lesevorgang indem der *fireEvent* nicht mehr exekutiert wird. Diese Variablen-Deklaration und die daraus folgende Abfrage wird in jeder Methode dieser Klasse verwendet und in weiteren Beschreibungen nicht mehr erwähnt.

Methode: Write

Wie auch die Lesemethode wird die Schreibmethode mittels eines übergeordneten *Listeners* aufgerufen. Jedoch wird beim Aufruf ein Array mit den zu schreibenden Daten übergeben. Der Methodenablauf ist identisch wie zuvor, lediglich wird nicht die Funktion *nfcRead* von einem der zwei internen *Listener* aufgerufen, sondern die Funktion *nfcWrite*, welche folgend gezeigt und beschrieben wird.

Listing 3.3: nfcIO.js: Write-Methode

```
1   function nfcWrite(nfcEvent){
2     /** Check if writing already happened */
3     if (execute== true) {
4
5       /** Creating of nfc record set */
6       var newRecords = new Array();
7       for (var i =0;i<data.length;i++){
8         newRecords[i] = ndef.textRecord(data[i]);}
9
10      /** Exection of writing */
11      nfc.write(newRecords,function(){successWrite();}, function(){failWrite();});
12    }
13    execute = false;}
14
15   /** success and fail message for nfcWrite with fireEvent*/
16   function successWrite(){
17     console.log('Success writing on nfc Transponder');
18     me.fireEvent('TagWrite','Schreiben auf RFID-Transponder war erfolgreich',true);}
19
20   function failWrite(){
21     console.log('FAILED: Writing ', false,this);
22     me.fireEvent('TagWrite','Schreiben auf RFID-Transponder war nicht erfolgreich'
23     ,false);}
```

In Zeile 6-8 werden die zu übergebenden Daten in ein für NFC-Transponder verständliches NDEF-Textformat konvertiert. Anschließend wird in Zeile 10-12 der neu erstellte zu übertragende Transponder-Inhalt *newRecord* mittels *nfc.write* auf den RFID-Transponder geschrieben. An den übergeordneten *Listener* des Methodenaufrufes wird bei Erfolg der in Zeile 18 befindende *fireEvent* oder bei Misserfolg der in Zeile 22-23 befindende *fireEvent* zurückgeliefert.

Methode: CheckWrite

Es gibt noch eine dritte Methode in der Klasse *nfcIO*. Diese wird jedoch nur textlich beschrieben, da sie nur für einen speziellen Fall entwickelt wurde und sehr umfangreich ist. Es handelt sich um die Methode *CheckWrite*, welche speziell für die Kopplung

entwickelt wurde. Diese Methode liest als erste die Transponder-ID und überprüft anschließend vor dem Schreiben, ob sich die zuvor eingelesene ID bereits in dem Attribut-Feld *RFID* eines GIS-Objektes befindet. Falls dies der Fall ist, ist ein Kopplung nicht möglich. Diese Überprüfung wurde eingebaut um eine Mögliche doppelte Kopplung des gleichen Transponders zu verhindern.

3.3 Kopplungsprozess

Der Ablauf des Kopplungsprozesses wird in Abschnitt 2.2.3 beschrieben und ist in Abbildung 2.5 erkennbar. Es wurde die vorhandene Client-Funktionalität des *featuresPanel*, welche zur Anzeige von in der Karte ausgewählten GIS-Objekten in einem Editor in Formularform dient, um einen Kopplungsbutton erweitert. Dieser Kopplungsbutton ruft die dazugehörige JavaScript-Funktion auf, welche in Unterabschnitt 3.3.2 erläutert wird.

3.3.1 Konfiguration von Grails

Wie bereits in Unterabschnitt 3.1.1 erläutert, muss Grails serverseitig über die *swebapp-config.groovy*-Datei konfiguriert werden. Als erstes wird die benötigte JavaScript Datei eingebunden, siehe Zeile 2-4. Des Weiteren wird der neue Button in Zeile 7-13 erstellt und mit der richtigen JavaScript Klasse in Verbindung gesetzt und abschließend der *featurePanelToolbar* hinzugefügt, wie in Zeile 16-20 ersichtlich.

Listing 3.4: swebapp-config.groovy: Konfiguration für Kopplung

```
1  // Adding JS class
2  client.customCode = [
3    javascript: [ "download/js/kopp/kopplung_feature_action.js"
4  ]
5
6  // Compontents configuration
7  client.actions = [
8    "gt.kopplungFeature": [
9      JSClassName: "Gt.custom.action.KopplungFeatureAction",
10     itemConfig: [
11       iconCls: 'action',
12       text: 'Kopplung'
13 ] ] ]
14
15 // References
16 swebapp.ref = [
17   featuresPanelToolbar: [
18     itemNames: ['features.popFromViewStack', '->',
19     'gt.kopplungFeature','gt.closeFeature']
20 ] ]
```

3.3.2 Kopplung Feature Action

Da das direkte Eingehen auf den Programmcode in diesem Fall zu unübersichtlich und umfangreich wäre, wird der Ablauf dieser Funktion anhand des folgenden UML-Aktivitätsdiagramm beschrieben.

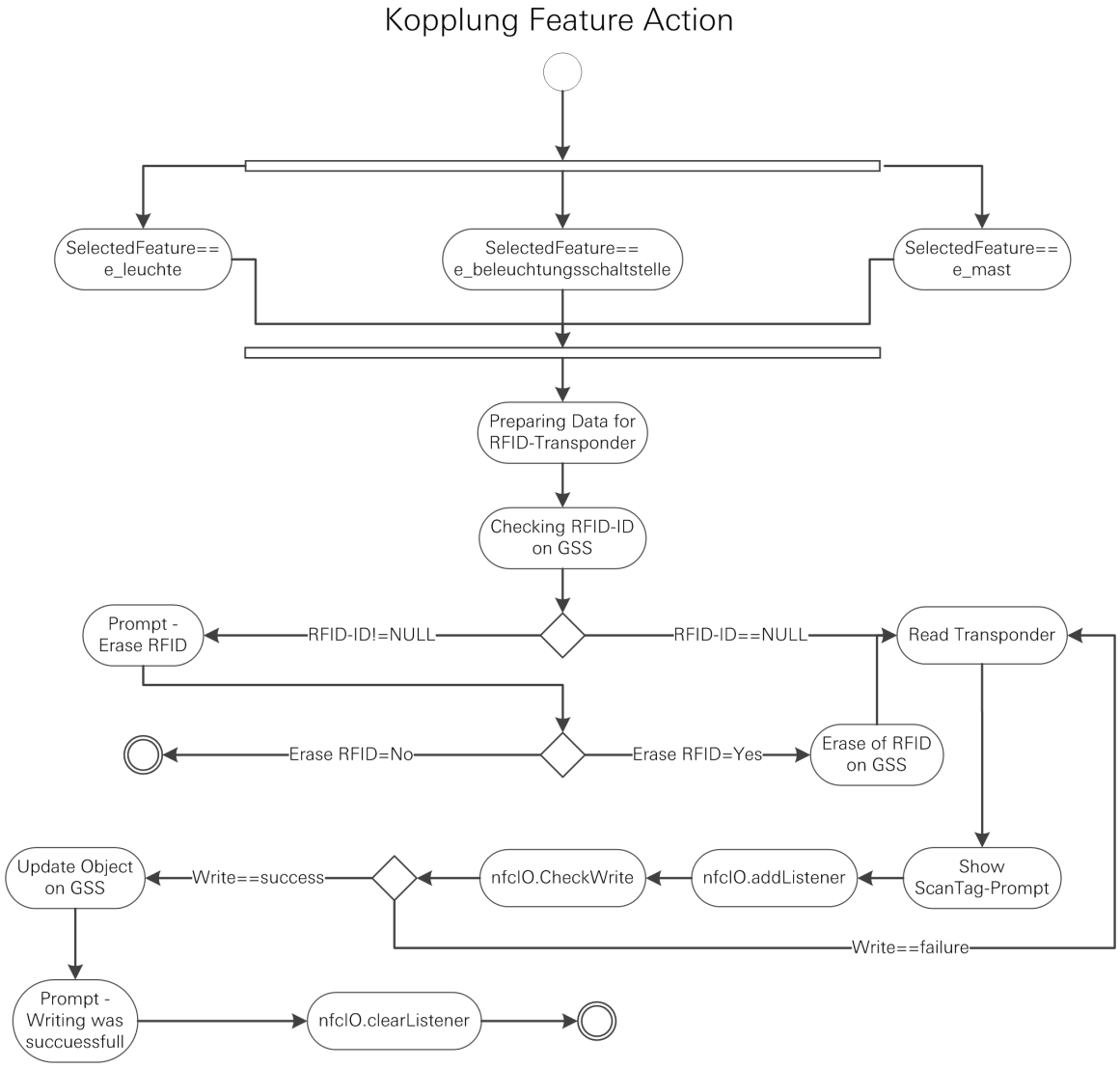


Abbildung 3.3: UML-Aktivitätsdiagramm - Kopplung Feature Action (Code)

Erster Schritt beschäftigt sich mit der Prüfung, ob es sich bei dem aus der Karte ausgewählten Objekt um eine Leuchte, Beleuchtungsschaltstelle oder einen Mast handelt. Nur dann ist der Kopplungsbutton auswählbar. Anschließend werden abhängig vom Objekttyp verschiedene Informationen, welche später auf den Transponder geschrieben werden, vorbereitet. Dann wird überprüft, ob das ausgewählte Objekt bereits eine

eingetragene RFID-Id hat. War diese Überprüfung erfolgreich, das heißt es ist eine ID eingetragenen, kann diese gelöscht werden. War die Überprüfung nicht erfolgreich, keine ID eingetragen, wird der Read-Vorgang gestartet.

Der Benutzer wird mittels einer Prompt-Message aufgefordert das Smartphone an den Transponder zu halten. Der *nfcIO-Listener* und die *CheckWrite-Methode*, siehe Abschnitt 3.2.2, werden initialisiert. War das Schreiben erfolgreich, wird die Transponder-ID in das Attributfeld "RFID"des ausgewählten Objektes eingetragen, der Benutzer per Prompt-Message darüber informiert und der *Listener* mittels *nfcIO.clearListener* gelöscht und somit der Prozess beendet. War der Schreibvorgang nicht erfolgreich, wird der Schritt *Read Transponder* erneut ausgeführt.

3.4 Instandhaltungsprozess

Der Instandhaltungsprozess wurde jeweils für die drei möglichen Objekttypen, Leuchte, Mast und Beleuchtungsschaltstelle, individuell konfiguriert beziehungsweise entwickelt. Da der grundsätzliche Aufbau und die Funktionalitäten sehr ähnlich und teilweise identisch sind, wird folgend beispielhaft die Instandhaltung des Objekttyp Leuchte erläutert. Der Ablauf dieses Prozess kann der Abbildung 2.6 entnommen werden. Prinzipiell ist zu erwähnen, dass bei diesem Prozess ein speziell konfigurierter Wizard verwendet wird. Aus Verständnisgründen folge eine Definition von Wizard aus (en.wikipedia.org, 2013c).

> *A software wizard or setup assistant is a user interface type that presents a user with a sequence of dialog boxes that lead the user through a series of well-defined steps. Tasks that are complex, infrequently performed, or unfamiliar may be easier to perform using a wizard. In contrast, an expert system guides a user through a series of (usually yes/no) questions to solve a problem.*

Es handelt sich somit um eine Art Leitsystem für den Benutzer durch diesen Teil der Anwendung. Die Grundfunktionalitäten des Wizards, wie zum Beispiel um einen Seite nach vorne oder hinten zu wechseln, stammen aus den Client *SWebApp-Core*, siehe Unterabschnitt 3.1.2, und müssen vorab in der *swebapp-config.groovy*-Datei konfiguriert werden.

3.4.1 Konfiguration von Grails

Da die Konfiguration von Grails in diesem Prozess umfangreicher und aufwendiger ist, werden die folgenden Code-Passagen einzeln erläutert.

Hinzufügen von JS-Klassen

Es werden die benötigten JavaScript Klassen beziehungsweise Funktionen hinzugefügt. Da jedem Instandhaltung-Wizard-Seite ein eigenes JavaScript-File zugewiesen wird, ist die Liste umfangreicher.

Listing 3.5: swebapp-config.groovy: Konfiguration für Instandhaltung Leuchte - Hinzufügen von JS-Klassen

```
// Adding JS class
client.customCode = [
  javascript: [
    "download/js/inst/ReadRfidPage.js",
    "download/js/inst/InstMainPage.js",
    "download/js/inst/InstFormPage.js",
    "download/js/inst/InstLampenTauschPage.js",
    "download/js/inst/InstLeuchtenReinigungPage.js",
    "download/js/inst/InstStoerungPage.js",
    "download/js/inst/InstRecordDetailsPage.js",
    "download/js/inst/GotoSpatialContextAction.js",
    "download/js/inst/RelatedDocumentsListAction.js",
    "download/js/inst/InstHistoryPage.js",
    "download/js/inst/InstUpdate.js",
]   ]
```

Hinzufügen von Instandhaltung im Menü

Da die Instandhaltung über das Menü-Panel aufgerufen wird, muss der dementsprechende Punkt hinzugefügt werden.

Listing 3.6: swebapp-config.groovy: Konfiguration für Instandhaltung Leuchte - Hinzufügen zum Menü-Panel

```
// References
swebapp.ref = [
  menuPanelMenu: [
    itemNames: ['startInst','client.logout'],
      items: [
        startInst: [
          action: 'component.showPanel',
          actionConfig: [
            componentName: "inst"
          ],
          itemConfig: [
            text: 'Instandhaltung'
]  ]  ]  ]  ]
```

Konfiguration des Wizard

Diese Konfiguration ist sehr umfangreich und kann daher nicht vollständig gezeigt werden und wurde auf die wichtigsten Punkte minimiert. Es ist vor allem darauf zu achten, welche JavaScript-Klasse von welcher Komponente beziehungsweise *Page*

verwendet wird. In Abbildung 3.4 ist der Aufbau beziehungsweise Ablauf des Wizards zuerkennen.

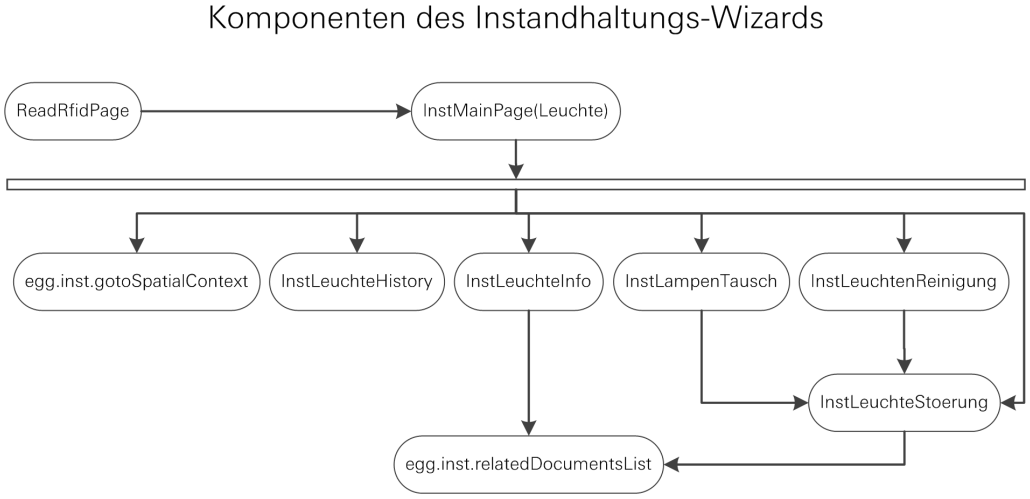


Abbildung 3.4: UML-Aktivitätsdiagramm - Komponenten des Instandhaltungs-Wizards

In Zeile 3 des nachfolgenden Listings wird die Instandhaltung *inst* zu der Client-Anwendung hinzugefügt. Zeile 5 bis 48 zeigen die Bestandteile von dieser Instandhaltungskomponente. Es handelt sich hierbei, wie bereits zuvor erwähnt, um die Konfiguration des Wizards.

Es wird bekannt gegeben, dass beim Öffnen des Wizards, als erstes die Seite *readRFID* geladen werden soll, welche den Benutzer auffordert den RFID-Transponder zu scannen. Nach erfolgreichem Lesen und Finden des entsprechenden Beleuchtungsobjektes, wird man, in diesem Fall auf die Seite *instLeuchte*, weitergeleitet. In Zeile 24 bis 29 werden für die Toolbar entsprechende *Pages* definiert. Die Aktionen dieser *Pages* werden in Zeile 30 bis 40 beschrieben und verweisen wiederum auf, hier nicht genauer aufgeführte, *Pages* mit eigenen dahinterstehenden JavaScript-Klassen. In Zeile 42 bis 47 werden die einzelnen Wartungstätigkeiten von Leuchten definiert und verweisen wiederum auf darunterliegende *Pages*. Diese Struktur vertieft sich noch um einige Ebenen, wird aber auf Grund von Unübersichtlichkeit hier nicht weiter erläutert.

Listing 3.7: swebapp-config.groovy: Konfiguration für Instandhaltung Leuchte - Komponenten

```
1  // Changes Components content
2  client.componentManager = [
3    componentNames: ['map','features','menu','inst'],
4    components: [
5      inst: [
6        className:
7          "com.grintec.web4sw.configuration.components.ComponentConfig",
8          properties: [
9            panelNames: ["panel"],
10           panels: [
11             panel: [
12               className:
13                 "com.grintec.web4sw.configuration.wizard.WizardPanelConfig",
```

```
14                properties: [
15                  title: 'Instandhaltung',
16                  iconCls: 'settings',
17                  autostart: false,
18                  startPageName: "readRfid",
19                  pages: [
20                    readRfid: [className: "egg.inst.ReadRfidPageConfig"]
21                    instLeuchte: [className: "egg.inst.InstMainPageConfig",
22                      properties: [JSClassName: "egg.inst.InstMainPage",
23                        toolbar: [itemNames: [
24                          "backToPreviousPage",
25                          "->",
26                          "instLeuchteInfo",
27                          "instLeuchteHistory",
28                          "egg.inst.gotoSpatialContext",
29                          "closePanel"],
30                          items: [
31                            backToPreviousPage: [
32                              action: 'wizard.backToPreviousPage'],
33                            instLeuchteInfo: [
34                              action: 'wizard.gotoPage',
35                              actionConfig: [pageName: "instLeuchteInfo"]],
36                            instLeuchteHistory: [
37                              action: 'wizard.gotoPage',
38                              actionConfig: [pageName: "instLeuchteHistory"]],
39                            closePanel: [action: 'panel.close']
40                          ]
41                        ],
42                        titleTemplate: "Wartungstaetigkeiten: Leuchte",
43                          listTemplates: [
44                            instLampenTausch: "Lampentausch",
45                            instLeuchtenReinigung: "Leuchtenreinigung",
46                            instLeuchteStoerung: "Stoerung melden"
47                          ]
48                      ]
49                      ...
50  ] ] ] ] ] ] ] ] ]
```

3.4.2 Instandhaltungs-Klassen

Die in Abbildung 3.4 gezeigten *Wizard-Pages* erhalten ihre Funktionalitäten durch verknüpfte und somit dazugehörige JavaScript-Klassen. Diese Klassen werden folgend einzeln erläutert.

ReadRfidPage - egg.inst.ReadRfidPage

Die Klasse *egg.inst.ReadRfidPage* wird beim Start des Wizard initialisiert und fordert den Benutzer auf einen RFID-Transponder zuscannen. Mittels der gescannten RFID-ID wird das am GSS dazugehörige GIS-Objekt gesucht und gespeichert. Des Weiteren werden die sich am Transponder befindenden Daten, wie Kopplungsdatum und Störmeldung, in einen *DataStorage* gespeichert. Folgend ist der Ablauf dieser Klasse dargestellt. Waren all diese Schritte erfolgreich, wird die Seite*InstMainPage* geöffnet und die folgende Klasse initialisiert.

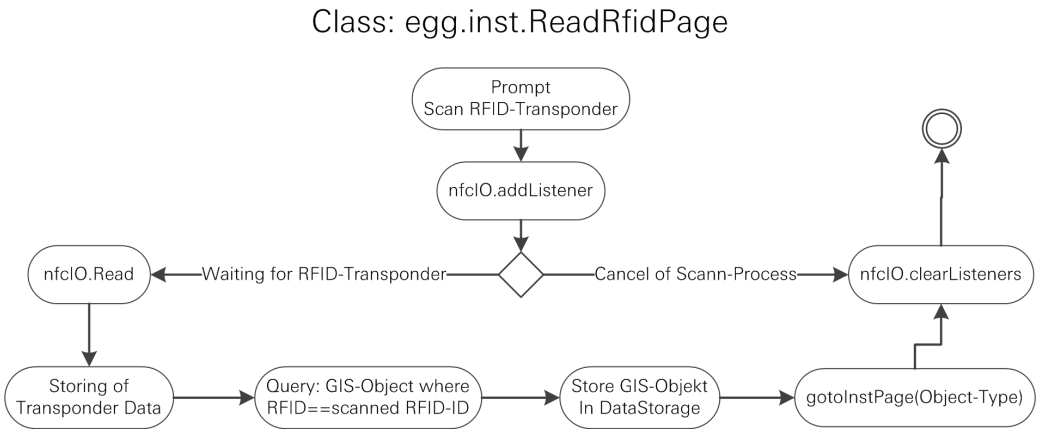


Abbildung 3.5: UML-Aktivitätsdiagramm - egg.inst.ReadRfidPage (Klasse)

InstMainPage - egg.inst.InstMainPage

Die Funktionalität dieser Klasse ist nicht besonders umfangreich. Sie ist im wesentlichen für die Übergabe und Verwaltung des zuvor erstellten *DataStorage* von der darüberstehende Klasse zuständig. Des Weiteren können über dieser *Page* alle weiteren *Pages* aufgerufen werden.

egg.inst.gotoSpatialContext

Diese *Page* hat die gleiche Bezeichnung wie die dazugehörige Klasse. Sie ist für die Kartendarstellung des GIS-Objektes zuständig. Diese Klasse gehört zu der Client-Core-Funktionalität, siehe Unterabschnitt 3.1.2, und wurde von der GRINTEC GmbH zur Verfügung gestellt.

InstLeuchteHistory - egg.inst.InstHisoryPage

Hierbei handelt es sich wiederum um eine Klasse, welche dafür zuständig ist, die zuvor in den *DataStorage* eingelesenen Daten in die dafür konfigurierten Form-Elemente zu schreiben. Die Daten stammen direkt aus dem GIS und vom RFID-Transponder. Diese Daten geben dem Benutzer einerseits Auskunft, wann die letzten Wartungstätigkeiten durchgeführt wurden und anderseits werden die letzten Störmeldungen des ausgewählten Objektes gezeigt.

InstLeuchteInfo - egg.inst.InstRecordDetailsPage und egg.inst.relatedDocumentsList

Auch die Klasse *egg.inst.InstRecordDetailsPage* hat die Aufgabe, die Informationen des *DataStorage* in die dafür vorgesehenen Form-Elementen zu integrieren. Zusätzlich wird die Klasse *egg.inst-relatedDocumentsList*, welche über einen Button in der Toolbar aufgerufen werden kann, eingebunden. Diese Klasse stammt aus der Client-Core-Funktionalität von GRINTEC GmbH und dient zur Anzeige von Verbunddokumenten.

InstLeuchtenStoerung - egg.inst.InstStoerungPage

Die Störungsmeldung wird aus dem aktuellen Datum, dem eingeloggten Benutzernamen, einem Störungsgrund aus einer Dropdown-Liste und einem individuellen Störungstext automatisiert zusammengestellt. Hierbei entsteht eine kurze Störmeldung ohne individuellen Störungstext für den RFID-Transponder und eine umfangreichere Meldung das GIS. Des Weiteren könnte ein Bild zum GIS-Objekt hinzugefügt werden. Da die Störung auch auf dem Transponder gespeichert wird, wird der Benutzer aufgefordert diesen zu scannen. Nach erfolgreichem Scannen wird die Störmeldung in das Attribut-Feld "Bemerkungen" des ausgewählten GIS-Objektes geschrieben. Folgend ist der Ablauf dieser Klasse abgebildet.

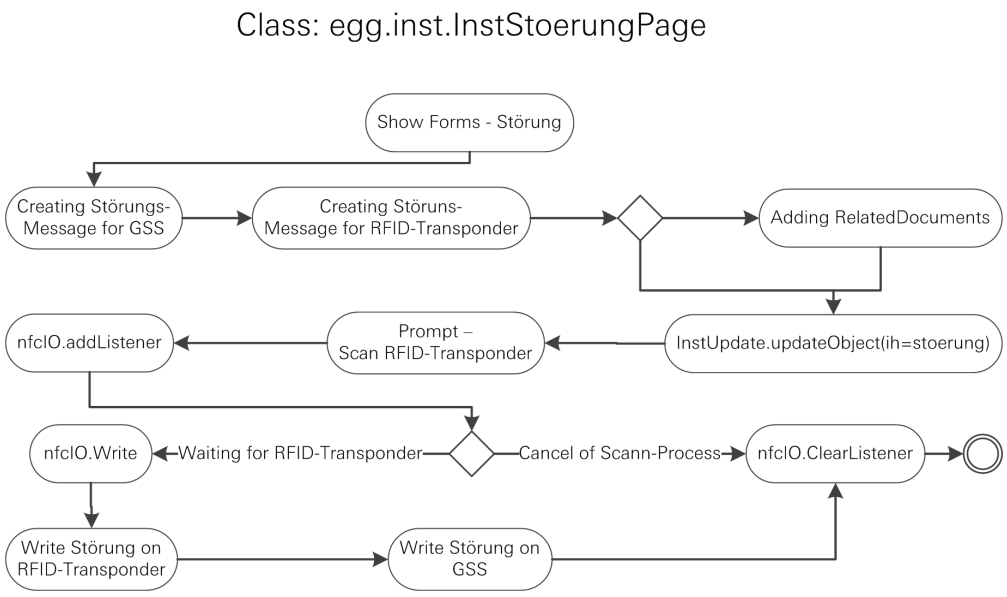


Abbildung 3.6: UML-Aktivitätsdiagramm - egg.inst.InstStoerungPage (Klasse)

instLampenTausch - egg.inst.InstLampenTauschPage

Die Klasse *egg.inst.InstLampenTauschPage* dient zur Überprüfung und Speicherung der Instandhaltungstätigkeit Lampentausch. Es wird überprüft, ob alle Checkbox- und Dropbox-Felder vollständig ausgefüllt wurden. Falls dies nicht zutrifft, wird auf die Seite *InstLeuchteStoerung* weitergeleitet. Wurde alles vollständig ausgefüllt, werden die in Unterabschnitt 2.2.4 aufgeführten Daten zu dem zuvor gescannten und anschließend selektierten GIS-Objekt gespeichert. Diese Informationen werden nicht auf den Transponder geschrieben. In Abbildung 3.7 ist der Ablauf der Klasse zu erkennen.

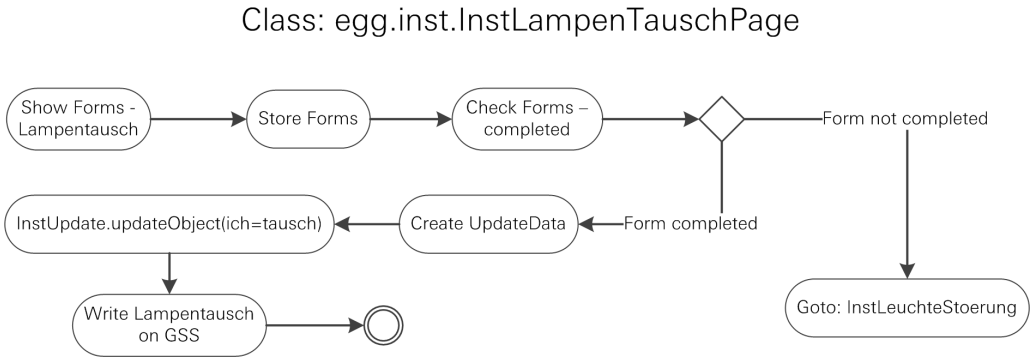


Abbildung 3.7: UML-Aktivitätsdiagramm - egg.inst.LampenTauschPage (Klasse)

instLeuchtenReinigung - egg.inst.InstLeuchtenReinigung

Der Ablauf und die Funktionalität dieser Klasse ist identisch zu *egg.inst.InstLampenTauschPage*. Nur die Attribute, welche in das dazugehörige GIS-Objekt gespeichert werden, sind unterschiedlich und können dem Unterabschnitt 2.2.4 entnommen werden.

3.5 Visualisierung der Betriebsmittel in der Karte

Um dem Benutzer eine Übersicht, welche Wartungstätigkeiten bereits erledigt, welche noch offen und welche überfällig sind, zu bieten, sollte eine Methode entwickelt werden, welche GIS-Objekte je nach Instandhaltungsstatus einfärbt. Diese Vorgang wurde beispielhaft für die Wartungstätigkeit Leuchtenreinigung entwickelt. Es soll nach folgendem Schema eingefärbt werden.

- Rot: Datum der nächsten Reinigung ist vor dem aktuellen Datum
- Orange: Datum der nächsten Reinigung ist nach dem aktuellen Datum
- Grün: Jahr der letzten Reinigung ist gleich wie aktuelles Jahr

Diese Visualisierung wurde in einer für SmallWorld verständliche Zeichenmethode in der Skriptsprache Magik entwickelt und muss beim Start des GSS geladen werden. Nach dem Laden dieser Methode wird der Objekttype Leuchte nach den oben stehenden Regeln eingefärbt. Folgend befindet sich der Code der entwickelten Methode.

Listing 3.8: gt_draw.magik: Zeichenmethode für Leuchtenreinigung

```
#------------------------------------------------------------
# method gt_draw()
#------------------------------------------------------------
_pragma(classify_level=restricted, topic={grintec, e_leuchte})
_method e_leuchte.gt_draw(a_window, a_geometry, a_rwo_style, draw_flag?)
  ## DRAW_FLAG? is a boolean which tells the method if the object is in
  ##       a normal or highlight state (if it is true then you draw it,
  ##             if it is false then you un-draw it).

  ## get acutal gis style
  gis_st << a_rwo_style.actual_gis_style

  _if draw_flag?
  _then
  # wenn naechste reinungung am gesetzt ist (=!= unset)
    _if _self.naechste_reinigung_am _is _unset _orif
    ((_self.naechste_reinigung_am _isnt _unset
      _and _self.naechste_reinigung_am.year > date.now().year )_andif
      (_self.reinigung_am _isnt _unset _and _self.reinigung_am.year <> date.now().year))
    _then
    # wenn keine der bedinungen eintritt, dann normal zeichnen
      a_geometry.draw_physically(a_window, gis_st)
    _else
    #wenn Datum der naechsten reinung schon vorbei ist,
    dann rot sonst orange
    _if _self.naechste_reinigung_am < date.now()
    _then
      set_colour << colour.called("red")
    #wenn das naechste reinigung am jahr das gleiche wie
    heuer ist dann orange
    _elif _self.naechste_reinigung_am.year = date.now().year
    _then
      set_colour << colour.called("orange")
    #wenn das reinigung am jahr das gleiche wie heuer ist dann green
    _elif _self.reinigung_am.year = date.now().year
    _then
      set_colour << colour.called("green")
    _endif

    modified_fill_style << gis_st.copy_with_colour(set_colour)
    a_geometry.draw_on(a_window, modified_fill_style)

  _endif
  _return _true
  _else
    _return _false
  _endif

_endmethod
$
```

4 Ergebnisse und Problematiken

Vorab ist zu erwähnen, dass alle Screenshots direkt auf dem Smartphone gemacht wurden. Das Datum, an dem diese Screenshots gemacht wurden, war der 25.09.2013. Darum beziehen sich alle Anzeigen und Funktionen auf dieses Datum.

4.1 Grundfunktionalitäten SWebApp

Die folgenden Abbildungen zeigen die Grundfunktionen von SWebApp und die Auswahlmöglichkeit von verschiedenen Kartenlayern. In Abbildung 4.1a ist der Anmeldungsscreen zu erkennen. Hier muss sich der Benutzer mit einen im GIS gespeicherten Benutzernamen und Kennwort anmelden.

Abbildung 4.1b zeigt dem Benutzer die Hauptansicht, welche nach erfolgreicher Anmeldung erscheint. In der im Bildschirm oben befindlichen Toolbar findet er in der Reihenfolge von links nach rechts die folgenden Buttons: vorherige Kartenansicht, eigene Position anzeigen, Zoom in, Zoom out, Auswahl der verschiedenen Kartenlayer. Der Kartenlayer wird anschließend im Detail gezeigt und erläutert.

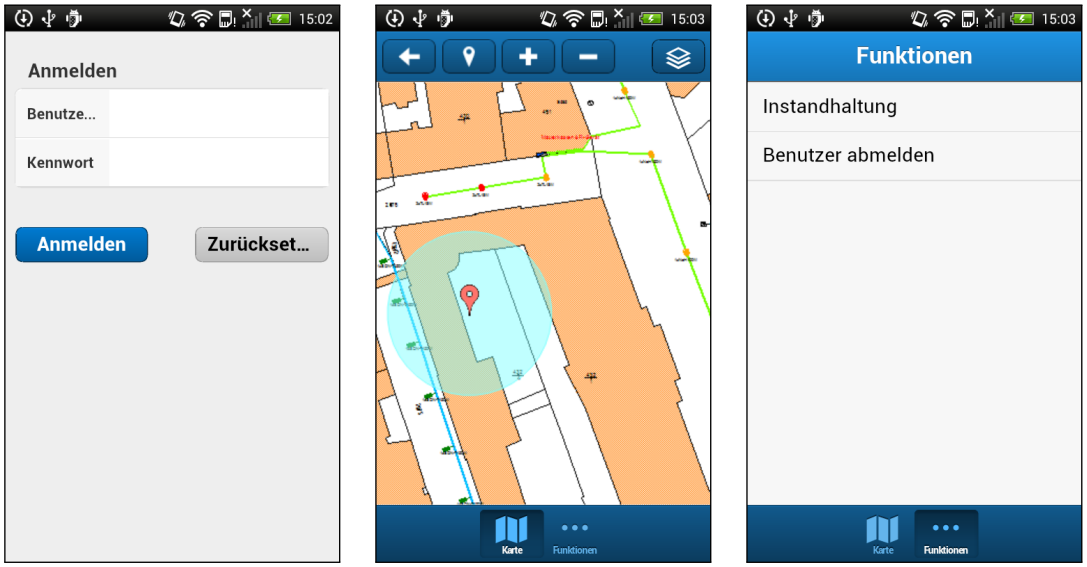
Abbildung 4.1c zeigt das vorkonfigurierte Funktionsmenü. Hier wurden die Buttons "Instandhaltung", dieser wird in Abschnitt 4.3 detailliert erklärt, und der Button "Benutzer abmelden", welcher nach Betätigen wieder zum Screen Abbildung 4.1a führt, hinzugefügt.

Nach Auswahl des sich in der Hauptansicht befindenden Kartenlayer-Buttons erscheint der in Abbildung 4.2a ersichtliche Screen. Hier kann der Benutzer auswählen welche Hintergrundkarten oder Themenkarten angezeigt werden sollen. Des Weiteren kann hier auch das Standortsymbol ausgeblendet werden.

Bei den in Abbildung 4.2b befindlichen Hintergrundkarten handelt es sich um die gängigen Open Street Map-, Google- und Bing-Karten, wobei natürlich nur eine dieser Auswahlmöglichkeiten angezeigt werden kann.

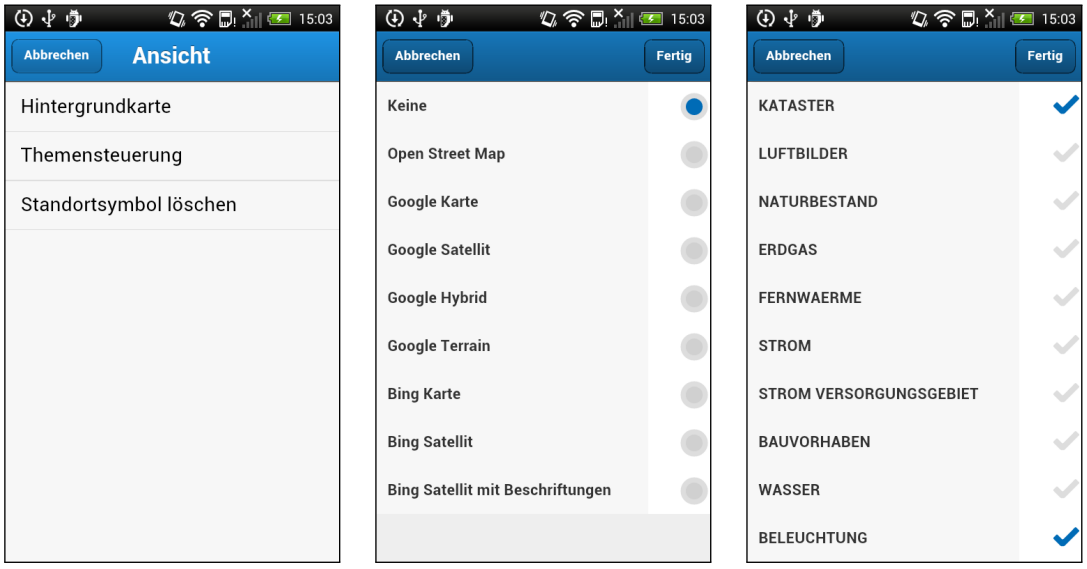
Abbildung 4.2c zeigt die Layer, welche aus dem NIS stammen. Hier können all möglichen Betriebsmittel angezeigt werden. Im Fall dieses speziellen Prototypen sind die Layer "BELEUCHTUNG" und "KATASTER" ausgewählt.

Diese Grundfunktionalitäten stammen aus der Basisentwicklung von SWebApp von der GRINTEC GmbH und wurden für den Prototypen zur Verfügung gestellt. Diese mussten jedoch für die speziellen Anforderungen des Prototypen vorkonfiguriert



(a)Anmeldungsscreen (b)Hauptansicht (c)Funktionen-Menü

Abbildung 4.1: Grundfunktionalitäten SWebApp



(a)Hauptauswahl Layers (b)Auswahl Hinterkarten (c)Auswahl Themenkarten

Abbildung 4.2: Kartenlayer SWebApp

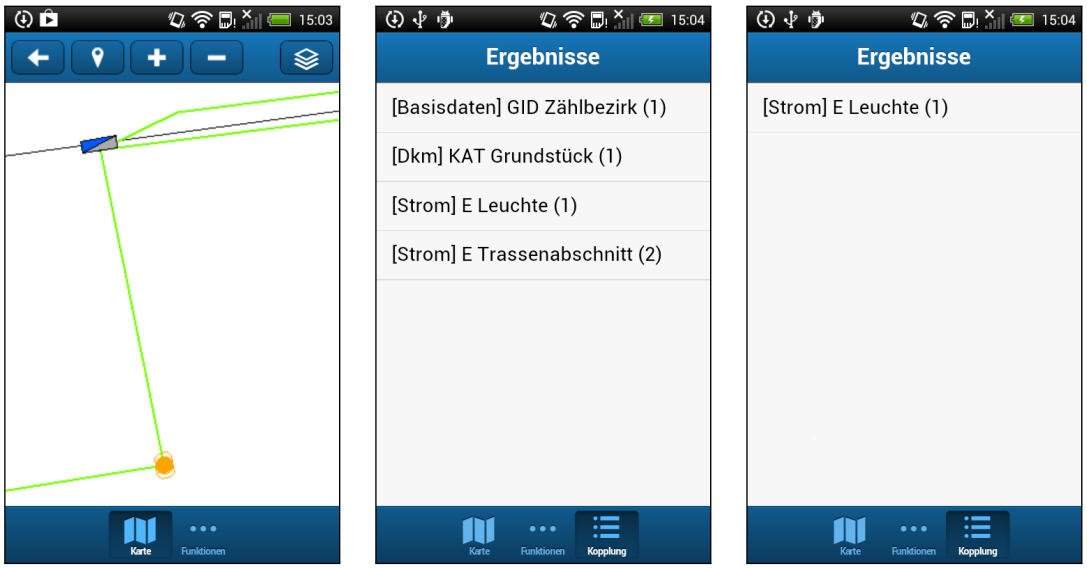
werden, wie bereits in Abschnitt 3.1 erläutert. Es wurden also die Buttons der Toolbar konfiguriert, welche Layer in der Hauptansicht beim Start angezeigt werden und auch welche Auswahlmöglichkeiten im Funktionsmenü erscheinen.

4.2 Kopplungsprozess

Der Kopplungsprozess startet mit der Auswahl des zu koppelnden Beleuchtungsobjektes in der Karte der Hauptansicht. In Falle der Abbildung 4.3a handelt es sich um die in orange gekennzeichnete Leuchte. Für die Positionierung des Smartphones und somit für die einfachere Bestimmung des vor sich befindlichen Betriebsmittels kann die GPS-Position des Smartphones zu Hilfe gezogen werden.

Nach Auswahl des Objektes erscheint derzeit eine Ergebnisliste, siehe Abbildung 4.3b, in welcher alle Objekte auf der Karte im Umkreise von 24 Pixel vom Auswahlpunkt anzeigt werden. Diese Pixelanzahl ist konfigurierbar und wurde in diesem Fall empirisch ermittelt. Wird ein Nicht-Beleuchtungsobjekt ausgewählt, wird im darauffolgenden Screen der Kopplungsbutton nicht frei geschaltet und ist somit funktionslos.

Zukünftig, momentan leider noch nicht einstellbar, sollen nur die drei Beleuchtungsobjekte Leuchte, Mast und Beleuchtungsschaltstelle angezeigt werden, wenn sie sich innerhalb des 24 Pixel Auswahlradius befinden. Diese mögliche Ergebnisansicht ist in Abbildung 4.3c zu erkennen. Somit wäre die Auswahl eines anderen GIS-Objektes für die Kopplung nicht möglich.



(a)Auswahl auf Karte

(b)Derzeit: Auswahl in Ergebnisliste (alle Betriebsmittel)

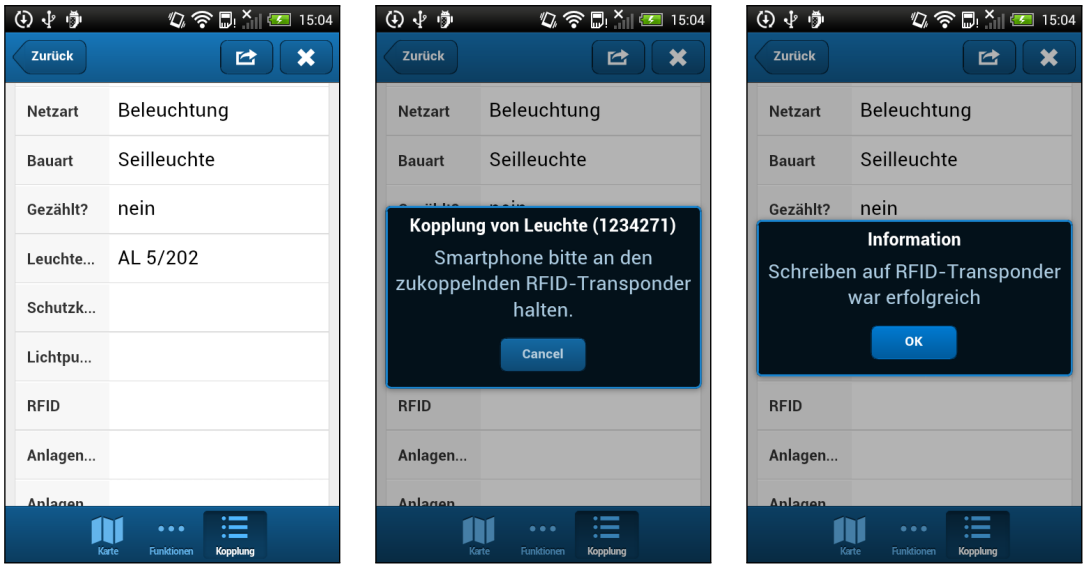
(c)Zukünftig: Auswahl in Ergebnisliste (nur Beleuchtungsmittel)

Abbildung 4.3: Kopplungsprozess (1)

Nach Auswahl des Betriebsmittels "Leuchte" erscheinen einige Informationen dieses Betriebsmittels direkt aus dem GIS. Es kann unter anderem nachgeschaut werden, ob eine "RFID" bereits eingetragen wurde und die Leuchte somit bereits gekoppelt ist, wie in Abbildung 4.4a erkennbar ist. Will der Benutzer trotz eingetragener "RFID" die Kopplung durchführen, wird er darauf hingewiesen und gefragt, ob die eingetragene RFID gelöscht werden soll. In der Toolbar des Kopplungsscreens, findet man einen "Zurück"-Button, um wieder zur Objektauswahlliste zu gelangen, einen "Kopplungs"-Button und einen "Schließen"-Button.

Nach Auswahl des "Kopplungs"-Button erscheint eine Prompt-Meldung, welche den Benutzer auffordert den RFID-Transponder zu scannen, wie in Abbildung 4.4b erkennbar. Wird der "Cancel"-Button der Prompt-Meldung betätigt, wird der Kopplungsprozess abgebrochen.

Wurde der RFID-Transponder erfolgreich gescannt, erscheint eine dementsprechende Prompt-Meldung, siehe Abbildung 4.4c. War das Schreiben auf den Transponder nicht erfolgreich, wird der Benutzer darauf hingewiesen.



(a) Anzeige von ausgewählten Betriebsmittel

(b) Aufforderungen zum Scannen

(c) Erfolgreiches Scannen

Abbildung 4.4: Kopplungsprozess (2)

4.3 Instandhaltungsprozess

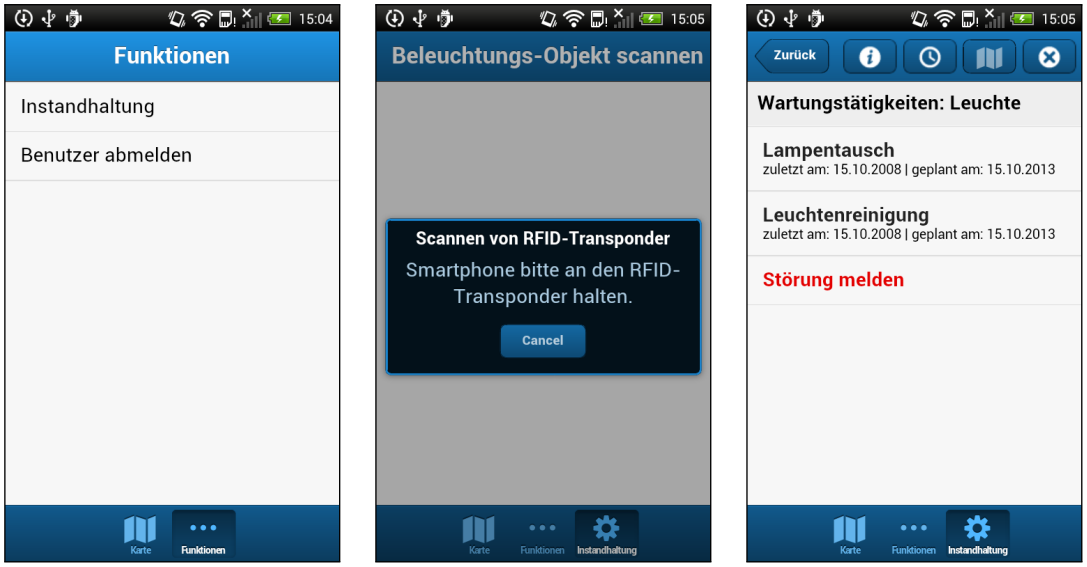
4.3.1 Grundelemente

Folgend werden die Grundelemente von "Instandhaltung" gezeigt. Wichtig ist, dass der Instandhaltungs-Hauptscreen je nach Beleuchtungsobjekt andere Wartungstätigkeiten anzeigt. In diesem Fall wird beispielhaft das Beleuchtungsobjekt "Leuchte" gezeigt.

Der Instandhaltungsprozess wird über Funktionen → Instandhaltung gestartet, siehe Abbildung 4.5a.

Im darauffolgenden Screen, in Abbildung 4.5b erkennbar, wird der Mitarbeiter aufgefordert den am Beleuchtungsobjekt befindlichen RFID-Transponder zu scannen.

Nach erfolgreichem Scannen des Transponders wird der Instandhaltungs-Hauptscreen für die gescannte Leuchte gezeigt, siehe Abbildung 4.5c. In der Toolbox findet man einen "Zurück"-, einen "Informationen"-, einen "History"-, einen "Karten"- und einen "Schließen"-Button. Im Hauptfenster des Instandhaltungs-Hauptscreens befinden sich die "Wartungtstätigkeiten" und auch die Möglichkeit eine Störung zu melden.



(a) Starten des Instandhaltungsprozesses

(b) Aufforderungen zum Scannen

(c) Nach erfolgreichen Scannen-Hauptscreen (Leuchte)

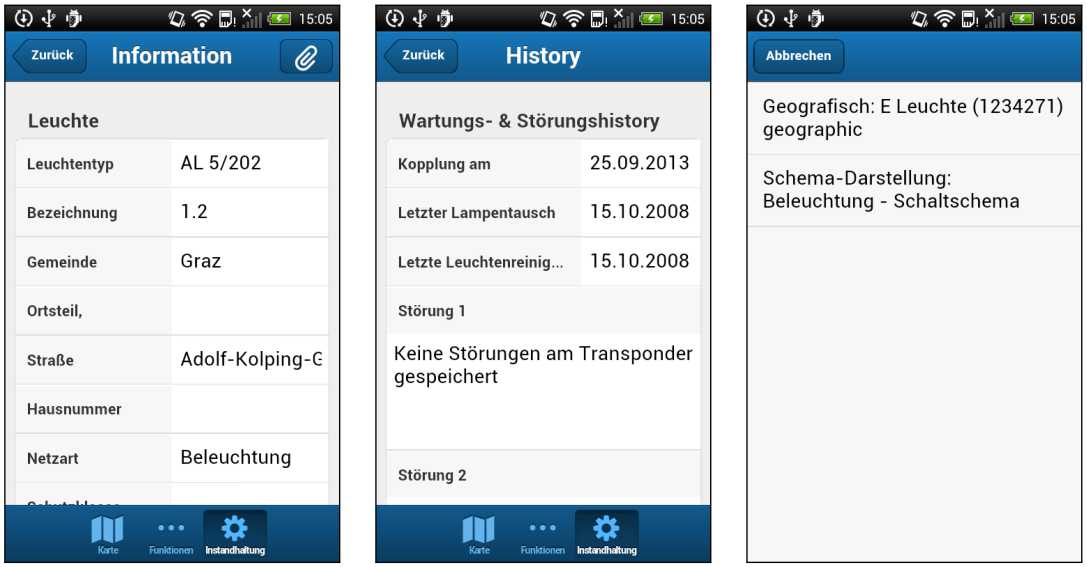
Abbildung 4.5: Instandhaltung-Grundelemente (1)

Folgend werden die verschiedenen, vom Instandhaltungs-Hauptscreen erreichbaren, Seiten erläutert und gezeigt.

In Abbildung 4.6a sieht man die Seite, welche beim Betätigten des "Information"-Buttons geöffnet wird. Sie zeigt die allgemeinen Informationen der Leuchte. Rechts in der Toolbox besteht auch die Möglichkeit sich die Verbunddokumente, hierbei handelt sich um Dokumente, welche an dem GIS-Objekt angehängt sind, des Betriebsmittels anzeigen zulassen. Mit dem "Zurück"-Button gelangt man wiederum zum Instandhaltungs-Hauptscreen.

Abbildung 4.6b zeigt die History-Seite und somit Informationen darüber wann die Kopplung stattgefunden hat, wann die letzten Wartungstätigkeiten durchgeführt wurden und die letzten drei Störmeldungen.

Mit Auswahl des "Karten"-Buttons gelangt man auf die Seite, welche in Abbildung 4.6c erkennbar ist. Hier besteht die Möglichkeit das GIS-Objekt in verschiedenen Darstellungen anzeigen zulassen.



(a)Instandhaltung (Leuchte) - Information

(b)Instandhaltung (Leuchte) - History

(c)Instandhaltung (Leuchte) - Karte

Abbildung 4.6: Instandhaltung-Grundelemente (2)

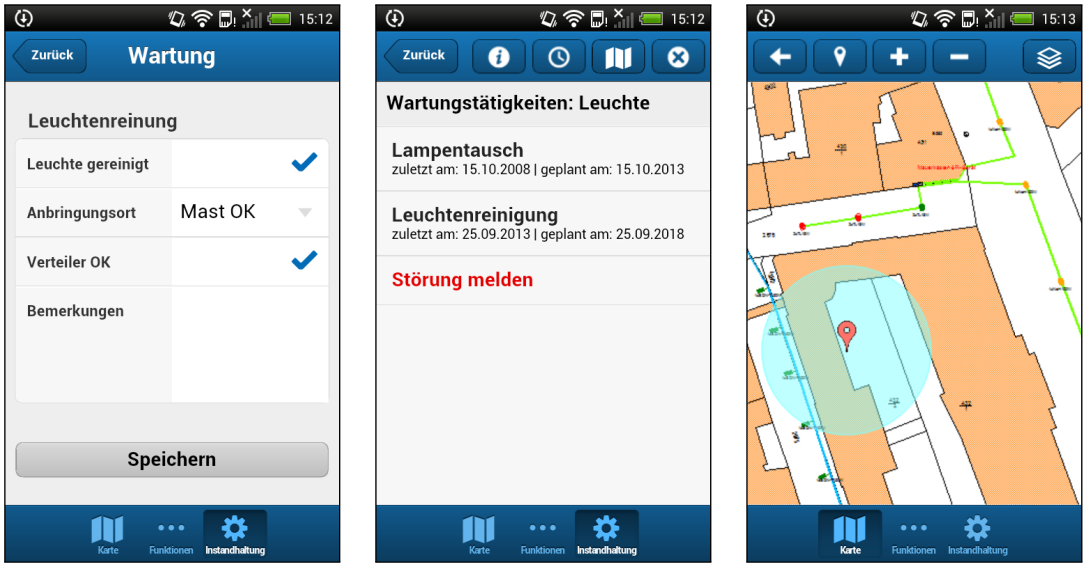
4.3.2 Wartungstätigkeiten

Dieser Abschnitt zeigt die erfolgreiche Durchführung der Wartungstätigkeit "Leuchtenreinigung". In diesem Beispiel wird auch die Echtzeitfähigkeit und die direkte Verbindung zum GIS ersichtlich. In Abbildung 4.5c ist zu erkennen, dass die letzte Reinigung am 15.10.2008 stattgefunden hat und das diese bis 15.10.2013 wieder durchgeführt werden soll. Aus diesem Grund hat das Leuchten Symbol in Abbildung 4.1a, die Farbe orange. Eine Erläuterung findet man in Abschnitt 3.5.

Nun wird laut Abbildung 4.7a die Reinigung vollständig durchgeführt. Die Leuchte wird gereinigt, der Mast und auch der Verteiler als "in Ordnung" beurteilt.

Nach dem Speichern gelangt der Benutzer wiederum auf den Instandhaltungs-Hauptscreen und erkennt, wie in Abbildung 4.7b ersichtlich, dass das Datum der letzten Reinigung auf das "heutige" Datum, also 25.0.2013, geändert wurde und automatisch das Datum der nächsten Reinigung auf 25.09.2018, also in 5 Jahren, gesetzt wurde.

Wenn der Mitarbeiter nun die Karte aktualisiert, siehe Abbildung 4.7c, ist erkennbar, dass sich die Farbe der Leuchte von orange auf grün, das heißt die Tätigkeit wurde im heurigen Jahr erledigt, geändert hat. Diese Visualisierung wurde derzeit nur für die Wartungstätigkeit "Leuchtenreinigung" realisiert.



(a)Wartungstätigkeit: Leuchtenreinigung

(b)Leuchtenreinigung mit neuem Datum

(c)Visualisierung der Wartungstätigkeit

Abbildung 4.7: Wartungstätigkeit: Leuchtenreinigung - erfolgreich abgeschlossen

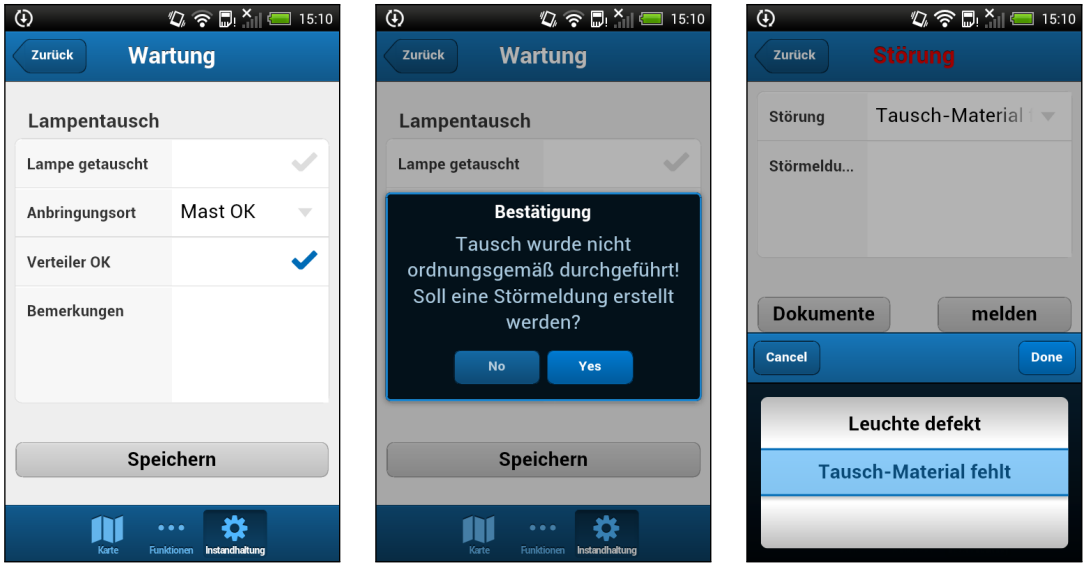
4.3.3 Störung

Störung aus Wartungstätigkeiten

Nun wird die Wartungstätigkeit "Lampentausch" ausgewählt, jedoch nicht vollständig durchgeführt. Das Checkbox-Feld "Lampe getauscht" wird nicht abgehakt, da zum Beispiel nicht die richtige Lampe vorhanden ist, wie in Abbildung 4.8a ersichtlich.

Nach betätigen des "Speichern"-Buttons, wird der Benutzer darauf hingewiesen, dass der Tausch nicht ordnungsgemäß durchgeführt wurde, und ob dieser eine Störmeldung erstellen will, siehe Abbildung 4.8b.

Wird dies bestätigt, wird man auf die Störungsseite weitergeleitet, siehe Abbildung 4.8c, und kann aus einer vorgefertigten Liste den Störmeldungsgrund auswählen.



(a)Wartungstätigkeit: Lampentausch

(b)Abfrage: Störung melden

(c)Störung aus Wartungstätigkeit - Dropdown-Box

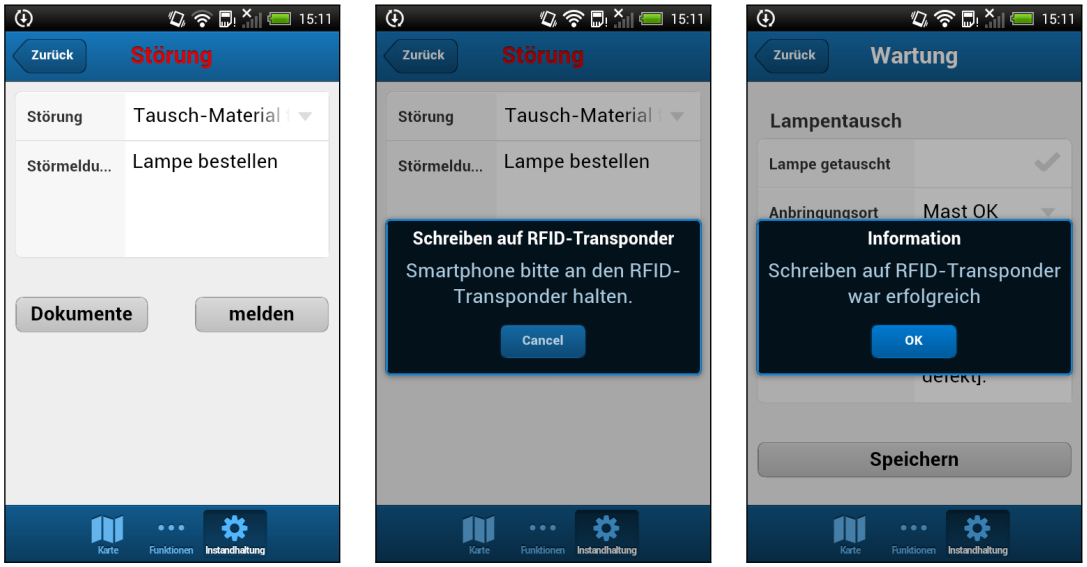
Abbildung 4.8: Störung aus Wartungstätigkeit: Lampentausch (1)

Abschließend wird noch ein individueller Störmeldungstext eingegeben, siehe Abbildung 4.9a, und es könnte ein Dokument, wie zum Beispiel ein Foto, hinzugefügt werden.

Nach Betätigen des "melden"-Buttons, wird der Benutzer aufgefordert den RFID-Transponder zu scannen, wie in Abbildung 4.9b ersichtlich. Die Störmeldung wird einerseits in das Feld "Bemerkungen" des ausgewählten GIS-Objektes und auch auf den RFID-Transponder geschrieben.

Nach erfolgreichem Schreiben auf den Transponder wird der Mitarbeiter darüber informiert, siehe Abbildung 4.9c.

Die Auswirkung einer Störmeldung beziehungsweise wo diese zu finden ist, wird etwas später in diesem Kapitel gezeigt.



(a)Störung aus Wartungstätigkeit - Störungstext

(b)Aufforderungen zum Scannen

(c)Erfolgreiches Schreiben auf RFID-Transponder

Abbildung 4.9: Störung aus Wartungstätigkeit: Lampentausch (2)

Explizite Störmeldung

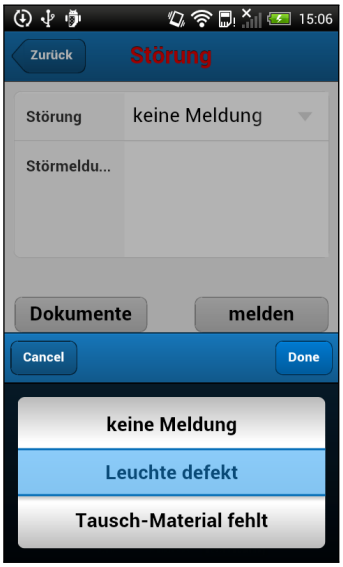
Nun wird die explizite Störmeldung schriftlich und bildlich beschrieben. Nachdem der Benutzer am Instandhaltungs-Hauptscreen, siehe Abbildung 4.5c, den Punkt "Störung melden" wählt, erscheint die auf Abbildung 4.10a ersichtliche Störungsseite, auf welcher er nun einen Störmeldungsgrund wählen kann.

Mit einem Klick auf den "Dokumente"-Button, besteht die Möglichkeit Dokumente, in diesen Fall Bilder, hinzu zufügen. Er muss zwischen neues Bild mit der Kamera aufnehmen oder Bild aus Album wählen, wie in Abbildung 4.10c ersichtlich.

In diesem Fall wurde Kamera gewählt und ein neues Foto von der defekten Leuchte gemacht und hinzugefügt, siehe Abbildung 4.11a und Abbildung 4.11b.

Abschließend gelangt man wieder zur Störungsseite und kann mittels "melden"-Button die Störung speichern. Hier nicht in Screenshots angeführt, wird der Benutzer wiederum aufgefordert den RFID-Transponder zu scannen und die Daten werden zum dementsprechenden GIS-Objekt und Transponder hinzugefügt. Außerdem wird zum GIS-Objekt das Foto als Verbunddokument angehängt.

(a)Störung aus Wartungstätigkeit - Störungstext

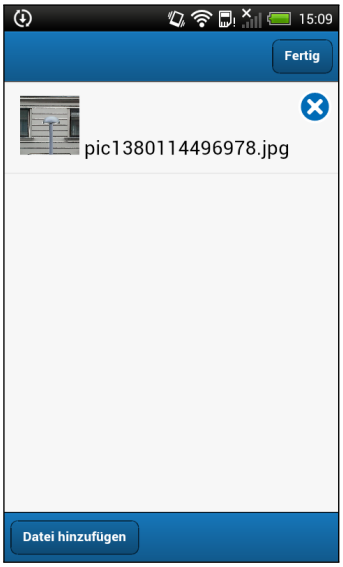


(b)Auswahl des Störungsgrundes

(c)Auswahl neues Foto oder Album

Abbildung 4.10: Explizite Störmeldung (1)

(a)Foto mit Kamera



(b)Foto der Kamera in Dokumente

(c)Erfolgreiches Schreiben auf RFID-Transponder

Abbildung 4.11: Explizite Störmeldung (2)

Zeigen von Störmeldungen

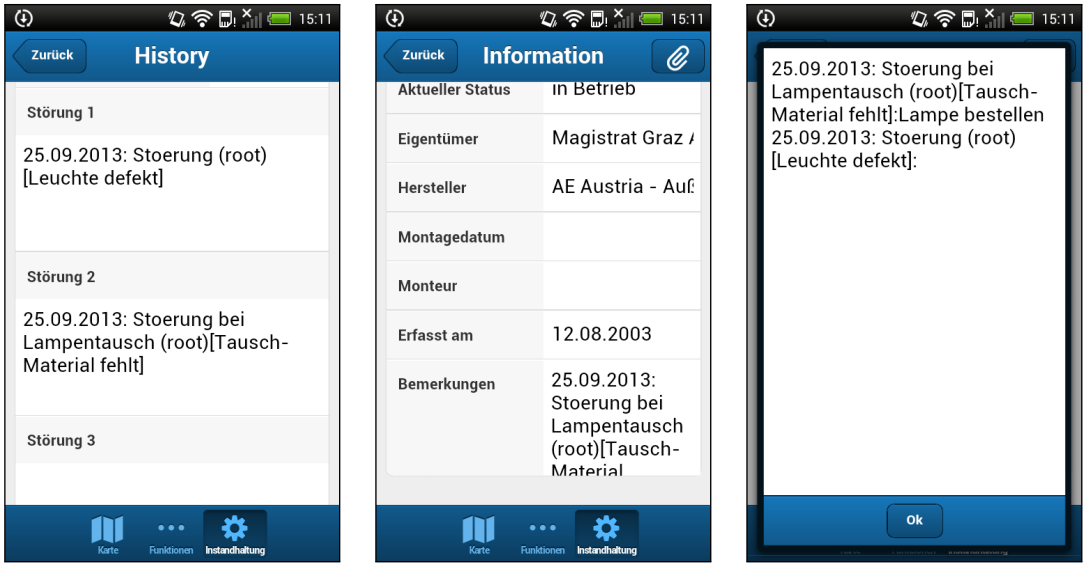
Wie bereits zuvor erwähnt, werden die Störmeldungen einerseits auf dem RFID-Transponder und anderseits auch zu dem dementsprechenden Objekt im GIS gespeichert.

Aufgrund von endlichen Speicherplatzes am Transponder wird auf diesen nur folgende Information geschrieben:

- Datum der Störung
- Ist Störungmeldung aufgrund fehlgeschlagene Wartungstätigkeit oder Explizit
- Login-Name des Mitarbeiters, welcher die Störung gemeldet hat
- Störmeldungsgrund aus Dropdown-Liste

Eine umfangreichere Störmeldung findet man im "Bemerkungen"-Feld des GIS-Objektes. Hier werden zusätzlich der individuelle Störmeldungsgrund und auch eventuelle Fotos gesichert. Diese Meldung findet man in den Informationen des Beleuchtungsobjektes, siehe Abbildung 4.12a. Mit einen Klick auf das Textfeld sieht der Benutzer den gesamten Text, wie auch in Abbildung 4.12c erkennbar ist.

Die am Transponder gesicherte Störmeldung findet man in der Beleuchtungsobjekt-History, siehe Abbildung 4.12a.



(a)Störmeldung am RFID-Transponder - History

(b)Störmeldung im GIS - Information

(c)Störmeldung im GIS - Gesamtansicht

Abbildung 4.12: Zeigen von Störmeldungen (2)

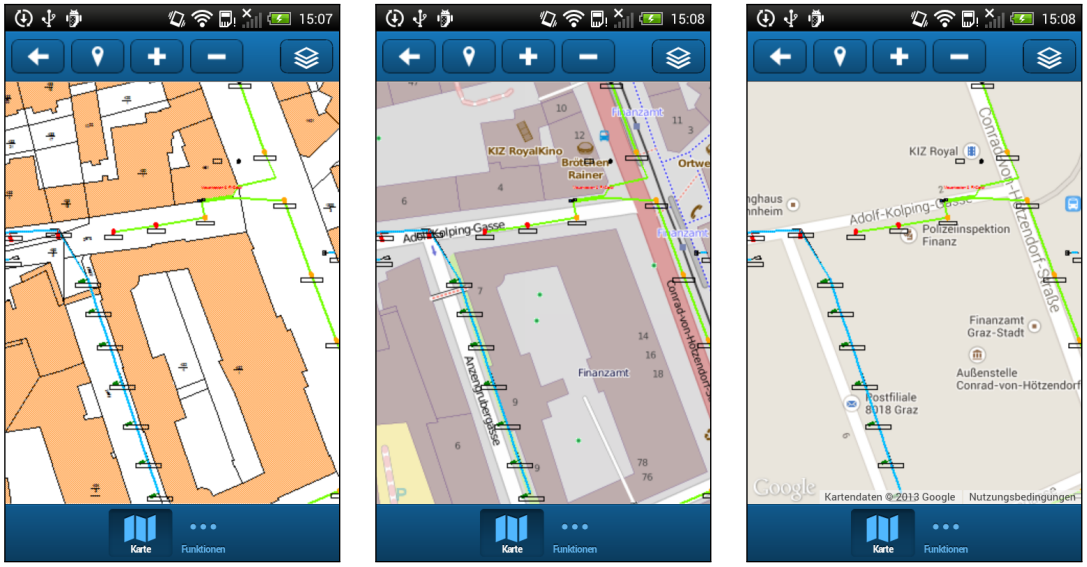
4.4 Visualisierung der Betriebsmittel in der Karte

Es lassen sich, wie bereits zuvor erwähnt, verschiedene Hintergrundkarten und alternativ dazu Kataster- und Naturbestandsdaten des Netzbetreibers einblenden. Deshalb werden in Abbildung 4.13 die Betriebsmittel in der Karte ohne Hintergrundkarte, jedoch mit Kataster, mit der Open Street Map Hintergrundkarte ohne Kataster und Google Maps Hintergrundkarte ohne Kataster gezeigt.

Die Open Street Map in Abbildung 4.13b ist auf ihrem maximalen Zoom-Level, also größtmöglichen Maßstab, und deshalb können die Beleuchtungssymbole nicht größer angezeigt werden. Dies ist sehr hinderlich für die Identifizierung und Auswahl des korrekten Beleuchtungsobjektes bei der Koppelung.

Die Google Map könnte noch in größeren Maßstab angezeigt werden. Jedoch ist, wie bereits in Abbildung 4.13c erkennbar, ein Versatz zwischen Betriebsmittel und Hintergrundkarte zuerkennen, welcher sich dadurch noch vergrößert. Dies liegt vielleicht auch daran, dass Google die Straßen als Linienobjekte darstellt.

Der Nachteil von einer Anzeige der Betriebsmittel ohne Hintergrundkarte, wie in Abbildung 4.13a dargestellt, ist, dass keine Straßenbezeichnungen angezeigt werden und darum die Orientierung schwer fällt.



(a)ohne Hintergrundkarte mit Kataster

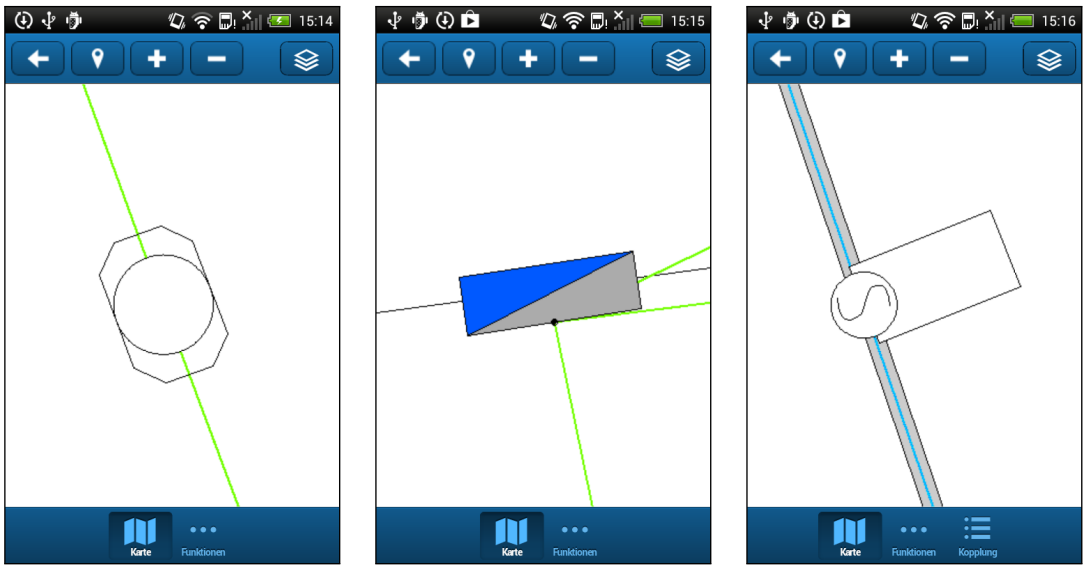
(b)Hintergrundkarte: Open Street Map

(c)Hintergrundkarte: Google Maps

Abbildung 4.13: Visualisierung - Hintergrundkarten

Wie bereits in den vorherigen Kapiteln erwähnt, wurde eine beispielhafte spezielle Visualisierung für das Beleuchtungsobjekt Leuchte beziehungsweise den Leuchtenreinigungs-Zustand erstellt, siehe Abschnitt 3.5. Diese Einfärbungen sind in Abbildung 4.13a erkennbar.

Folgend sieht man eine Auswahl der drei verschiedenen Beleuchtungsobjekte im Detail, um die Symbolik ein wenig näher zu bringen. Dieser sehr große Maßstab ist nur Möglich, wenn keinerlei Hintergrundkarten, ausgenommen Kataster, eingeblendet sind und erleichtert die Auswahl des richtigen Betriebsmittel auf einem kleinen Display.



(a)Symbolik - Leuchte auf Seil

(b)Symbolik - Beleuchtungsschaltstelle

(c)Symbolik - Mast mit Leuchte

Abbildung 4.14: Symbolik - Beleuchtungsobjekte

5 Zusammenfassung und Ausblicke

5.1 Rückblick auf Aufgabenstellung

Ziel war es, im Umfeld eines Wirtschaftsunternehmens das Arbeitsgebiet Netzinformationssystem, die Entwicklung einer mobilen und kartenunterstützten Anwendung kennenzulernen und durchzuführen und schlussendlich die Außendiensttätigkeiten eines EVU mittels dieser Anwendung zu unterstützen und zu optimieren. Jedoch sollte das Produkt, also der Prototyp, so entwickelt werden, dass er für jede beliebige Anforderung geändert und eingesetzt werden kann. Nach einer dreimonatigen Projektplanungs-, einer fünfmonatigen Entwicklungs- und einer einmonatigen Test-Phase kann behauptet werden, dass diese Ziele verwirklicht wurden. Natürlich gab es zeitliche Überschneidungen dieser drei Phasen.

Folgend werden die drei Kernziele rückblickend betrachtet.

5.1.1 Projektplanung

Die Projektplanung bestand aus einer Ist- und Soll-Zustandsanalyse. Die Ist-Zustandsanalyse beinhaltete einige Treffen mit der Abteilung für öffentliche Beleuchtung der Energie Graz GmbH & Co KG um ihre Tätigkeiten in Prozesse zu gliedern und anschließend detailliert auf diese einzeln einzugehen. Die Prozesse wurden mittels UML-Diagrammen visualisiert. Für die Entwicklung dieser Diagramme waren detaillierte Hintergrundinformationen Voraussetzung und es wurde dadurch konkret auf fehlende Information hingewiesen.

Die Soll-Zustandsanalyse setzte einen vollständige Ist-Zustandsanalyse voraus und sollte die beschriebenen Prozesse durch Einsatz von neuen Technologien, in diesem Fall eine mobile und kartenunterstützte Applikation in Verbindung mit auf Beleuchtungsobjekten angebrachten RFID-Transpondern, optimieren. Konkret ging es darum, unzählige Auftragslisten durch digitale Formulare abzulösen und somit Verwaltungsaufwand zu verringern und für den Außendienstmitarbeiter eine übersichtliche Zielführung seiner Tätigkeit zu gewährleisten. Des weiteren wird nun die Planung der Wartungstätigkeiten durch automatisiert erstellte Wartungstermine und dementsprechende Visualisierung im GIS unterstützt. Störmeldungen können direkt mit der neuen Anwendung erstellt und mit Fotos versehen werden. Es wurde der Prozess "Kopplung" neu geschaffen, welcher die Verbindung zwischen GIS und an den Beleuchtungsobjekten befindlichen RFID-Transpondern verwirklicht.

5.1.2 Prototyp-Entwicklung

Der Prototyp wurde möglichst modular und konfigurierbar entwickelt. Vorab wurde analysiert, welche Art von mobiler Applikation, siehe Unterabschnitt 2.4.1, entwickelt werden soll. Wie bereits in den Kernzielen aufgezählt, wurden die Prozesse

- Erst Kopplung von auf Betriebsmittel befindenden RFID-Transponder mit dazugehörigem GIS-Objekt
- Abwicklung von Instandhaltungstätigkeiten mittels mobilem Gerät
- Visualisierung der Betriebsmittel abhängig von Instandhaltungsaufgaben

in die Applikation vollständig integriert. Die Visualisierung wurde nur beispielhaft für die Instandhatlungstätigkeit "Leuchtenreinigung" durchgeführt, kann jedoch nach Belieben für jede weitere Tätigkeit integriert werden.

5.1.3 Usability-Test und Fehler- beziehungsweise Problembehebung

Der Usability-Test der Anwendung wurde aufgrund eigener Erfahrung und Feedbacks der GRINTEC GmbH und Energie Graz GmbH & Co KG durchgeführt und erfolgreich abgeschlossen. Die Fehler- beziehungsweise Problembehebung wurde fortlaufend und nach Entwicklung des Prototypen durchgeführt und alle lösbaren Fehler wurden behoben. Die noch offenen Fehler und Probleme werden in dem folgenden Abschnitt beschrieben.

5.2 Ausblicke über weitere Schritte zur Umsetzung der Ergebnisse

Da es sich um einen Prototypen, welche auf Kundendaten basiert, handelt, müssten einige Schritte eingeleitet werden, um diesen zu einer vollständig einsatzfähigen Anwendung zu machen. Erster Schritt wäre die Android-Hybridanwendung über den Google PlayStore erreichbar zu machen. Anschließend müsste die Applikation an das GIS der Energie Graz GmbH & Co KG angebunden werden um den Zugriff zu deren aktuellen Daten und Datenmodell zu gewährleisten. Des Weiteren müssten die benötigten GIS-Objekte um das Attribut-Feld "RFID" erweitert werden. Abschließend müssten aus Datenschutzgründen die Lese- und Schreibmethoden auf den RFID-Transponder verschlüsselt werden.

5.3 Ausblicke über weiteren Forschungs- bzw. Handlungsbedarf für Probleme

Die Weiterentwicklungsmöglichkeiten dieser Anwendung sind umfangreich. Nachfolgend werden einige noch nicht gelöste beziehungsweise derzeit nicht lösbare Probleme angesprochen und ein paar Erweiterungsaussichten aufgeführt.

5.3.1 Offene Probleme

NFC-Plugin

Wie bereits in Unterabschnitt 3.2.2 erwähnt, bietet die JavaScript-Bibliothek Cordova noch keinen Zugriff auf die NFC-Komponente von Android-Smartphones. Aus diesem Grund wurde das frei verfügbare Plugin von Chariot Solutions in Cordova und somit auch in die Anwendung eingebunden. Dieses Plugin bietet jedoch noch keine 100%-fehlerfreie Funktionalität. Aus fortlaufenden NFC-Schreib-Tests mit der Prototyp-Anwendung ergab, dass dieser mit 5-10%-Wahrscheinlichkeit beim Schreiben auf den Transponder abstürzt. Es konnte jedoch nicht herausgefunden werden, ob dies an dem RFID-Transponder oder am Plugin liegt. Es wird jedoch vermutet, dass das frei-entwickelte Plugin noch keine 100%ige Stabilität aufweist. Die Hoffnung besteht, dass dieser Fehler durch ein neues Update, oder eine Weiterentwicklung des Plugins oder durch Integration eines offiziellen Cordova NFC-Zugriff behoben wird.

Öffnen von Verbunddokumenten

Wie in Unterabschnitt 4.3.1 angesprochen, können am GIS-Objekt angehängte Dokumente, sogenannte Verbunddokumente, angezeigt werden. Diese Anzeige ist derzeit nicht funktionstüchtig, da beim Aufruf des Dokumentes die Applikation abstürzt. Da es sich hierbei um eine Core-Funktionalität von GRINTEC GmbH handelt, war es nicht im Umfang und Umsetzungsbereich der Masterarbeit diesen Fehler zu beheben. Dieser Fehler tritt nur bei der hybriden Applikation auf. Wird SWebApp jedoch als Webapplikation über den Browser gestartet, dadurch fehlt natürlich der Zugriff auf NFC, ist das Öffnen von Verbunddokumenten fehlerfrei möglich.

Zoomlevel Hintergrundkarten

Abschnitt 4.4 weist bereits auf die Zoom-Problematik aufgrund der Hintergrundkarten von Open Street Map und Google Maps hin. Bei Einblendung dieser Hintergrundkarten ist ein Zoom nur bis zum größten Maßstab der Kartenlieferanten möglich und somit wird eine Detailansicht der GIS-Beleuchtungsobjekte verwehrt. Eine Lösungsmöglichkeit wäre, dass bei Detailansicht die Hintergrundkarten ausgeblendet

werden und diese somit nur bis zu einem bestimmten Maßstab sichtbar sind. Dieser Lösungsweg müsste jedoch noch auf Usability getestet werden, da dem Nutzer somit gewünschte Informationen automatisch ausgeblendet werden.

Listenauswahl SelectFeature

Wie bereits in Abschnitt 4.2 erwähnt und in Abbildung 4.3b ersichtlich, werden bei der Auswahl des zu koppelnden Betriebsmittels alle GIS-Objekte im Umkreise von 24 Pixel innerhalb des Auswahlpunkt in der Karte angezeigt. Dies ist prinzipiell sinnvoll, da sich auch zwei oder mehrere relevante Beleuchtungsbetriebsmittel in diesem Radius befinden können. Jedoch wäre es benutzerfreundlicher, dass nur Beleuchtungsobjekte in der darauffolgenden Auswahlliste erscheinen. Hierbei handelt es sich wiederum um eine Core-Funktionalität und konnte deshalb nicht im Umfang der Masterarbeit behoben werden.

Sprache der Mitteilungsboxen

Die Buttons der Mitteilungsboxen, wie zum Beispiel in Abbildung 4.9b erkennbar, sind derzeit noch auf Englisch und müssten anhand einer Übersetzungstabelle ins Deutsche übersetzt werden.

5.3.2 Erweiterungsaussichten

Offline-Fähigkeit

Ein Verbindung zum Internet muss vorhanden sein, um diese Applikation zu nutzten, die Daten in Echtzeit abzufragen und upzudaten. Für Gebiete in denen die Internetverbindung nicht gewährleistet ist, wäre eine Offline-Fähigkeit der Applikation von äußerst großem Vorteil.

Anbindung anderer Verwaltungssysteme und -datenbanken

Der Datenaustausch der Applikation ist derzeit nur mit dem GIS und dem RFID-Transponder möglich. Natürlich wäre eine Verbindung zu anderen Verwaltungssystemen und -datenbanken äußerst von Vorteil, wie zum Beispiel zum Omni-Tracker-System, siehe Prozess - Störungsmeldung und -behebung auf Seite 17. Es könnten mehr Informationen in der Anwendung angezeigt und verarbeitet werden und diese wäre somit umfangreicher.

CSS-Konfigurierbarkeit

Das Ändern von Styles, wie Farben und Größen von Elementen, im Zuge der Entwicklung ist derzeit noch sehr kompliziert und nicht leicht einstellbar. Es wäre ein Anliegen diese Darstellungsparameter einfacher konfigurierbar zumachen, um für jeden Kunden ein gewünschtes Design liefern zu können.

Literatur

Bill, Ralf (2010). *Grundlagen der Geo-Inforamtionssysteme*. Wichmann.

de.wikipedia.org (2013a). *Demilitarized Zone*. URL: **http://de.wikipedia.org/wiki/Demilitarized_Zone**.

— (2013b). *Grails*. URL: **http://de.wikipedia.org/wiki/Grails**.

— (2013c). *Near Field Communication (de)*. URL: **http://de.wikipedia.org/wiki/Near_Field_Communication**.

— (2013d). *RFID*. URL: **http://de.wikipedia.org/wiki/RFID**.

— (2013e). *Smallworld GIS*. URL: **http://de.wikipedia.org/wiki/Smallworld_GIS**.

en.wikipedia.org (2013a). *Groovy Server Pages*. URL: **http://en.wikipedia.org/wiki/Groovy_Server_Pages**.

— (2013b). *Near Field Communication (en)*. URL: **http://en.wikipedia.org/wiki/Near_field_communication**.

— (2013c). *Wizard (software)*. URL: **http://en.wikipedia.org/wiki/Wizard_(software)**.

Finkenzeller, Klaus (2012). *RFID Handbuch Grundlagen und praktische Anwendungen von Transpondern*. 6. Aufl. München: Hanser.

Forum, NFC (2013). *NFC-Forum*. URL: **http://www.nfc-forum.org**.

Gartner (2013). *Gartner Says by 2016, More Than 50 Percent of Mobile Apps Deployed Will be Hybrid*. URL: **http://www.gartner.com/newsroom/id/2324917**.

Günther, Sylvia (2009). *GIS-Lernkurs*. URL: **http://gis-lernkurs.elcms.net/content/e131/e178/**.

Newell, Richard G. (1989). *Smallworld Technical Papper No. 1. Ten Difficult Problem in Building GIS*.

Spiering Markus / Haiges, Sven (2011). *HTML5-Apps für iPhone und Android*. 2. Aufl. München: Franzis.

Abkürzungsverzeichnis

AJAX Asynchronous JavaScript and XML
API Application Programming Interface
App Application
CSS Cascading Style Sheets
DMZ Demilitarized Zone
EC Electronic Cash
EVU Energieversorgungsunternehmen
GE General Electric
GIS Geoinformationssystem
GPS Global Positioning System
GSM Global Systm for Mobile Communications
GSP Groovy Server Pages
GSS Smallworld GeoSpatial Server
JSP JavaServer Pages
HTML Hypertext Markup Language
HTTP Hypertext Transfer Protocol
ID Identifikationsnummer
JS JavaScript
JSON JavaScript Object Notation
LAN Local Area Network
NFC Near Field Communication
NDEF NFC Data Exchange Format
NIS Netzinformationssystem
ORC Optical Character Recognition
PDF Portable Document Format
QR Quick Response
RFID Radio Frequency Identification
RMI Remote Method Invocation
SIM Subscribe Identity Module
SW Smallworld
TCP/IP Transmission Control Protocol/Internet Protocol
UML Unified Modeling Language
WAN Wide Area Network
WAR Web Application Archive
WMS Web Map Service

Abbildungsverzeichnis

2.1 UML-Aktivitätsdiagram: Holzmastprüfung (Ist-Zustand) 5
2.2 UML-Aktivitätsdiagram: Standsicherheitsprüfung (Ist-Zustand) 6
2.3 UML-Aktivitätsdiagram: Anlagenprüfung (Ist-Zustand) 7
2.4 Systeminfrastruktur (Soll-Zustand) 12
2.5 UML-Sequenzdiagram: Kopplung (Soll-Zustand) 13
2.6 UML-Sequenzdiagram: Instandhaltung (Soll-Zustand) 15
2.7 Bsp. QR-Code 18
2.8 Bestandteile eines RFID-Systems 22
2.9 NFC - Active Mode 22
2.10 NFC - Passive Mode 23
2.11 Web-, Hybrid- und Native-App 28
2.12 Logo Sencha 29
2.13 Smallworld Fachschalen (Abbildung von GE) 32

3.1 SWebApps - Softwarestruktur 33
3.2 SWebApps - Clientstruktur 35
3.3 UML-Aktivitätsdiagramm - Kopplung Feature Action (Code) 40
3.4 UML-Aktivitätsdiagramm - Komponenten des Instandhaltungs-Wizards 43
3.5 UML-Aktivitätsdiagramm - egg.inst.ReadRfidPage (Klasse) 45
3.6 UML-Aktivitätsdiagramm - egg.inst.InstStoerungPage (Klasse) 46
3.7 UML-Aktivitätsdiagramm - egg.inst.LampenTauschPage (Klasse) 47

4.1 Grundfunktionalitäten SWebApp 50
4.2 Kartenlayer SWebApp 50
4.3 Kopplungsprozess (1) 51
4.4 Kopplungsprozess (2) 52
4.5 Instandhaltung-Grundelemente (1) 53
4.6 Instandhaltung-Grundelemente (2) 54
4.7 Wartungstätigkeit: Leuchtenreinigung - erfolgreich abgeschlossen 55
4.8 Störung aus Wartungstätigkeit: Lampentausch (1) 56
4.9 Störung aus Wartungstätigkeit: Lampentausch (2) 57
4.10 Explizite Störmeldung (1) 58
4.11 Explizite Störmeldung (2) 58
4.12 Zeigen von Störmeldungen (2) 59
4.13 Visualisierung - Hintergrundkarten 60
4.14 Symbolik - Beleuchtungsobjekte 61

Tabellenverzeichnis

2.1 Relevante GIS-Objekte und Anzahl 8
2.2 Atrribute von Objekt-Info (Ist-Zustand) 8
2.3 Attribute von Lokation (Ist-Zustand) 8
2.4 Attribute von E Mast (Ist-Zustand) 9
2.5 Attribute von E Leuchte (Ist-Zustand) 9
2.6 Attribute von E Beleuchtungsschaltstelle (Ist-Zustand) 9
2.7 Vergleich der verschiedenen Identifikationssystemen 20
2.8 Verschiedene RFID-Sendefrequenzbereiche 21

Listings

3.1 swebapp-config.groovy: Integration von nfcIO.js 36
3.2 nfcIO.js: Read-Methode 37
3.3 nfcIO.js: Write-Methode 38
3.4 swebapp-config.groovy: Konfiguration für Kopplung 39
3.5 swebapp-config.groovy: Konfiguration für Instandhaltung Leuchte - Hinzufügen von JS-Klassen 42
3.6 swebapp-config.groovy: Konfiguration für Instandhaltung Leuchte - Hinzufügen zum Menü-Panel 42
3.7 swebapp-config.groovy: Konfiguration für Instandhaltung Leuchte - Komponenten 43
3.8 gt_draw.magik: Zeichenmethode für Leuchtenreinigung 48

Appendix