HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

HUBER+SUHNER

Management-Software für Richtfunk-Systeme

Bachelorarbeit

Abteilung Informatik
Hochschule für Technik Rapperswil

Herbstsemester 09/10

Autoren:	Lukas Wilhelm, Damien Vouillamoz
Betreuer:	Prof. Dr. Peter Heinzmann
Projektpartner:	Huber+Suhner AG, Herisau
Experte:	Dr. Th. Siegenthaler
Gegenleser:	Prof. Hans Rudin

Bachelorarbeit

Aufgabenstellung Management-Software für Richtfunk-Systeme

Ausgangslage

Die Firma HUBER+SUHNER AG ist Hersteller eines Richtfunk-Systems (SENCITY®Link 60, kurz SL60) welches bei einer Funkfrequenz von 60GHz arbeitet. Bei diesem Richtfunk-System handelt es sich um eine drahtlose Ethernet-Bridge, welche Datenmengen von ≥100Mbit/s (netto, full-duplex) übertragen kann.

Die Geräte können heute einzeln über ein Webinterface konfiguriert werden. In Netzwerken, in denen mehrere solche Richtfunkstrecken installiert sind, entsteht deshalb die Problematik des effizienten und zentralen Managements der Geräte.

Ziel

Für die Anwender von SENCITY®Link soll eine Anwendung (stand alone Client, GUI-Software) entwickelt werden, mit der man zentral von einem Rechner aus alle SL60 Terminals in einem Netzwerk ansprechen und Massenabfragen/-änderungen vornehmen kann.

Nach dem einmaligen Einrichten der Software (Erfassen aller Geräte im Netzwerk, evtl. automatisch) sollen beispielsweise folgende Funktionen möglich sein:

- Detektion der (in lokalen Netzen) vorhandenen SENCITY Geräte
- Firmware Updates auf einmal für alle Geräte durchführen (gibt es da noch Sicherheitsanforderungen? Wie ist es mit dem Roll Back? Kommentar S.R.: Sicherheitsanforderungen sind noch zu diskutieren; Rollbackmöglichkeit zu einer vorgängigen Version denkbar)
- zentrales Speichern und Hochladen der Konfigurationen
- Konfigurationsänderungen welche für alle Geräte gleich sind auf einmal durchführen
- Überwachungsfunktionen wie:
    - Erreichbarkeit der Geräte
    - Menge der übertragenen Daten über den Link
    - Aufzeichnung von Linkunterbrüchen
- Empfangspegel aller Links mit einem Klick abfragen und darstellen
- Administrationsfunktionen zu den installierten Geräten?
    - Serienummer
    - Installationsort
    - Zuständige Person
    - Parameter

Anforderungen

Die Software soll auf Windows-, Linux- und Mac-Betriebssystemen lauffähig sein und sich nach Möglichkeit der auf dem Gerät bereits vorhandenen Schnittstellen (HTTP: read, write; SNMP: read) bedienen.

Datum: Betreuer:

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

Erklärung[1]

Die vorliegende Arbeit basiert auf Ideen, Arbeitsleistungen, Hilfestellungen und Beiträgen gemäss folgender Aufstellung:

Gegenstand, Leistung	Person	Funktion
Design, Datenhaltung, GUI	Lukas Wilhelm	Autor der Arbeit
Anforderungsspezifikation, Domainanalyse, Netzwerk	Lukas Wilhelm	Co-Autor
Anforderungsspezifikation, Domainanalyse, Netzwerk	Damien Vouillamoz	Autor der Arbeit
Design, Datenhaltung, GUI	Damien Vouillamoz	Co-Autor
Aufgabenstellung, Pflichtenheft, Betreuung während der Arbeit	Prof. Dr. P. Heinzmann	Verantwortlicher Professor
Idee, Entwicklung und zur Verfügungsstellung SL60 Geräte, Pflichtenheft	S. Racine, Huber+Suhner AG	Industriepartner
Feedback zur Anforderungsspezifikation und Domainanalyse	Prof. H. Rudin	Gegenleser

Ich erkläre hiermit,

- dass ich die vorliegende Arbeit gemäss obiger Zusammenstellung selber und ohne weitere fremde Hilfe durchgeführt habe,
- dass ich sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben habe.

Rapperswil, den..... .....
Student

Rapperswil, den..... .....
Student

[1] Diese Erklärung basiert auf der Muster-Erklärung in den *Richtlinien der HSR zur Durchführung von Projekt-, Studien-, Diplom- oder Bachelorarbeiten* vom 16. Februar 2009.

Inhaltsverzeichnis

Teil I.

Einleitung

1. Abstract

Abteilung	Informatik
Namen der Studierenden	Lukas Wilhelm, Damien Vouillamoz
Studienjahr	HS 09/10
Titel der Bachelorarbeit	Management-Software für Richtfunk-Systeme
Examinator	Prof. Dr. Peter Heinzmann
Experte	Dr. Th. Siegentaler
Gegenleser	Prof. Hans Rudin
Themengebiet	Internet-Technologien und -Anwendungen
Projektpartner	Huber+Suhner AG, Herisau
Institut	Institut für Internet-Technologien und -Anwendungen

Die Firma Huber+Suhner AG ist Hersteller eines Richtfunk-Systems (SENCITY®Link 60, kurz SL60), welches bei einer Funkfrequenz von 60GHz arbeitet. Bei diesem Richtfunk-System handelt es sich um eine drahtlose Ethernet-Bridge mit 100Mbit/s-Mikrowellenlink.

Die entwickelte Applikation SL60-MS (SL60 Management Software) ermöglicht die automatisierte Konfiguration von bis zu 1000 SL60-Geräten. Die Geräte werden in Gruppen und Links unterteilt. Die Konfiguration kann über eine detaillierte Eingabemaske eingegeben werden. Auf die Geräte angewandte Konfigurationen können rückgängig gemacht werden. Ausserdem werden Informationen zum Status der Geräte angezeigt. Dies erleichtert das Auffinden von fehlkonfigurierte Geräten.

Aus technischer Sicht ist die entwickelte Lösung eine plattformunabhängige Java Applikation. Die Datenhaltung wird über Hibernate und eine integrierte HSQL Datenbank gelöst. Für das Logging, die SNMP Kommunikation sowie die Darstellung von Graphen werden externe Bibliotheken benutzt (Snmp4J, Log4J, JFreeChart). Die Applikation wurde so aufgebaut, dass das System mit geringem Aufwand um die Konfigurationsmöglichkeit zukünftiger Firmware Versionen angepasst werden kann.

Die erarbeitete Lösung ist eine Prototyp-Version, welche für die Festlegung von detaillierten Anforderungen an ein Endprodukt, sowie für erste Tests genutzt werden kann. Die SL60-MS Software kann bereits im produktiven Einsatz für die Firmware Version 1.46 in homogener Umgebung eingesetzt werden. In inhomogenen Umgebungen mit verschiedenen Firmware Versionen kann sie als Grundlage für eine Weiterentwicklung genutzt werden. Die Abarbeitungs-Strategie zur erfolgreichen Konfiguration aller Geräte muss momentan noch manuell umgesetzt werden. Dies müsste

in einem zukünftigen Produkt überarbeitet und automatisiert werden.

2. Management Summary

Ausgangslage:
Die Firma Huber+Suhner AG ist Hersteller eines Richtfunk-Systems (SENCITY®Link 60, kurz SL60), welches bei einer Funkfrequenz von 60GHz arbeitet. Bei diesem Richtfunk-System handelt es sich um eine drahtlose Ethernet-Bridge mit 100Mbit/s-Mikrowellenlink. Die Geräte können heute nur einzeln über ein Webinterface konfiguriert und abgefragt werden. Um in Netzwerken mit mehreren SENCITY® Richtfunkstrecken die Geräte effizient konfigurieren und abfragen zu können, hat Stéphane Racine von Huber+Suhner AG die HSR angefragt, bei der Entwicklung einer zentralen Management Software behilflich zu sein.

Vorgehen:
Wegen den Schnittstellen des SL60-Geräts wurde entschieden die Software nicht auf einem bestehenden Management System aufzubauen, sondern als Java Standalone Applikation umzusetzen. Zur Entwicklung und für Tests standen im Labor zwei SL60-Geräte zu Verfügung.

Ergebnis:
Die entwickelte "Management Software für Richtfunksysteme" ist eine plattformunabhängige Java Applikation. Sie ermöglicht die automatisierte Konfiguration mehrerer SL60 Geräte mittels detaillierter Eingabemaske. Die SL60-Geräte werden übersichtlich in Gruppen und Links zusammengefasst. Ausserdem werden Informationen zum Status der Geräte angezeigt. Dies erleichtert das Auffinden von fehlkonfigurierten Geräten.

Ausblick:
Die realisierte Applikation bildet als "Prototyp" eine gute Grundlage für Weiterentwicklungen. Sie kann in homogenen Umgebungen, d.h. in Konfigurationen, bei welchen alle Geräte dieselbe Firmware 1.46 besitzen, bereits produktiv eingesetzt werden. Leider muss man mit der vorliegenden Lösung die SL60-Geräte in der "richtigen Reihenfolge" anwählen, damit sie alle konfiguriert werden können. Durch eine geschicktere Abarbeitungsstrategie könnte man diesen Nachteil wettmachen. In einer weiteren Phase könnte man die Unterstützung weiterer Firmware Versionen umsetzen.

Folgende Abbildungen helfen zum Verständnis des Systems.

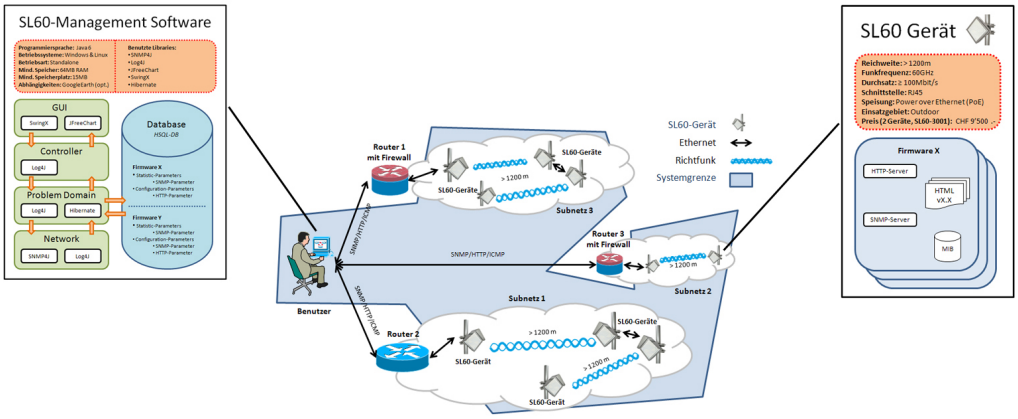


Abbildung 2.1.: Überblick des SL60 Systems

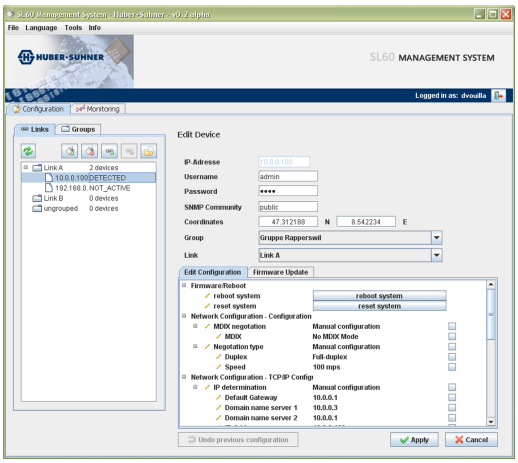


Abbildung 2.2.: Screenshot der SL60 Management-Software. Er zeigt die Konfiguration eines Gerätes.

3. Änderungsgeschichte des Dokuments

Datum	Version	Änderung	Autor
22.09.2009	0.10	Erstellung Zeitplan, Erstellung Anforderungsspezifikation	lw
29.09.2009	0.20	Weiterarbeit an Zeitplan und Anforderungsspezifikation, Erstellung Projektplan	lw, dv
05.10.2009	0.30	Erstellung Domainanalyse	dv
12.10.2009	0.40	Weiterarbeit Domainanalyse, Systemsequenzdiagramme	lw, dv
20.10.2009	0.50	Verbesserung Anforderungsspezifikation, Glossar	lw
22.10.2009	0.60	Domainanalyse - Systemoperationen	dv
28.10.2009	0.70	Erstellung Software Architecture Document	lw
09.11.2009	0.71	Erstellung Studienkapitel. Enthält neu Kapitel Vergleiche und Geräteerkennung.	lw
12.11.2009	0.72	Erstellung Design Kapitel, Network Package	dv
16.11.2009	0.73	Arbeit an Design Kapitel, Diagramme aller Packages	lw
20.11.2009	0.74	Systemtestdokumentationen erstellen und nachführen	lw
23.11.2009	0.75	Meilensteine, Releases nachführen	dv
26.11.2009	0.76	Überarbeitung Kapitel Design, Nachführung Anforderungsspezifikation	lw
01.12.2009	0.77	Systemtestdokumentation nachführen	dv
07.12.2009	0.80	Erste Version Management Summary, Poster	dv
07.12.2009	0.81	Erste Version Abstract	dv
17.12.2009	0.85	Überarbeitung Management Summary, Poster, Abstract abgeschlossen	dv, lw
15.12.2009	0.86	Meilensteine, Releases nachführen	dv
15.12.2009	0.87	Persönliche Bericht	dv, lw
16.11.2009	0.75	Anhang	lw
17.12.2009	1.00	vollständige Überarbeitung, Korrektur Layout	lw

Teil II.

Technischer Bericht

4. Studie

4.1. Vergleich Anforderungen mit Mitbewerber Tools

Produkt / Funktion	**SL60-MS**	**Motorola Canopy**	**Proxim Vision ES**	**Bridgewave (Webinterface)**	**Lightpointe (Flight Manager, direkt angeschlossenes Gerät)**
FR-01 Authentisierung:	Ja	Ja, ähnlich Anforderung	Ja	Ja, sehr ähnlich	Nein
FR-02 Benutzer verwalten inkl. Rollen zuweisen:	Ja	Ja, ähnlich Anforderung	Nein	Ja. (Benutzer "Anfänger"-User kann nichts ändern)	Nein
FR-03 scanbasierte Geräteerkennung:	Ja	Ja	Ja	n/r	Nein
FR-04 Intervall Parameter für Geräte Erkennung setzen:	Ja	n/a	n/a	n/r	Nein
FR-05 listenbasierte Geräteerkennung:	Ja	Ja	Ja	n/r	Ja
FR-06 Darstellung der Geräte:	auf Liste	auf Liste	n/a	n/r	n/r
FR-07 Detailansicht eines Geräts:	Ja	Ja	n/a	Ja	Ja
FR-08 Darstellung der Geräte auf einer Landkarte:	Ja	Ja	Ja	Nein	
FR-09 schematische Darstellung der Geräte und Verbindungen:	Ja	Nein	n/a	Nein	Nein

FR-10 Konfiguration durchführen:	Einzel, Massen	Einzel, Massen	Einzel, Massen	Einzel	Einzel
FR-11 vordefinierte Templates verwalten und anwenden:	Ja	Ja	Ja, ähnlich	Nein	Nein
FR-12 Konfigurations-History verwalten:	Ja	Ja (“Rückgängig-Option für 1 Schritt”, einzelne Parameter sind aber ausgeschlossen)	Nein	Nein	Nein
FR-13 Netzwerkstatistiken anzeigen:	Ja	Ja	Ja	Ja	Ja
FR-14 Geräte- und Verbindungsstatistiken exportieren und drucken:	Ja	Ja (nur exportieren)	Ja	Ja (da Internetseite)	Ja
FR-15 logische Fehlkonfiguration erkennen:	Ja	n/a	n/a	Nein	Nein
FR-16 Systemlog eines Geräts anzeigen:	Ja	Ja	Ja	Ja	Nein
FR-17 SNMP Trap Ansicht:	Ja	Ja	Ja	n/r	Nein
FR-18 Geräte gruppieren:	Ja	Ja	Ja	n/r	n/r
FR-19 Import/Export der Konfiguration:	Ja	Ja	Ja	Ja	Nein
XX-XX: Authentisierung der Geräte	Nein	Ja, (Shared Keys)	n/r	Nein	
XX-XX: Script Engine (Skript ausführen das bestimmte Konfiguration durchführt)	Nein	Ja	Nein	Nein	Nein
XX-XX: verwendete Protokolle	HTTP/SNMP v.2	HTTP/SNMP v.2	SNMP (v1,v2, v3WORP)	SNMP (v1,v2), SSL	Snmp und Serial,Com Link

XX-XX: Plattform	Windows (ab XP), Linux, (Mac)	Windows, Linux	Windows (ab 2003)	Webinterface	Windows (ab 95)
XX-XX: Minimum System Requirements		n/a	Dual Core 2Ghz, 2GB Ram, 125 MB Diskspace	Browser	n/a
XX-XX: Auto-Calibration	Nein	Nein	Nein	Ja	Nein
XX-XX: Ablaufzeit eines Passworts einstellen	Nein	Nein	Nein	Ja	Nein
XX-XX: exportierte Konfiguration als änderbares Text-File	Nein	Nein	Nein	Ja	Nein

Tabelle 4.1.: Vergleich Anforderungen mit Mitbewerber Tools. Informationen zu den Abkürzungen finden im Kapitel 4.1.2

4.1.1. Auswahl der Produkte

Huber+Suhner hat gewünscht, dass für die Festlegung der Anforderungen die in den Vergleich aufgenommenen Systeme studiert werden. Folgend sind Ergänzungen zu den einzelnen Produkten festgehalten:

Motorola Canopy

Die Software für das System Motorola Canopy bietet unzählige Funktionalitäten, die hier aber nicht alle aufgezählt werden können. In den Vergleich wurden darum nur die wichtigsten aufgenommen. Das System ist sehr professionell und durchdacht aufgebaut.

Proxim Vision ES

Die Software für das System Proxim Vision ist einfach und simpel aufgebaut. Es bietet die grundlegendsten Funktionalitäten an und ist übersichtlich gehalten.

Bridgewave (Webinterface)

Bridgewave bietet für ihr System ein Webinterface an, aber kein Netzwerkmanagement Tool. Dafür ist die interessante Funktionalität "Auto-Calibration" vorhanden. Diese bietet eine automatische Konfiguration an, so dass der Benutzer möglichst wenig selber einstellen muss.

Lightpointe (Flight Manager)

Das Netztwerkmanagement Tool von Lightpointe ist schon ein paar Jahre alt und bietet nur die grundlegendsten Funktionalitäten an.

4.1.2. Informationen zu den verwendeten Abkürzungen in Tabellen

Je nach Möglichkeiten des Systems kommt die Realisierung einer gewissen Anforderung gar nicht in Frage, dies wurde jeweils mit **n/r** (für "nicht relevant") gekennzeichnet. Wenn Informationen zu einer Anforderung nicht verfügbar waren, aber das Nicht-Vorhandensein dieser nicht zu 100% ausgeschlossen werden konnte, wurde dies mit **n/v** für ("nicht verfügbar") gekennzeichnet.

4.2. Vergleich mit bereits existierenden SNMP-Management Systemen

Produkt / Funktion	**MIB Explorer**	**MG-SOFT MIB Browser**	**HiliSoft SNMP MIB Browser**
Webseite	http://www.mibexplorer.com/	http://www.mg-soft.si/mgMibBrowserPE.html	http://www.hilisoft.com/
Kategorie	MIB Browser	MIB Browser	MIB Browser
Code	closed source	closed source	closed source
Kosten	$299	$425-$899	$49.95
Aktivitäten (Letzter Release)	1. November 2005	Seit 2004 Releases praktisch jeden Monat. Letzter Release: 27. Juli 2009	Seit Mai 2007 all zwei Monate einen Release. Letzter Release war im Mai 2008
Plattform	Alle Plattformen	Windows ME, Windows 2000, Windows XP, Windows Server 2003, Windows Vista, Windows Server 2008 / Linux (Intel x86 Architektur) / OS X v10.4.x Tiger (Intel x86 and PowerPC), OS X v10.5.x Leopard (Intel x86 and PowerPC)	Windows
Umgebung	JRE v1.4 (JRE v1.5 Empfohlen)	Bei Windows-Plattformen: MS IE 4.0 oder älter	n/v

Parameter lesen	Ja	Ja	Ja
Auf N Agents Parameter lesen	Nein	Nein	Nein
Parameter setzten	Ja	Ja	Ja
Auf N Agents Parameter setzen	Nein	Nein	Nein
Parameter iterativ lesen	Ja	Ja	Ja
Satz von Parameter auf einmal lesen	Ja	Ja	Ja
Graph für Netzwerkstatistiken	Ja	Ja	Ja
Logische Fehlkonfiguration erkennen	n/v	Nein	Nein
Informationen mittels HTTP lesen	Ja	Nein	Nein
Informationen mittels HTTP schreiben	Ja	Nein	Nein

Systemlog eines Geräts anzeigen	n/v	n/v	n/v
Ansicht SNMP-Traps	Ja	Ja	Ja
Geräte Gruppieren ?	Nein	Nein	Nein
Verwendete Protokolle	SNMPv1/v2c/v3	SNMPv1/v2c/v3	SNMPv1/v2c/v3
Erweiterbar durch Scriptsprache	Ja	Nein	Nein
Minimum System Requirements	128 MB RAM, 60 MB Harddisk, Support für lange Dateinamen	mind. 64 MB RAM (96 MB RAM sind empfhohlen)	n/v
Passwortschutz der Software	Nein	Nein	Nein
Import/Export der Konfiguration	Ja	Ja	Ja
exportierte Konfiguration als änderbares Text-File	Ja	Ja	Nein
Dokumentation	Ja (PDF, 135 S.)	Ja (PDF, 184 S.)	Nein, nur FAQs und wenig nützliche F1-Help

Tabelle 4.2.: Vergleich mit bereits existierenden SNMP-Management Systemen. Informationen zu den Abkürzungen im Kapitel 4.1.2

4.2.1. Auswahl der Produkte

Für die Evaluation wurde eine Auswahl von drei zu vergleichenden Produkten getroffen. Bei den für den Vergleich ausgewählten Produkten wurde vor allem auf Funktionalitäten geachtet, welche für die SL60-MS relevant sind. Folgend einige Ergänzungen zu den einzelnen Produkten:

MIB Explorer

Obwohl dieses Produkt closed source und kostenpflichtig ist, wurde es seit ca. 5 Jahren nicht mehr erweitert. Nebst der Unterstützung auf allen Plattformen, besteht die Möglichkeit SNMP-Traps einzusehen. Sicherlich am interessantesten ist die Möglichkeit dieses Produkt mit eigenen Scripts zu erweitern.

MG-SOFT MIB Browser

Der MG-SOFT MIB Browser ist mit Abstand das teuerste Produkt. Die rund $425-$899 Kosten erlauben zwar eine Unterstützung der Software auf allen Plattformen, doch überdeckt es nicht die vom Kunden gewünschten Funktionalitäten, und es ist in keiner Weise erweiterbar.

HiliSoft SNMP MIB Browser

Der HiliSoft SNMP MIB Browser bietet viele interessante und nützliche Funktionalitäten wie Netzwerktool (Ping, Traceroute), Leistungsgraph, Trap-Deamon und das Laden von verschiedenen MIB-Modulen. Leider läuft diese Software nur auf der Windows-Plattform. Eine Weiterentwicklung um die fehlenden, aber vom Kunden verlangten Funktionalitäten, ist aufgrund der closed source policy nicht möglich. Es besteht im Gegensatz zum "MIB Browser" keinerlei Möglichkeit dieses Produkt zu erweitern.

4.2.2. Entscheid

Nach reichlicher Evaluation erfolgte die Einsicht, dass keines der evaluierten Produkte für den gewünschten Gebrauch übernommen oder erweitert werden kann. Die Möglichkeit den "MIB Browser" mittels Scripts zu erweitern wurde nicht in Betracht gezogen. Einerseits behindert die wenig verbreitete Scriptsprache Velocity Template Language (VTL) die Erweiterbarkeit, und andererseits ist es sehr schwierig, wenn nicht gar unmöglich, all die gewünschten Funktionalitäten mittels dieser Skriptsprache umzusetzen.
Aus dem Grund dass kein bestehendes Produkt übernommen werden kann, wurde entschieden sich mit SNMP-API's auseinanderzusetzen. Damit müsste zumindest dieses Rad nicht neu erfunden werden.
Aus Zeit-, Lizenz- und Infrastruktur-Gründen war es nicht möglich alle Produkte im Praxiseinsatz zu testen. Wenn mehr Zeit zur Verfügung gestanden hätte, wären die verschiedenen Produkte und Infrastrukturen beschafft und deren Probleme und Lösungsansätze ausgiebig analysiert worden.

4.3. Vergleich bereits existierender SNMP API's

Produkt / Funktion	**Open JDMK**	**SNMP4J**	**Java SNMP API 4.1**	**WebNms SNMP Api 4**
Webseite	https://opendmk.dev.java.net	http://www.snmp4j.org	http://www.ireasoning.com	http://www.webnms.com/snmp/
Beschreibung	offene Implementation von SNMP (früher im JDMK von SUN intregriert)	SNMP4J ist eine Enterprise Open Source Implementation von SNMP für Java2SE 1.4.0. SNMP4J unterstützt das Generieren bzw. Absetzen von Befehlen (manager) sowie das Antworten bzw. Reagieren auf Befehle (agent).	SNMP API ist eine für alle Plattformen benutzbare SNMP Java API für die Erstellung von Netzwerk-Management Systemen. Alle SNMP Versionen werden voll unterstützt.	WebNms SNMP Api ist eine Java Bibliothek die neben den grundsätzlichen SNMP Funktionen auch weitere, auf Netzwerkmanagement Programme ausgelegte, Funktionen anbietet.
Code	open-source, GNU GPL Lizenz	open source, Apache Lizenz	closed source	closed source

Kosten	Kostenlos	Kostenlos	$595 pro Entwickler	$995 und $595 einmalig pro Entwickler + $995 jährlich pro Entwickler (MIB-Browser, MIB Module File loading, Database Support, Http Tunneling)
Lizenz für Weitervertreibung	muss wegen Copyleft Prinzip wieder unter GNU GPL Lizenz angeboten werden?	nicht nötig (Code unter Apache 2.0 license, Benutzung der Api muss ausgewiesen werden und Api Lizenz File einmal angezeigt werden.)	ist im Preis beinhaltet	Distribution Lizenz (einmalig $10'000)
Aktivitäten (Letzter Release)	Oktober 2007	Seit 2004 Releases praktisch jeden Monat. Letzter Release: 27. Juli 2009	28. Juli 2009	unbekannt
Entwickler	früher SUN, jetzt Community	2-er Team	Firma iReasoning	Firma ZOHO Corporation
Plattform / Umgebung	Java 1.4.2 oder später	Java 1.4.1 oder später	JDK 1.2 oder später	JDK 1.2 oder später
Parameter lesen	Ja	Ja	Ja	Ja

Parameter setzen	Ja	Ja	Ja	Ja
Auf N Agents Parameter lesen	Nein	Nein	Nein	Nein
Auf N Agents Parameter setzen	Nein	Nein	Nein	Nein
Parameter iterativ lesen (GET NEXT)	Ja	Ja	Ja	Ja
Satz von Parametern auf einmal lesen (GETBULK)	Ja	Ja	Ja	Ja
Satz von Parametern auf einmal setzen	Nein	Nein	Nein	Ja
automatisiertes Polling	Ja	Nein	Ja	Ja
Graph für Netzwerkstatistiken	Nein	Nein	Nein	Ja (einfacher Graph)
Logische Fehlkonfiguration erkennen	Nein	Nein	Nein	Nein

Informationen mittels HTTP lesen	Nein	Nein	Nein	Nein
Informationen mittels HTTP schreiben	Nein	Nein	Nein	Nein
Systemlog eines Geräts anzeigen	Nein	Nein	Nein	Nein
vereinfachtes Trap Listening	Ja	Ja	Ja	Ja
FR-18 Geräte gruppieren	Nein	Nein	Nein	Nein
Import/Export der Konfiguration	Nein	Nein	Nein	Nein
Mib Module File Parser	Nein (aber Mib Compiler Tool Mitlieferung)	Nein (aber Möglichkeit über kaufbare Erweiterung)	Ja	Nein (nur in Professional Edition)
Verwendete Protokolle	SNMP v1,v2c,v3	SNMP v1,v2c,v3	SNMP v1,v2c,v3	SNMP v1,v2c,v3

Minimum System Requirements Windows, Solaris, Linux	Minimum 32 MB Ram, jedes Os auf dem Java läuft	n/v	Windows, Solaris oder Linux. CPU: Minimum 500 MHz Pentium Processor. Memory: Minimum 128 MB RAM. Speicherplatz: Minimum 150 MB	
Passwortschutz der Software	Nein	Nein	Nein	Nein
exportierte Konfiguration als änderbares Text-File	Nein	Nein	Nein	Nein
Dokumentation	JavaDoc, Beispiel Code, Anleitungen	ausführliche JavaDoc, detailliertes Klassendiagramm, FAQs	JavaDoc, User Guide, FAQs, Beispiele	JavaDoc, Beispielcode, Guide, Faqs. Sehr gut dokumentiert.
Einfachheit des Aufbaus	etwas unübersichtlich	sehr übersichtlich	ok	ok

Tabelle 4.3.: Vergleich bereits existierender SNMP API's. Informationen zu den Abkürzungen im Kapitel 4.1.2

4.3.1. Auswahl der Produkte

Für die Evaluation wurde eine Auswahl von vier zu vergleichenden Produkten getroffen. Die Auswahl erfolgte aufgrund der Programmiersprache, Einfachheit, Lizenzbestimmungen und Plattformunabhängigkeit. Diese vier Produkte wurden unter dem Punkt 4.3 einander gegenüber gestellt.
Folgend einige Ergänzungen zu den einzelnen Bibliotheken:

Open JDMK

Das Open JDMK Projekt ging aus dem Java DMK Projekt hervor. Es bietet eine SNMP Implementation. Seit der Ausgliederung aus dem SUN JDMK sind keine weiteren Releases herausgekommen. Es besteht darum die Unsicherheit der Weiterführung dieses Projekts.

SNMP4J

Die SNMP4J API unterstützt alle drei SNMP-Versionen und wurde für den Enterprise Einsatz entwickelt. Die API ist mittels JavaDoc dokumentiert. Diese API ist Open-Source und unter der Apache Lizenz freigegeben. Es bestehen bereits MIB-Browser die auf SNMP4J aufbauen. Die Funktionsfähigkeit kann darum als gegeben betrachtet werden.

Java SNMP API 4.1

Die Java SNMP API 4.1 bietet ziemlich viel. Sie unterstützt alle SNMP-Versionen auf allen Plattformen und das Dokumentationsmaterial bzw. das Beispielmaterial ist vorhanden. Die API muss für jeden Entwickler gekauft werden. Die Kosten bewegen sich in vernünftigem Rahmen. Insbesondere weil die Weitervertreibungslizenz bereits beinhaltet ist.

WebNms SNMP API 4

Bietet von den betrachteten API's am meisten Funktionalitäten an. Es vereinfacht auch die Graphen-Erstellung aus Parameter-Werten. Der grosse Negativpunkt sind die hohen Kosten dieser API.

4.3.2. Entscheid

Aus dem Vergleich in Tabelle 4.3 kann man folgende Rangliste anhand der erfüllten Funktionalitäten erstellen:

4.3.3. Funktionserfüllung

1. WebNms SNMP Api 4
2. Open DMK und Java SNMP API 4.1
3. SNMP4J

4.3.4. Schlussfolgerung

WebNms kommt aus Kundensicht wegen dem hohen Preis nicht in Frage. Open DMK und Java SNMP API 4.1 erfüllen ähnlich viele Funktionalitäten. Die API SNMP4J liegt minim hinter den beiden anderen. Die einzige Funktionalität die es nicht anbietet, ist das automatisierte Polling von Werten auf den einzelnen Geräten. Wenn die Software auf dem Open DMKD aufbauen würde, taucht die Frage auf ob sie wegen dem Copyleft Prinzip auch wieder unter der GNU GPL Lizenz freigegeben werden muss. Aus unserer Sicht würden wir dies mit Nein beantworten, weil keine Änderungen am Code der API gemacht werden. Folglich bestehen in den Rechten zur Weiterverbreitung keine wesentlichen Unterschiede zwischen dem Open DMK und der JAVA SNMP API 4.1. Wenn man den Aufbau der einzelnen API's und die Dokumentation anschaut, überzeugt uns SNMP4J am meisten. Daneben bietet es auch schon eine integrierte Unterstützung für das bekannte Logging-System "LOG4J".

Nach einer Betrachtung all dieser Faktoren haben wir uns für SNMP4J entschieden. Dies, obwohl man verglichen mit den anderen API's, minime zusätzliche Funktionen selber entwickeln muss. Die für uns überzeugenden Faktoren sind der Erfolg dieser API (Dutzende von Referenzimplementationen), die Apache Lizenz, die ständige Weiterentwicklung, der Preis, sowie die gute Dokumentation.

4.4. Open Source Netzwerkmanagement Tools

Nach dem Entscheid für die SNMP4J Bibliothek wurden Open Source Netzwerkmanagement Systeme untersucht die explizit auf Java und einer SNMP Bibliothek aufbauen (vorzugsweise SNMP4J). Das Ziel dabei ist die Abklärung, ob neben der reinen SNMP Funktionalität auch weitere Funktionalitäten aus einer Bibliothek bezogen werden können.

4.4.1. Open NMS

Open NMS ist ein Open Source Netzwerkmanagement System, welches auf Java aufbaut. Die Benutzung erfolgt über einen Webbrowser. Die Grundkonfiguration des Systems erfolgt über XML Dateien.

4.4.1.1. bereitgestellte Funktionalitäten

Open NMS ist sehr detailliert konfigurierbar und bietet eine breite Funktionsfülle an. Als Einziges, der während dieser Arbeit evaluierten Produkte, bietet es die Möglichkeit Parameter über HTTP auszulesen. Standardmässig wird als SNMP Bibliothek die bekannte SNMP4J API verwendet. Hervorzuheben sind die detaillierten Statistikfunktionen. Darin beinhaltet ist auch die Erstellung von Graphen für beliebige Parameter.

4.4.1.2. Erweiterungsmöglichkeiten

Open NMS ist auf das Monitoring ausgelegt. Die wichtigste Funktionalität die hinzugefügt werden müsste, wäre die Konfiguration der Geräte. Die Oberfläche des Systems müsste den Bedürfnissen des Kunden angepasst werden, insbesondere wäre eine Vereinfachung der Benutzeroberfläche nötig.

4.4.1.3. Installation

Für die Installation auf einem Windows System muss eine Postgres Datenbank zur Verfügung gestellt, sowie der Open NMS Installer ausgeführt werden. Die Installation auf einem Linux System funktioniert ähnlich. Es ist aber zusätzlicher Konfigurationsaufwand in Form der Setzung von Berechtigungen und Pfadvariablen nötig.

4.4.1.4. Auslieferung

Die Plattform ist unter der GNU GPL Lizenz freigegeben. Das bedeutet in diesem Fall, dass Code (insbesondere Plugins) der mit dem Open NMS kommuniziert, wieder unter der GNU GPL Lizenz freigeben werden muss. Nach Auslegung der Open NMS Entwickler gehört dazu jeder Code, welcher Open NMS-Klassen durch das **import** Statement von Java einbindet.

4.4.1.5. Dokumentation

Das Open NMS System ist für die Entwicklerseite unvollständig durch JavaDoc (siehe [NMSb]) dokumentiert. Das bedeutet, dass vielerorts Beschreibungen zu Klassen fehlen. Weitere Dokumentation, inbesondere für die Plugin-Entwicklung, fehlt leider. Die Dokumentation für Benutzer ist ausführlich in einem Wiki (siehe [NMSa]) gehalten.

4.4.1.6. Screenshots

Folgend zwei Screenshots der Benutzeroberfläche des Open NMS Systems. Die erste Abbildung zeigt die Detailansicht eines einzelnen überwachten Geräts. Die zweite Abbildung zeigt graphische Statistiken über Alarme und Ausfälle des zu überwachenden Systems.

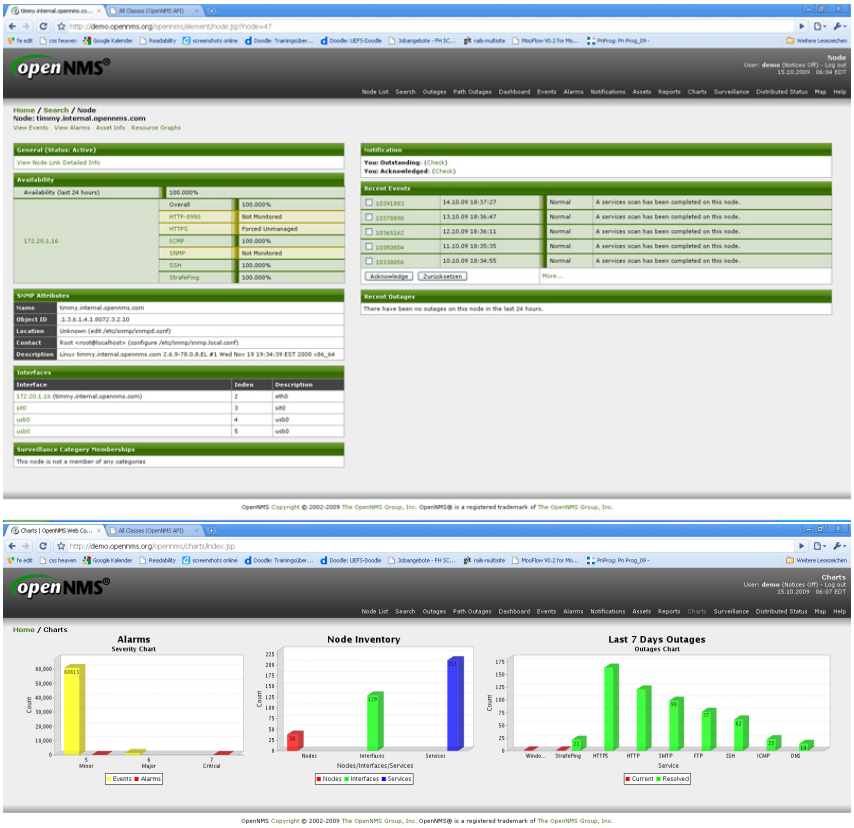


Abbildung 4.1.: Screenshots Open NMS

4.5. Geräteerkennung

Dieses Kapitel dient zur Festlegung wie SL60-Terminals in der SL60-MS Software (automatisch) erkennt werden. Das Verfahren wird in zwei Kategorien unterteilt.

4.5.1. listenbasierte Erkennung

Die listenbasierte Erkennung basiert auf dem Prinzip, dass IP-Adressen von Terminals als Auflistung in die Software importiert oder einzeln hinzugefügt werden können. Um sicher zu gehen, dass die angegebenen IP-Adresse auch aktiv sind, wird auf diesen ein ICMP-Ping abgesetzt. Nachdem ein Terminal geantwortet hat, soll mit einem SNMP GET und dem Community-String geprüft werden, ob es sich tatsächlich um ein SL60-Terminal handelt. Alternativ könnte man auch einen ARP-Request auf diesen Host absetzen. Dann würde man mit der erhaltenen MAC-Adresse prüfen ob das Terminal von Huber+Suhner hergestellt wurde.
listenbasierter Erkennungsablauf:

1. Importieren bzw. Hinzufügen von IP-Adressen von SL60-Terminals. Diese Geräte werden als "inaktiv" gekennzeichnet. Alternativ könnte hier mit einer Liste von MAC-Adressen gearbeitet werden, welche dann mittels Reverse-ARP die IP-Adressen der SL60-Terminals ermittelt.

   Das Reverse-ARP funktioniert umgekehrt zu ARP (Address Resolution Protocol). Mittels einer MAC-Adresse kann somit eine IP-Adresse ermittelt werden. Reverse-ARP sendet dazu eine Reverse-ARP Request-Broadcast mit der eigenen MAC-Adresse als Inhalt an die am Netzwerk angeschlossene Rechner. Ein benötigter Reverse-ARP-Server, welcher alle Zuordnungen von IP- und MAC-Adressen kennt, sendet darauf hin eine Antwort mit der IP-Adresse an die MAC-Adresse im Inhalt des Requests (Reverse-ARP-Reply).

   Ethernet-Broadcasts sind auf Subnetze beschränkt, so dass Rerverse-ARP nur in einem Subnetz eingesetzt werden kann. Wurde ein lokales Netzwerk in Subnetze aufgeteilt, so muss in jedem dieser Subnetze, in dem Reverse-ARP-fähige Workstations eingesetzt werden, ein eigener RARP-Server vorhanden sein. Aufgrund der Nicht-Funktionalen Anforderungen, dass die Software ohne grossen Abhängigkeiten laufen soll, ist dieses Verfahren nicht zu empfehlen.

2. Die hinzugefügten IP-Adressen werden auf ihre Erreichbarkeit geprüft.
   a) Ist eine IP-Adresse nicht erreichbar, wird diese markiert und für den weiteren Ablauf ausser Acht gelassen. Der Zustand dieses Gerätes wechselt zu "nicht detektierbar".
   b) Kam eine Antwort von diesem Gerät, ist bekannt, dass das Gerät erreichbar ist. Der Zustand wechselt in den Zustand "detektiert".

3. Mittels einem SNMP GET und dem Community-String wird geprüft, ob es sich bei den "detektierten" Terminals um Geräte der SL60-Familie handelt (mittels OID "iso.org.dod.internet.mgmt.mib-2.system.sysDescr"). Alternativ könnte mittels einem ARP-Request der Manufacturer-Code des Terminals geprüft werden.

a) Handelt es sich **nicht** um ein SL60-Terminal, wird der Status für dieses Terminal auf "nicht erkannt" gsetzt.

b) Handelt es sich um ein SL60-Terminal, wird der Zustand von "aktiv" in "erkannt" gewechselt. "Erkannte" Terminals werden dann in einer Liste oder auf einer Karte angezeigt. Sie stehen dem Benutzer fortan zur Konfiguration zur Verfügung.

4.5.2. scanbasierte Erkennung

4.5.2.1. Scan einer IP-Range

Unter Angabe von Start- und End-IP-Adressen werden die dazwischen liegenden Hosts zuerst angepingt und anschliessend auf den Gerätetyp geprüft.

Der Kunde möchte diese Methode aber aus Sicherheitsgründen nicht umsetzen.

4.6. User Modes

Benutzer welche die Admin-Rolle besitzen können neue Benutzer erstellen, sowie vorhandene Benutzer administrieren. Benutzer mit der User-Rolle können nur die eigenen Daten verändern.

5. Anforderungspezifikation

5.1. Einleitung

5.1.1. Zweck

Ziel dieses Dokumentes ist es, die Anforderungen an die Software SL60-MS detailliert zu beschreiben. Mit Hilfe von Use Cases sollen notwendige Schritte für die Bedienung der Software aufgelistet werden.

5.2. Allgemeine Beschreibung

Die folgende Abbildung liefert einen Überblick des zu verwaltenden SL60-MS Systems. Es zeigt den Einsatzbereich und das Umfeld der SL60-MS. In diesem System sollen Benutzer der SL60-MS (typischerweise ein Netzwerkmanager) die Möglichkeit haben, mehrere SL60-Terminals auf ein Mal zu konfigurieren und sich so einen grossen Arbeitsaufwand zu ersparen.

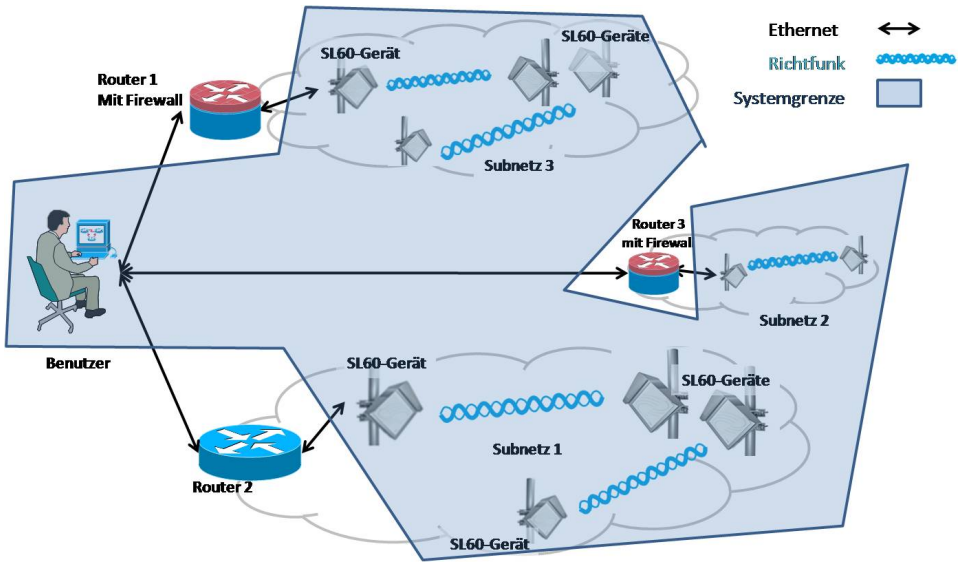


Abbildung 5.1.: Überblick des SL60-Systems

Der Benutzer hat das Ziel ein oder mehrere SL60-Terminals zu konfigurieren. Die Terminals befinden sich in unterschiedlichen Subnetzen. Sie lassen sich in logische Gruppen einteilen. Die Router und Firewall Einstellungen befinden sich nicht im

Einflussbereich der Software-Entwickler. Es wird aber vorausgesetzt dass SNMP und HTTP Verkehr zwischen der Software und den einzelnen Terminals erlaubt ist. Die SL60-MS läuft auf einem fest installierten Computer (im folgenden SL60-MS Host System genannt). Für kurzfristig nötige Konfigurationen wäre aber auch ein Einsatz auf einem mobilen Laptop möglich.

5.2.1. Benutzergruppen

Der Benutzer der Software ist typischerweise ein Netzwerkmanager oder Systemadministrator. Der Benutzer hat Netzwerkenntnisse und kann eine Software mit grafischem User Interface bedienen. Es kann nicht vorausgesetzt werden, dass sich der Benutzer mit jedem einzelnen Parameter des Geräts auskennt. Darum soll für unbedarftere Benutzer ein User-Modus bereit gestellt werden, in dem nicht alle Parameter geändert werden können. Ferner sollen ausführliche Hilfestellungen und Beschreibungen der verschiedenen Parameter verfügbar sein. Damit diese Benutzergruppe nicht überfordert wird, werden nicht alle Statistiken angeboten. Als Admin-Modus wird der Zustand der Software bezeichnet in dem alle Parameter und Statistiken zugänglich sind. Die tatsächlich realisierten Unterschiede zwischen den einzelnen Modi sind im Kapitel 4.6 beschrieben.

5.3. Funktionale Anforderungen

5.3.1. Überblick

In der folgenden Abbildung wird ein Überblick über die funktionalen Anforderungen an die SL60-MS gezeigt. Die Anforderungen sind in die Kategorien Fault-, Configuration-, Accounting-, Management-, und Security Management (FCAPS) eingeteilt worden.

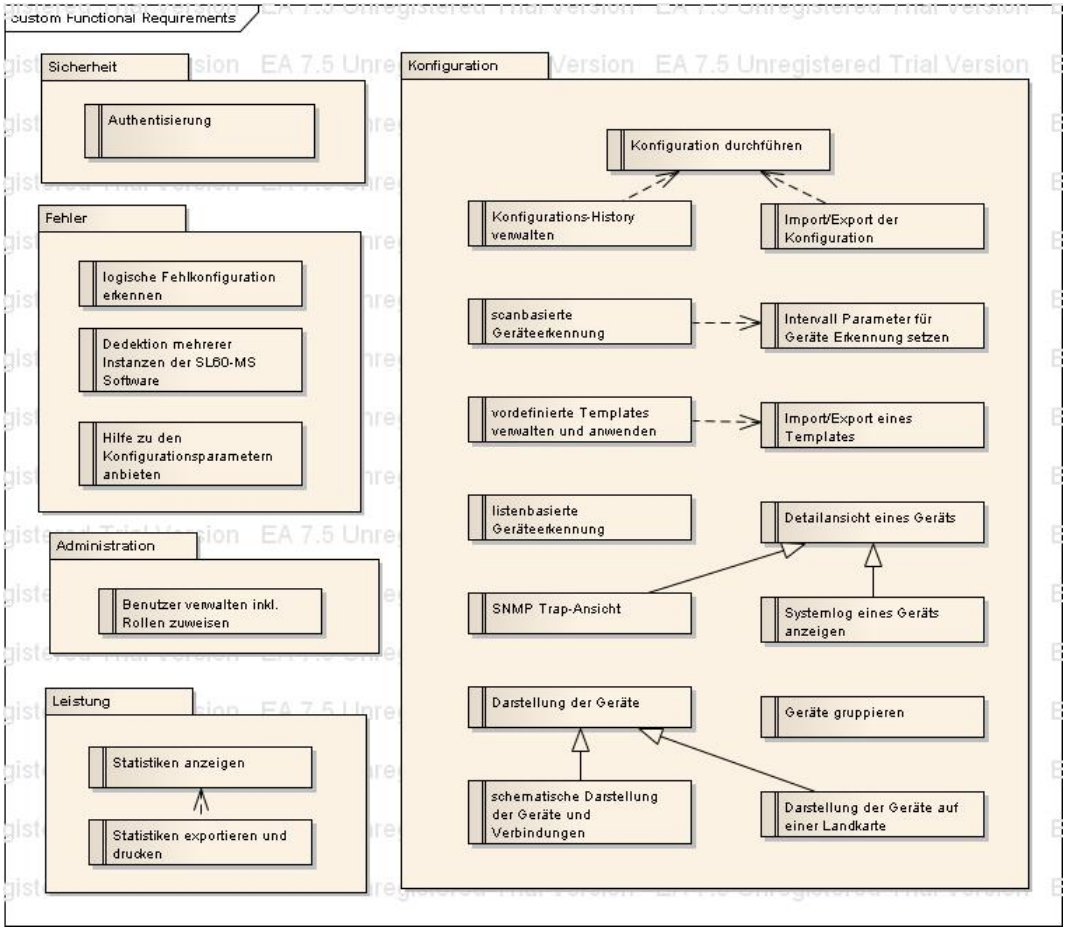


Abbildung 5.2.: Überblick der funktionalen Anforderungen

Die folgenden funktionalen Anforderungen werden auf Stufe eines Elementary Business Process definiert. Priorität A steht für “must have”, B für “can have” und C für “nice to have”.

5.3.2. Sicherheit

5.3.2.1. Authentisierung

ID	FR-01
Beschreibung	Nach dem Aufstarten der Software muss sich der Benutzer zuerst authentisieren. Die restlichen Funktionen der Software sind erst danach zugänglich. Bei Auslieferung ist ein Standard-Benutzer vorhanden.
Priorität	A

5.3.3. Administration

5.3.3.1. Benutzer verwalten inkl. Rollen zuweisen

ID	FR-02
Beschreibung	Ein Administrator kann neue Benutzer erstellen, bearbeiten und löschen. Dazu gehört auch die Zuweisung einer Admin- oder User-Rolle.
Priorität	B

5.3.4. Konfiguration

5.3.4.1. Konfiguration durchführen

ID	FR-10
Beschreibung	Für ein einzelnes Gerät oder mehrere SL60-Terminals (d.h. eine Terminalgruppe) kann eine Konfiguration inkl. Firmware-Update durchgeführt werden.
Priorität	A

5.3.4.2. Konfigurations-History verwalten

ID	FR-12
Beschreibung	Konfigurationen, die auf einem oder mehreren SL60-Terminals gemacht wurden, werden in der Software abgelegt und können rückgängig gemacht werden (“step-back”)
Priorität	A

5.3.4.3. Import/Export der Konfiguration

ID	FR-19
Beschreibung	Die Konfiguration eines SL60-Terminals oder einer Terminalgruppe kann gesichert und wiederhergestellt werden. Die exportierte Datei soll per Text-Editor veränderbar sein.
Priorität	B

5.3.4.4. listenbasierte Geräteerkennung

ID	FR-05
Beschreibung	Der Benutzer kann SL60-Terminals statisch hinzufügen oder entfernen. Das kann über eine Liste von IP-Adressen in einer XML-Datei oder der manuellen Eingabe von einzelnen IP-Adressen geschehen.
Priorität	A

5.3.4.5. scanbasierte Geräteerkennung

ID	FR-03
Beschreibung	Durch periodisches Scannen von (parametrisierbaren) IP-Ranges werden alle vorhandenen Netzwerkgeräte mit diesen Adressen angesprochen. Falls eine IP-Adresse bzw. ein netzwerkfähiges Gerät Antwort gibt, wird geprüft ob es sich tatsächlich um ein SL60-Terminal handelt. Da diese Methode vom Kunden aus Sicherheitsgründen nicht erwünscht ist, erhält diese Anforderung die tiefste Priorität.
Priorität	C

5.3.4.6. Intervall Parameter für Geräte Erkennung setzen

ID	FR-04
Beschreibung	Die Intervallzeit nach der erneut Geräte gesucht werden, kann vom Benutzer gesetzt werden. Der Default-Wert und der Wertebereich dieser Zeitangabe werden in einer späteren Phase definiert.
Abhängigkeiten	Eine Unterfunktionalität der Funktionalität “scanbasierte Geräteerkennung”.
Priorität	C

5.3.4.7. vordefinierte Templates verwalten und anwenden

ID	FR-11
Beschreibung	Es stehen bereits vordefinierte Templates (verschiedene Standard-Einstellungen) zur Verfügung, mit denen die SL60-Terminals konfiguriert werden können. Die Templates können verwaltet sowie angewendet werden. Für die Templates besteht die Einschränkung, dass Parameter die gerätespezifisch sind (bsp. Systemname) ausser Acht gelassen werden müssen.
Prioriät	B

5.3.4.8. Import/Export eines Templates

ID	FR-20
Beschreibung	Die im System erfassten Templates können gesichert und wiederhergestellt werden.
Abhängigkeiten	Eine Unterfunktionalität der Funktionalität “vordefinierte Templates verwalten und anwenden”.
Priorität	C

5.3.4.9. Geräte gruppieren

ID	FR-18
Beschreibung	Die erkannten SL60-Terminals können in Gruppen zusammengefasst werden. Für die Gruppe kann eine Bezeichnung und die dazugehörigen SL60-Terminals festgelegt werden.
Priorität	A

5.3.4.10. Darstellung der Geräte

ID	FR-06
Beschreibung	Es wird eine Liste mit allen erkannten SL60-Terminals angezeigt.
Priorität	A

5.3.4.11. Darstellung der Geräte auf einer Landkarte

ID	FR-08
Beschreibung	Wenn zu einem SL60-Terminal Koordinaten erfasst wurden, wird es auf einer Landkarte angezeigt. Verbindungen und Status werden ebenfalls angezeigt.
Abhängigkeiten	Eine Unterfunktionalität der Funktionalität “Darstellung der Geräte”.
Priorität	B

5.3.4.12. schematische Darstellung der Geräte und Verbindungen

ID	FR-09
Beschreibung	Der Benutzer kann sich die SL60-Terminals und Verbindungen schematisch darstellen lassen. Die Darstellung soll als Liste erfolgen und Überblick geben, zwischen welchen SL60-Terminals ein SL60-Link existiert. Der Status dieses Links wird auch angezeigt. Informationen wie Subnetze oder IP-Adressen werden ignoriert.
Abhängigkeiten	Eine Unterfunktionalität der Funktionalität "Darstellung der Geräte".
Priorität	C

5.3.4.13. Detailansicht eines Geräts

ID	FR-07
Beschreibung	Eine Detailansicht aller Informationen (Drill Down) eines SL60-Terminals ist möglich.
Priorität	A

5.3.4.14. Systemlog eines Geräts anzeigen

ID	FR-16
Beschreibung	Das Systemlog eines SL60-Terminals kann angezeigt werden.
Priorität	B

5.3.4.15. SNMP Trap Ansicht

ID	FR-17
Beschreibung	Falls im Netzwerk kein Netzwerkmanagement Tool im Einsatz ist, registriert sich die Software als SNMP Trap Server und zeigt dem Benutzer die aktuellen Meldungen zu den SL60-Terminals an.
Priorität	C

5.3.5. Leistung

5.3.5.1. Netzwerkstatistiken anzeigen

ID	FR-13
Beschreibung	Statistiken zum Netzwerkverkehr können angezeigt werden. Die Werte können in einem (MRTG)-Graph oder als Zahlenwerte angezeigt werden.
Priorität	B

5.3.5.2. Geräte- und Verbindungsstatistiken exportieren und drucken

ID	FR-14
Beschreibung	Geräte- und Verbindungsstatistiken können exportiert und gedruckt werden.
Priorität	C

5.3.6. Fehler

5.3.6.1. Detektion mehrerer Instanzen der SL60-MS

ID	FR-21
Beschreibung	Es soll dedektiert werden wenn mehr als eine SL60-MS gleichzeitig im selben Netzwerk am Laufen ist.
Priorität	B

5.3.6.2. logische Fehlkonfiguration erkennen

ID	FR-15
Beschreibung	Einfache Konfigurationsfehler werden von der Software erkannt. Es wird eine Meldung ausgegeben wenn bsp. eine IP doppelt vorhanden ist.
Priorität	C

5.3.6.3. Hilfe zu den Konfigurationsparametern anbieten

ID	FR-22
Beschreibung	Es sollen Erklärungen zu den einzelnen Konfigurationsparametern abgerufen werden können.
Priorität	B

5.4. Qualitative Anforderungen

Die qualitativen Anforderungen kapseln schwer messbare Anforderungen die zur Kundenzufriedenheit führen. Sie klären Fragen wie “Wie viel?”, “Wie schnell?” und “Wie gut?” die Software laufen soll. Die Kategorien sind dem Qualitätsmodell nach ISO Standard 9126 (siehe [fS01]) entnommen.

5.4.1. Leistungsanforderungen

Auch wenn die Performance relativ stark von der Hardwareumgebung, der Systembenutzung und der Netzauslastung abhängig ist, sollte das SL60-Terminal auf Aktionen vom Benutzer in weniger als einer Sekunde reagieren. Bei diesen Benutzerinteraktionen wird stets “Subsecond Response Time” angestrebt. Auch wenn die angeforderten

Informationen noch nicht zur Verfügung stehen, soll dem Benutzer signalisiert werden, dass seine Aktion registriert wurde und in Bearbeitung ist. Es soll angegeben werden wie lange die Bearbeitung typischerweise noch dauert. Die Dauer für die Beschaffung der angeforderten Informationen (Geräteerkennung, Statistik generieren und anzeigen), soll unter 5 Sekunden liegen. Dies lässt den Benutzer produktiv arbeiten.

5.4.2. Mengenanforderungen

Die Software sollte in der Lage sein maximal 1000 SL60-Terminals zu verwalten. Es können 100 verschiedene Benutzer eingerichtet werden. Es besteht die Einschränkung, dass immer nur ein Benutzer gleichzeitig aktiv sein kann. Ausserdem sollte ein SL60-Terminal nur von einer SL60-MS konfiguriert werden, da sie sich die verschiedenen Instanzen sonst gegenseitig Einstellungen überschreiben könnten.

5.4.3. Anforderungen an Schnittstellen

5.4.3.1. Hardware Schnittstellen

Um die Software ordnungsgemäss benützen zu können, bedarf es einer Schnittstelle für einen Netzzugang (inkl. Rechte). Ausserdem müssen Schnittstellen für das Entgegennehmen von Benutzereingaben und das Ausgeben von Informationen vorhanden sein. Für die Benutzereingaben empfiehlt sich eine Tastatur und für die Ausgabe ein Bildschirm. Weitere Bedingungen für die Hardware finden Sie im Kapitel 5.4.4.2.

5.4.3.2. Kommunikationsschnittstellen

Die Kommunikation zwischen SL60-MS und SL60-Terminals soll über die Protokolle HTTP und SNMP laufen. Wenn dieser Verkehr von Firewall und Routern zugelassen wird, dürfen keine Kommunikationsprobleme auftreten.

5.4.4. Randbedingungen für den Entwurf

5.4.4.1. Einschränkungen bezüglich Software

Aufgrund bereits vorgegebener Anforderungen wird für diese Software die Programmiersprache Java in der Version 6 verwendet.

5.4.4.2. Einschränkungen bezüglich Hardware

Mindestanforderung an das SL60-MS Host System sind die Mindestanforderungen an Java 6. Diese unterscheiden sich je nach Plattform. Für alle Plattformen sind aber minimal 64MB Arbeitsspeicher und 100MB Speicherplatz notwendig. Genauere Angaben sind auf SUN's Internetseite über Java zu finden (siehe [SUN]).

5.4.5. Qualitätsmerkmale

5.4.5.1. Funktionalität

5.4.5.1.1. Sicherheit Die Kommunikation zwischen den SL60-MS Geräten und der SL60-MS findet aufgrund der Nichtunterstützung von SNMPv3 und dessen Sicherheitsmechanismen, unverschlüsselt statt. Der Zugang zur SL60-MS wird nur durch Eingabe von korrekten Benutzernamen und Passwort erlaubt.

5.4.5.2. Zuverlässigkeit

Die Software soll trotz nebenläufigem Detektieren von vorhandenen Geräten weiterhin bedienbar sein. Umgekehrt soll während der Bedienung der Software das Netzwerk auf neue bzw. noch nicht detektierte Geräte untersucht werden. Zudem dürfen keine Statusmeldungen verloren gehen.

5.4.5.2.1. Reife Durch ausgiebiges Testen soll einerseits eine geringe Versagenshäufigkeit durch Fehlerzustände erreicht werden, und anderseits die Fehlgriffe eines Benutzers in Folge einer unklaren Benutzerschnittstelle vermieden werden.

5.4.5.2.2. Fehlertoleranz Trotz möglicher Software-Fehler oder Nicht-Einhaltung der spezifizierten Schnittstelle, soll die Software möglichst ohne Datenverlust weiterhin nutzbar bleiben.

5.4.5.2.3. Robustheit Logische Fehlkonfigurationen sollen vor dem Speichern der Daten erkannt werden und das Speichern dieser verhindert werden. Desweiteren soll die Software einen Schutz vor unüblichem bzw. unerwartetem Beenden bieten, um keinen Datenverlust zu riskieren.

5.4.5.2.4. Wiederherstellbarkeit Beim Eintritt eines unerwarteten Fehlers (durch Software-Fehler oder Nicht-Einhaltung der spezifizierten Schnittstellen), soll die Software möglichst ohne Datenverluste bzw. mit geringem Zeitaufwand wieder in einen gültigen Zustand hergestellt werden können.

5.4.5.3. Benutzbarkeit

5.4.5.3.1. Verständlichkeit Das User-Interface sowie die Funktionalitäten sollen für einen etwas unbedarfteren sowie für einen fortgeschrittenen Netzwerkmanager gut verständlich und selbsterklärend sein. Weiter muss Englisch verstanden werden um die Software zu benutzen. Die Interaktionsmöglichkeiten der Benutzerschnittstelle soll intuitiv gestaltet werden. Die Handhabung soll leicht erlernbar sein, so dass die Fehlerrate möglichst klein bleibt.

5.4.5.3.2. Erlernbarkeit Der Aufwand der für einen Benutzer anfällt um sich mit dem Softwaregebrauch vertraut zu machen, soll minimiert werden. Durch intuitive Gestaltung der Benutzerschnittstelle soll der Benutzer automatisch erkennen können, wo welche Aktionen ausgelöst werden können. Zudem soll bei der Gestaltung der Schnittstelle auch auf dem Benutzer bereits bekannte Konzepte und Modelle zurückgegriffen werden.

5.4.5.3.3. Bedienbarkeit Die Benutzerschnittstelle der Software sollte möglichst einfach gestaltet sein. Die Software sollte zudem dem Benutzer Missgriffe verzeihen können und diesen durch Bestätigungsfragen schützen. Um es weniger bedarften Benutzern möglichst einfach zu machen werden zwei verschiedene Detaillierungs-Stufen des Interfaces angeboten. Im User-Modus werden einige Optionen aus dem Admin-Modus ausgeblendet.

5.4.5.4. Effizienz

5.4.5.4.1. Zeitverhalten Die Verarbeitungszeit für Berechnungen und Informationsanforderungen soll so klein wie möglich gehalten werden. Bei längeren Verarbeitungszeiten oder Informationsanforderungen, sollte der Benutzer laufend über den Fortschritt informiert werden.

5.4.5.4.2. Verbrauchsverhalten Die Software soll die externen Ressourcen wie CPU-Zeit, Harddisk-Zugriffe, Netzauslastung auf eine Weise brauchen, in welcher andere Prozesse auf der Host-Maschine weiterlaufen können.

5.4.5.5. Änderbarkeit

5.4.5.5.1. Analysierbarkeit Mittels automatischen Unit Tests soll die Software einfach und schnell getestet werden können. Zusätzlich sollen im produktiven Einsatz der SL60-MS Nutzungsstatistiken erstellt werden. Damit soll überprüft werden wie oft welche Funktionen genutzt werden.

5.4.5.5.2. Modifizierbarkeit Die Software soll so entwickelt werden, dass Teile der Software einfach und ohne grossen Aufwand geändert und erweitert werden könnten. Ein sicherlich wichtiges Konzept zu Erreichung dieser Ziele ist das DRY-Konzept (Don't repeat Yourself). Zum Beispiel soll ohne grossen Aufwand eine neue Sprache hinzugefügt werden können. Ein anderes Beispiel wäre die Möglichkeit die Software später um die SNMP Version 3 zu erweitern.

5.4.5.5.3. Stabilität Nach Änderung der Software soll der Zustand mittels Unit Tests erneut getestet werden. So kann das Risiko, dass die Software nach den Änderungen unerwartete Seitenefekte auslöst, minimiert werden.

5.4.5.6. Übertragbarkeit

5.4.5.6.1. Anpassbarkeit Die Software lässt sich aufgrund der verwendeten Programmiersparache (Java) leicht auf jede Umgebung, auf welcher die JVM (Java Virtual Machine) in der gleichen Version installiert ist, portieren. Die Anwendung muss zwingend auf Windows (ab Windows XP) und Linux laufen. Unterstützung von Mac Os/X ist freiwillig. Zudem soll die Software in unterschiedlichsten Netzwerken ohne Fehler funktionieren.

5.4.5.6.2. Installierbarkeit Der Aufwand für die Installation ist gering. Durch Kopieren der Ordnerstruktur soll die Software in jedes mögliche Verzeichnis (auf dem Schreibrechte vorhanden sind) kopiert und dort ausgeführt werden können.

5.4.5.6.3. Koexistenz Pro Netzwerk darf jeweils nur eine Instanz der Software aktiv sein.

5.4.6. Andere Anforderungen

5.4.6.1. Konfigurierbarkeit

Die Texte innerhalb der Software sollen über Text Dateien konfigurierbar sein.

5.5. Use Cases

5.5.1. Überblick

In der folgenden Abbildung ist eine Übersicht der Use Cases gezeigt. Die Primärziele des Benutzers sind generell etwas mehr links dargestellt. Verschiedene Ziele wie bsp. "Authentisierung durchführen" sind rechts von den primären Use-Cases dargestellt, da Sie keine primären Interessen eines Benutzers darstellen. Die Use Cases wurden nur dann im Fully-Dressed Format ausgearbeitet wenn sie danach auch umgesetzt wurden.

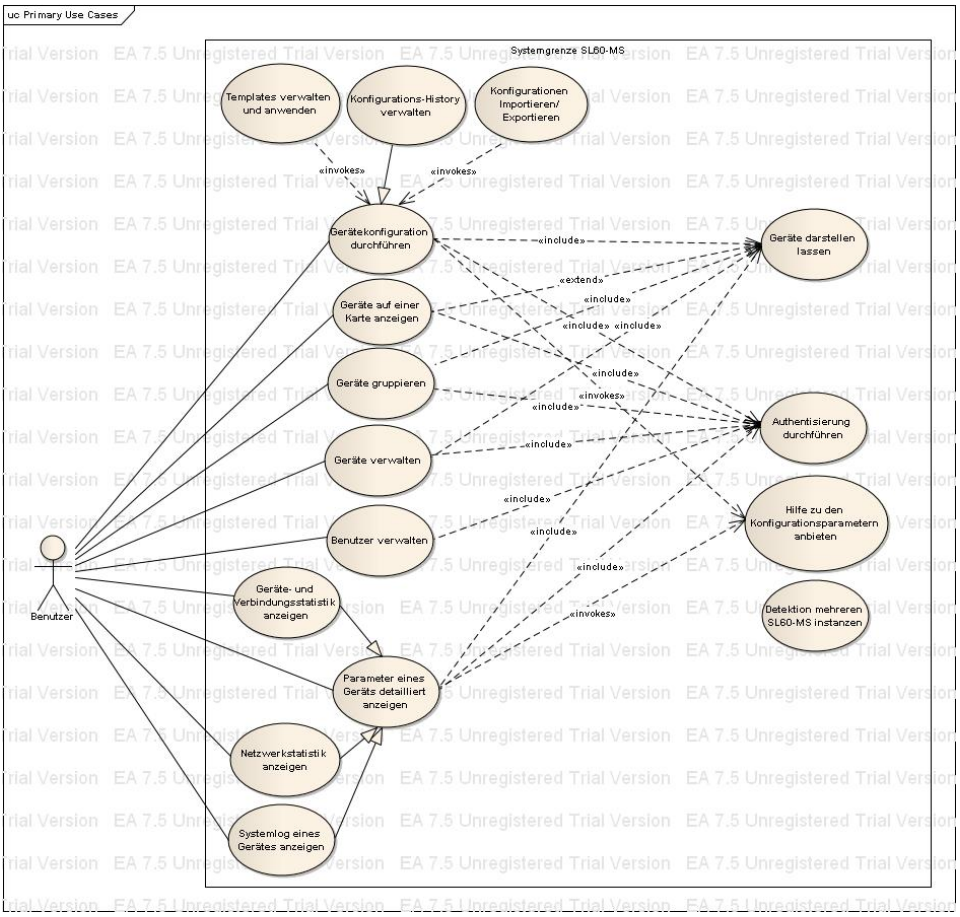


Abbildung 5.3.: Primäre Use Cases

5.5.2. Akteure

Benutzer Der Benutzer der SL60-MS ist typischerweise ein Netzwerk- oder Systemadministrator.

Netzwerkbetreiber Als Netzwerkbetreiber wird die Firma, welche auf das SL60-MS

System angewiesen ist, bezeichnet.

5.5.3. Use Cases

5.5.3.1. Geräte verwalten

ID	UC-01
Umfang	SL60-MS
Ebene	Userziel
Primärakteur	Benutzer
Stakeholder und Interessen	• Benutzer: Möchte seine SL60-Terminals in der Software erfassen und verwalten.
Vorbedingungen	Benutzer hat sich bei der Software angemeldet.
Nachbedingungen	Die vom Benutzer gewünschten SL60-Terminals wurden hinzugefügt, bearbeitet oder entfernt.
Standardablauf	1. Benutzer wählt aus, dass er SL60-Terminals verwalten möchte. 2. System bietet Oberfläche an, auf welcher die bereits hinzugefügten SL60-Terminals sichtbar sind. 3. Benutzer kann aus einer Liste ein SL60-Terminal auswählen. 4. System bietet eine Oberfläche an, auf welcher der Benutzer die Verbindungsinformationen zum ausgewählten SL60-Terminal ändern kann. Anschliessend kann dieser die neuen Angaben speichern.

Erweiterungen (oder alternative Abläufe)	**3-4 a** 1. Benutzer gibt Befehl um ein neues SL60-Terminal hinzuzufügen 2. System bietet Oberfläche an, auf welchem der Benutzer Angaben zum SL60-Terminal machen kann. Anschliessend kann dieser die Angaben speichern. **3-4 b** 1. Benutzer erstellt in einem XML-File eine Liste von IP-Adressen. 2. Benutzer importiert diese Liste. **3-4 c** 1. Benutzer kann ein SL60-Terminal aus einer Liste auswählen 2. Nach dem Wählen der Löschen-Aktion löscht das System das SL60-Terminal aus der Liste. **dauernd** 1. Benutzer gibt Befehl zum Abbrechen des aktuellen Vorgangs. 2. System bricht aktuellen Vorgang ab.
Spezielle Anforderungen	Nach jeder Änderung müssen diese Informationen persistent gespeichert werden.
Liste der Technik- und Datenvariationen	keine
Häufigkeit des Auftretens	Selten
Offene Fragen	

5.5.3.2. Gerätekonfiguration durchführen

ID	UC-02
Umfang	SL60-MS
Ebene	Userziel
Primärakteur	Benutzer
Stakeholder und Interessen	• Benutzer: Möchte SL60-Terminals nach seinen Wünschen konfigurieren. • Netzwerkbetreiber: Möchte dass seine Mitarbeiter in möglichst kurzer Zeit möglichst viele SL60-Terminals konfigurieren
Vorbedingungen	Benutzer hat sich bei der Software angemeldet. Ausserdem muss mindestens ein SL60-Terminal im System erfasst worden sein.
Nachbedingungen	Die vom Benutzer gewünschte Konfiguration wurde übernommen.
Standardablauf	1. Benutzer wählt ein SL60-Terminal aus, welches er konfigurieren möchte. 2. System bietet Oberfläche an, um das Gerät zu konfigurieren. Die gezeigten Einstellungen stellen die aktuell auf dem Gerät gespeicherte Konfiguration dar. 3. Benutzer trägt seine Änderungen in der Oberfläche ein. 4. Benutzer gibt den Befehl seine Änderungen auf das Gerät anzuwenden. 5. System startet die Übermittlung der Änderungen an das SL60-Terminal und zeigt dem Benutzer den Fortschritt an. 6. System protokolliert die getätigten Änderungen und gibt dem Benutzer zum Abschluss eine Erfolgsmeldung zurück.

Erweiterungen (oder alternative Abläufe)	
	1a 1. Benutzer wählt ein Link aus den er konfigurieren möchte 2. System bietet Oberfläche an, um den bzw. die Links zu konfigurieren. Die zur Verfügung stehenden Bearbeitungs-Felder sind nur für die Parameter vorhanden, welche für mehrere Geräte (2n) gleichzeitig gesetzt werden können. Die Werte dieser Felder stellen die letzte angewendete Konfiguration auf diesen Link dar. Falls keine vorherige Konfiguration vorhanden ist, sind die Werte dieser Felder noch nicht gesetzt.
	1b 1. Benutzer wählt eine Gruppe von SL60-Terminals aus die er konfigurieren möchte 2. System bietet Oberfläche an, um die Gruppe zu konfigurieren. Die zur Verfügung stehenden Bearbeitungs-Felder sind nur für die Parameter vorhanden, welche für mehrere Geräte (2n) gleichzeitig gesetzt werden können. Die Werte dieser Felder stellen die letzte angewendete Konfiguration auf diese Gruppe dar. Falls keine vorherige Konfiguration vorhanden ist, sind die Werte dieser Felder noch nicht gesetzt.
	5a 1. Falls die Übermittlung der Änderungen nicht erfolgreich abgeschlossen werden konnte, wird eine Fehlermeldung angezeigt. 2. Die Übermittlung der Änderungen kann erneut gestartet werden.
Spezielle Anforderungen	• Antwortzeit wird grösser mit der Anzahl ausgewählter SL60-Terminals
Liste der Technik- und Datenvariationen	
Häufigkeit des Auftretens	Laufend
Offene Fragen	

5.5.3.3. Geräte gruppieren

ID	UC-03
Umfang	SL60-MS
Ebene	Userziel
Primärakteur	Benutzer
Stakeholder und Interessen	• Benutzer: Möchte SL60-Terminals in Gruppen oder Links sortieren, damit er später anhand dieser Gruppen/Links Konfigurationen erstellen kann.
Vorbedingungen	Benutzer hat sich bei der Software angemeldet und es wurden bereits SL60-Terminals in die Software eingetragen.
Nachbedingungen	Die gewünschte Zuordnung von SL60-Terminals in Gruppen bzw. Links besteht.
Standardablauf	1. Benutzer erstellt eine neue Gruppe bzw. einen neuen Link. 2. Benutzer wählt die SL60-Terminals aus, die zu dieser Gruppe/Link gehören sollen. Ein SL60-Terminal darf aber nur einer Gruppe/Link angehören.
Erweiterungen (oder alternative Abläufe)	**1a** 1. Alternativ kann eine bereits bestehende Gruppe/Link ausgewählt werden um ihr SL60-Terminal hinzuzufügen oder zu entfernen.
Spezielle Anforderungen	
Liste der Technik- und Datenvariationen	
Häufigkeit des Auftretens	Selten
Offene Fragen	

5.5.3.4. Geräte darstellen lassen

ID	UC-04
Umfang	SL60-MS
Ebene	Userziel
Primärakteur	Benutzer
Stakeholder und Interessen	• Benutzer: Möchte SL60-Terminals visualisiert haben und diese einzeln sowie gruppiert auswählen können.
Vorbedingungen	Benutzer hat sich bei der Software angemeldet und es wurden bereits SL60-Terminals in die Software erfasst.
Nachbedingungen	
Standardablauf	1. Benutzer wählt aus, dass er sich die erfassten SL60-Terminals auf einer Liste anzeigen möchte
Erweiterungen (oder alternative Abläufe)	**1a** 1. Der Benutzer kann alternativ eine Kartenansicht auswählen. Falls in einem SL60-Link Paar Koordinaten-Informationen zu Verfügung stehen, wird dieser Link auf der Karte angezeigt.
Spezielle Anforderungen	• Falls SL60-Terminals nicht mehr erreichbar werden, muss dies visuell dem Benutzer kommuniziert werden.
Liste der Technik- und Datenvariationen	• Um einen SL60-Link auf der Karte anzeigen zu lassen, benötigt man beide der zwei Koordinaten-Informationen (N- bzw. S- und W- bzw. O-Abweichung)
Häufigkeit des Auftretens	Laufend
Offene Fragen	

5.5.3.5. Parameter eines Geräts detailliert anzeigen

ID	UC-05
Umfang	SL60-MS
Ebene	Userziel
Primärakteur	Benutzer
Stakeholder und Interessen	• Benutzer: Möchte detaillierte Informationen zu einem einzelnen SL60-Terminal anzeigen.
Vorbedingungen	Benutzer hat sich bei der Software angemeldet und es wurden bereits SL60-Terminals in der Software erfasst. Der Benutzer befindet sich in einer Ansicht in der er ein Terminal auswählen kann. Beim ausgewählten Gerät muss es sich um ein SL60-Terminal handeln und es muss erreichbar sein.
Nachbedingungen	
Standardablauf	1. Benutzer wählt das SL60-Terminal aus 2. Benutzer gibt Befehl, dass er sich Informationen zum Gerät anzeigen lassen will. (passiert automatisch bei der Auswahl) 3. System bietet eine Drill-Down Ansicht der Parameter an.
Erweiterungen (oder alternative Abläufe)	**2a** 1. Ist das Gerät nicht erreichbar, werden nur die Verbindungsinformationen angezeigt, welche er dann ändern kann.
Spezielle Anforderungen	
Liste der Technik- und Datenvariationen	
Häufigkeit des Auftretens	Laufend
Offene Fragen	

5.5.3.6. Konfiguration rückgängig machen

ID	UC-06
Umfang	SL60-MS
Ebene	Userziel
Primärakteur	Benutzer
Stakeholder und Interessen	• Benutzer: Möchte bereits auf SL60-Terminals angewendete Konfiguration rückgängig machen.
Vorbedingungen	Einer der Benutzer hat bereits einmal eine Konfiguration auf eines oder mehrere SL60-Terminals angewendet.
Nachbedingungen	Die letzte Konfiguration wurde zurückgespielt.
Standardablauf	1. Benutzer wählt das SL60-Terminal aus, für welches er die alte Konfiguration zurückspielen will. 2. System zeigt an, zu welchem Zeitpunkt eine oder mehrere Konfigurationen für ein SL60-Terminal durchgeführt wurden. 3. Benutzer wählt, dass er eine Konfiguration rückgängig machen möchte 4. System stellt die Konfiguration auf dem betroffenen SL60-Terminal wieder her. 5. System bestätigt dem Benutzer die Änderung der Konfiguration

Erweiterungen (oder alternative Abläufe)	**1-3 a** 1. Benutzer befindet sich im Use Case "Gerätekonfiguration durchführen" und möchte eine soeben angewandte Konfiguration rückgängig machen. Er hat die Möglichkeit diese Konfiguration mit einem einzigen Befehl rückgängig zu machen. **3-5 a** 1. Benutzer entscheidet sich die Konfiguration so zu belassen wie sie momentan ist. **dauernd** Es können maximal 10 Konfigurationen pro SL60-Terminal rückgängig gemacht werden.
Spezielle Anforderungen	
Liste der Technik- und Datenvariationen	
Häufigkeit des Auftretens	selten
Offene Fragen	

5.5.3.7. Authentisierung durchführen

ID	UC-07
Umfang	SL60-MS
Ebene	Subfunktion
Primärakteur	Benutzer
Stakeholder und Interessen	• Netzwerkbetreiber: Möchte Zugriff auf die Funktionen der Software nur für autorisierte Personen anbieten • Benutzer: Möchte Zugriff auf die Software erlangen.
Vorbedingungen	Person hat Kenntnis der Zugangsdaten eines Benutzers.
Nachbedingungen	Person hat sich am System authentisiert und ist nun authorisiert dies zu benutzen
Standardablauf	1. Benutzer startet die Software und wird gebeten seine Zugangsdaten anzugeben und diese anschliessend zu bestätigen 2. Wenn die Zugangsdaten korrekt sind, kann der Benutzer die Funktionalitäten der Software benutzen
Erweiterungen (oder alternative Abläufe)	**2a** 1. Wenn die Zugangsdaten falsch sind, erscheint eine entsprechende Meldung und die Eingabe kann ein weiteres Mal probiert werden.
Spezielle Anforderungen	Zu Beginn wird die Software mit einem Standard-Benutzer ausgeliefert.
Liste der Technik- und Datenvariationen	
Häufigkeit des Auftretens	sehr häufig (bei jedem Start der Software)
Offene Fragen	

5.5.3.8. Benutzer verwalten und Rollen zuweisen

ID	UC-08
Umfang	SL60-MS
Ebene	Userziel
Primärakteur	Benutzer
Stakeholder und Interessen	• Benutzer: Möchte die Benutzer verwalten.
Vorbedingungen	Benutzer hat sich bei der Software angemeldet und hat eine Admin-Rolle zugewiesen.
Nachbedingungen	Die gewünschten Änderungen wurden gespeichert.
Standardablauf	1. Benutzer ruft die Benutzerverwaltung auf. 2. System bietet Oberfläche an, auf welcher die bereits bestehenden Benutzer aufgelistet sind. 3. Benutzer betätigt den Befehl einen neuen Benutzer zu erstellen. 4. System bietet eine Oberfläche an, auf welcher der Benutzer Benutzername, Passwort, voller Name des Benutzers, Rolle und Status (aktiv/inaktiv) eigegeben werden kann. 5. Benutzer gibt Benutzerdaten ein und bestätigt das Speichern. 6. Das System speichert die Änderungen und aktualisiert die Liste der Benutzer

Erweiterungen (oder alternative Abläufe)	**2-5a** 1. Benutzer ist nicht der Admin-Rolle zugewiesen. 2. Benutzer kann nur seine eigenen Daten ändern. **3a** 1. Benutzer will einen Benutzer editieren. 2. Benutzer wählt einen bereits bestehenden Benutzer aus. **3-6** 1. Benutzer will einen Benutzer löschen. 2. Benutzer löst die Löschaktion aus. **6a** 1. Die eingegebenen Daten sind nicht valid. 2. System weist den Benutzer auf die Fehleingaben hin, und bietet die Möglichkeit diese nochmals einzugeben.
Spezielle Anforderungen	1. Nur Benutzer mit der Admin-Rolle können neue Benutzer erstellen. Benutzer mit der User-Rolle können jeweils nur ihre eigenen Daten ändern. 2. Es muss immer ein Benutzer mit der Admin-Rolle aktiv sein.
Liste der Technik- und Datenvariationen	keine
Häufigkeit des Auftretens	Selten
Offene Fragen	

5.5.3.9. Darstellung der Geräte auf einer Landkarte

ID	UC-09
Umfang	SL60-MS
Ebene	Userziel
Primärakteur	Benutzer
Stakeholder und Interessen	• Benutzer: Möchte SL60-Terminals eines Links auf einer Landkarte anzeigen lassen.
Vorbedingungen	Benutzer hat sich bei der Software angemeldet und mindestens ein Gerät mit Koordinaten-Informationen einem Link hinzugefügt.
Nachbedingungen	Die dem ausgewählten Link zugewiesenen SL60-Terminals sind, sofern diese Koordinaten-Informationen beinhalten, auf ein einer Karte sichtbar
Standardablauf	1. Benutzer lässt sich eine Liste aller Links anzeigen und wählt einen aus. 2. System prüft ob eine Darstellung auf einer Karte möglich ist. 3. System gibt dem Benutzer die Möglichkeit zum Auswählen der Kartenansicht 4. Benutzer wählt die Kartenansicht aus 5. System liest Statistikinformationen von den erreichbaren Geräten und generiert ein KML-File. 6. System startet das Karten-Programm (GoogleEarth) und übergibt diesem das KML-File
Erweiterungen (oder alternative Abläufe)	**3a** 1. System kann keine Anzeige generieren. 2. System stellt dem Benutzer keine Möglichkeit der Kartenansicht zur Verfügung

Spezielle Anforderungen	1. Das System muss den Pfad zur Karten-Applikation speichern.
Liste der Technik- und Datenvariationen	GoogleEarth wird als Karten-Programm benutzt.
Häufigkeit des Auftretens	Selten
Offene Fragen	

5.5.3.10. Systemlog eines Gerätes anzeigen

ID	UC-10
Umfang	SL60-MS
Ebene	Userziel
Primärakteur	Benutzer
Stakeholder und Interessen	• Benutzer: Bekommt Systemlog angezeigt
Vorbedingungen	Benutzer hat sich bei der Software angemeldet.
Nachbedingungen	Dem Benutzer wird das Systemlog in Form eines Graphen über die Zeit angezeit
Standardablauf	1. Benutzer wählt im Monitoring-Tab ein erreichbares SL60-Terminal aus. 2. Die Software verbindet zu diesem Gerät und liest das gerätespezifische Systemlog aus. 3. Es wird ein Diagramm über die Zeit generiert und anschliessen dem Benutzer angezeigt
Erweiterungen (oder alternative Abläufe)	
Spezielle Anforderungen	
Liste der Technik- und Datenvariationen	
Häufigkeit des Auftretens	gelegentlich
Offene Fragen	

5.5.3.11. Netzwerkstatistiken anzeigen

ID	UC-11
Umfang	SL60-MS
Ebene	Userziel
Primärakteur	Benutzer
Stakeholder und Interessen	• Benutzer: Möchte Netzwerkstatistiken anzeigen.
Vorbedingungen	Benutzer hat sich bei der Software angemeldet und mindestens ein erreichbares Gerät hinzugefügt.
Nachbedingungen	Dem Benutzer wird eine Übersicht über den Netzwerkverkehr angezeigt
Standardablauf	1. Benutzer wählt im Monitoring-Tab ein erreichbares SL60-Terminal aus. 2. Das System verbindet automatisch zu diesem Gerät und liest die gewünschten Statistikdaten aus. 3. Das System stellt dem Benutzer die Werte dar.
Erweiterungen (oder alternative Abläufe)	**1a** 1. Gerät ist nicht erreichbar. 2. System zeigt keine Statistik des Netzwerkverkehrs auf
Spezielle Anforderungen	
Liste der Technik- und Datenvariationen	
Häufigkeit des Auftretens	Selten
Offene Fragen	

5.5.3.12. Hilfe zu den Konfigurationsparametern anbieten

ID	UC-12
Umfang	SL60-MS
Ebene	Softwareziel
Primärakteur	Benutzer
Stakeholder und Interessen	• Benutzer: Möchte Hilfe zu den Konfigurationsparametern erhalten
Vorbedingungen	Benutzer hat sich bei der Software angemeldet und der Software den Befehl zum Editieren eines erreichbaren SL60-Terminals gegeben.
Nachbedingungen	Benutzer hat Hilfe zu den Konfigurationsparametern erhalten
Standardablauf	1. Benutzer fährt mit dem Maus-Cursor auf den Namen eines Konfigurationsparameters. 2. System zeigt in einem Tooltip eine Beschreibung zu diesem Konfigurationsparameter an.
Erweiterungen (oder alternative Abläufe)	
Spezielle Anforderungen	
Liste der Technik- und Datenvariationen	
Häufigkeit des Auftretens	Häufig
Offene Fragen	

5.5.3.13. scanbasierte Gerätererkennung starten (FR-03)

Der Benutzer kann eine scanbasierte Geräteerkennung starten, welche Geräte in der Broadcastdomain bzw. im Netzwerke automatisch erkennt und hinzufügt.

5.5.3.14. Import/Export der Konfiguration (FR-19)

Ein Benutzer kann Konfigurationen eines SL60-Terminals oder einer Geräte-/Linkgruppe speichern (exportieren) und wiederherstellen (importieren). Die gespeicherten Konfigurationsdaten können mittels eines einfachen Text-Editor von jedermann abgeändert werden.

5.5.3.15. Intervall Parameter für Geräte Erkennung setzen (FR-04)

Ein eingeloggter Benutzer soll die Intervallzeit für das periodische Suchen von Terminals ändern können. Ein default-Wert ist gegeben.0

5.5.3.16. Vordefinierte Templates verwalten und anwenden (FR-11)

Einem eingeloggten Benutzer stehen bereits vordefinierte Templates (verschiedene Standard-Einstellungen) zur Verfügung, welche er auf einem SL60-Terminal bzw. auf eine Geräte-/Linkgruppe anwenden kann. Der Benutzer kann diese Templates zudem verwalten. Für die Templates besteht die Einschränkung, dass Parameter die Gerätespezifisch sind (z.B Zusatzinformationen) ausser Acht gelassen werden müssen.

5.5.3.17. Import/Export eines Templates (FR-20)

Ein eingeloggter Benutzer kann Templates aus der Software exportieren. Diese kann er danach auch wieder importieren.

5.5.3.18. schematische Darstellung der Geräte und Verbindungen (FR-09)

Ein eingeloggter Benutzer kann die Geräte und Verbindungen schematisch darstellen lassen. Die Links sollen in einer Liste dargestellt werden. Dies soll einen Überblick geben, welches Terminal mit welchem anderen Terminal direkt verbunden ist und in welchem Zustand deren Verbindung ist.

5.5.3.19. Systemlog eines Geräts anzeigen (FR-16)

Ein eingeloggter Benutzer kann das System-Log eines bestimmten SL60-Terminals anzeigen lassen und als Text-Datei exportieren.

5.5.3.20. SNMP Trap Ansicht (FR-17)

Ein eingeloggter Benutzer kann veranlassen, dass die SL60-MS sich bei den für die Software sichtbaren SL60-Terminals als SNMP-Trap-Server registriert. Der Benutzer kann anschliessend die SNMP-Trap Nachrichten einsehen und auswerten.

5.5.3.21. Statistiken exportieren und drucken (FR-14)

Statistiken zum Netzwerkverkehr können, als CSV oder als Bild, exportiert oder gedruckt werden.

5.5.3.22. Dedektion mehrerer Instanzen der SL60-MS Software (FR-21)

Das System detektiert automatisch, wenn im selben Netzwerk mehrere SL60MS Instanzen aktiv sind und meldet den Befund dem eingeloggten Benutzer.

5.5.3.23. Logische Fehlkonfiguration erkennen (FR-15)

Logische Fehlkonfiguration, wie doppelt vorhandene IP-Adressen oder syntaktisch nicht korrekte IP-Adressen, werden vom System erkannt und gemeldet.

5.5.4. Schlussfolgerung

Obwohl das Open NMS System viele Funktionalitäten liefern würde, entscheiden wir uns dafür es nicht zu nutzen. Dies aus den Hauptgründen der unvollständigen Dokumentation und der enormen Komplexität. Die einfache Bedienbarkeit welche für die zu entwickelnde SL60-MS nötig ist, wäre nicht gewährleistet. Eine Rücksprache mit Huber+Suhner bestätigte die gemachten Überlegungen. Weitere Netzwerkmanagement Systeme (wie bsp. Nagios) kommen aufgrund ähnlicher Gründe nicht in Frage. Dies ist namentlich die zu hohe Komplexität dieser Systeme. Anhand der Rückmeldung von Huber+Suhner werden auch keine weiteren Systeme betrachtet, da nun als neue Anforderung die Realisierung einer Stand-alone Applikation dazu kam.

5.6. Übersicht der Anforderungsspezifikation

Es wird eine Applikation aufbauend auf der SNMP4J API erstellt, die den Spezifikationen der Anforderungsspezifikation genügt. Diese bietet weniger Funktionalitäten und Komplexität an, als sie mit einem herkömmlichen Netzwerkmanagement System möglich wären. Dafür ist sie genau auf das SL60-Terminal zugeschnitten und wird für den Durchschnittsbenutzer einfach zu bedienen sein.

5.6.1. Grundfunktionen

Die folgend aufgeführten Funktionen bilden zusammen den Hauptnutzen und den Kern der Applikation.

Priorität	**Anforderung**
A	FR-01 Authentisierung
A	FR-10 Konfiguration durchführen
A	FR-12 Konfigurations-History verwalten
A	FR-05 listenbasierte Geräteerkennung
A	FR-18 Geräte gruppieren
A	FR-06 Darstellung der Geräte
A	FR-07 Detailansicht eines Geräts
B	FR-02 Benutzer verwalten inkl. Rollen zuweisen
B	FR-19 Import/Export der Konfiguration
B	FR-11 vordefinierte Templates verwalten und anwenden
B	FR-08 Darstellung der Geräte auf einer Landkarte
B	FR-16 Systemlog eines Geräts anzeigen
B	FR-13 Netzwerkstatistiken anzeigen
B	FR-21 Detektion mehrerer Instanzen der SL60-MS
B	FR-22 Hilfe zu den Konfigurationsparametern anbieten

Tabelle 5.13.: Grundfunktionen der zu realisierenden SL60-MS

6. Domainanalyse

6.1. Einführung

6.1.1. Zweck

Dieser Abschnitt dient als Analyse und als Grundlage für das Design.

6.1.2. Übersicht

Im folgenden Kapitel ist das Domain Modell mit Erläuterungen zu finden. In den darauf kommenden Kapitel wird dann auf Systemsequenzdiagramme und Systemoperationen eingegangen.

6.2. Domain Modell

6.2.1. Strukturdiagramm

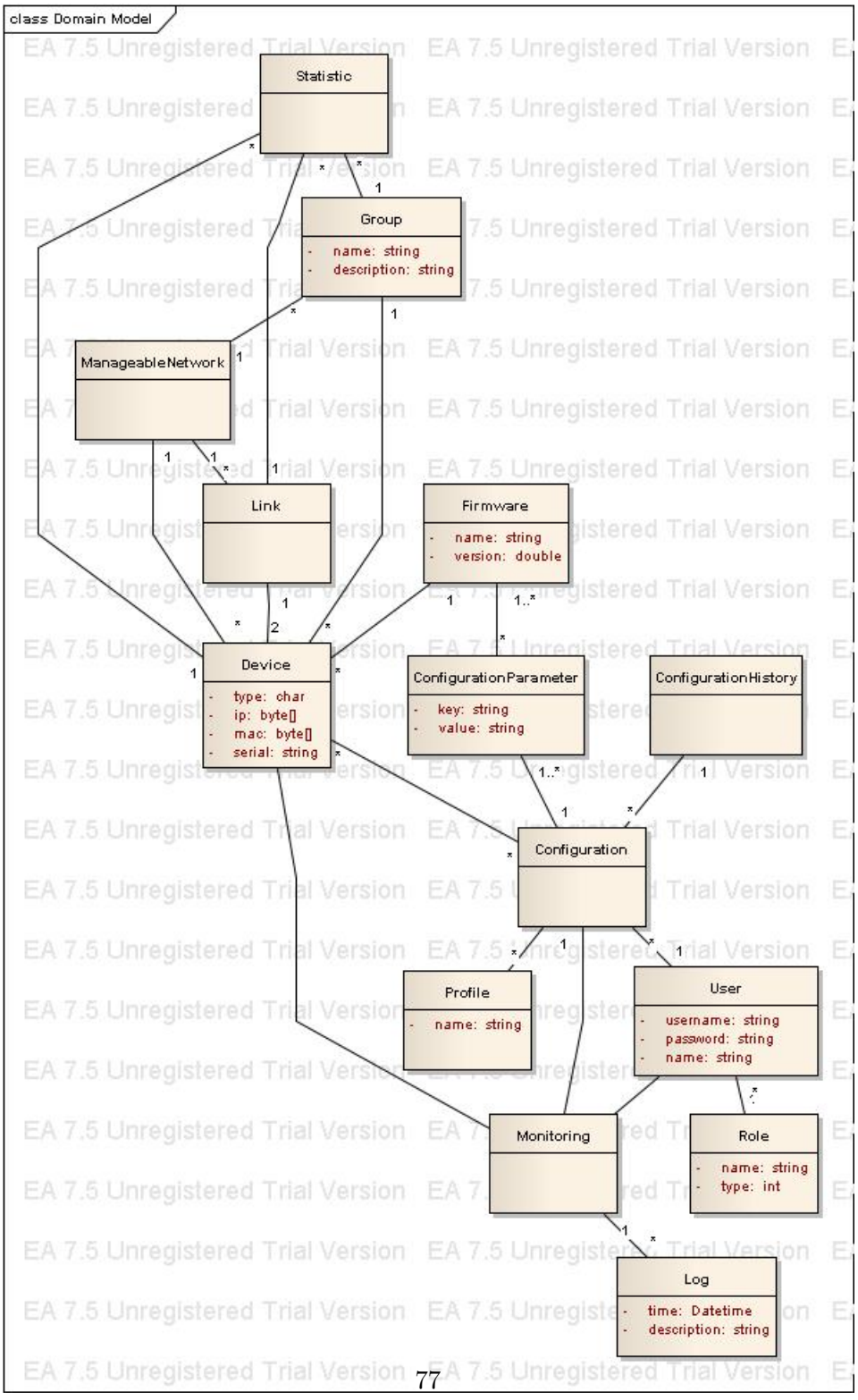


Abbildung 6.1.: Domainmodel - SL60-MS

6.2.2. Konzeptbeschreibung

6.2.2.1. User-Role Konzept

Diagramm

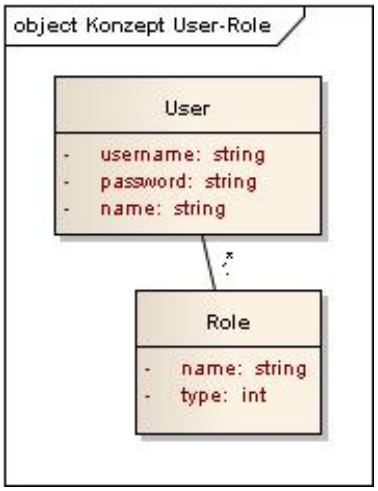


Abbildung 6.2.: User-Role Konzept

Beschreibung
Angelegte Benutzer erhalten ihre Rechte bzw. die Ansichtsmöglichkeiten über die ihnen zugewiesenen Rollen.

Attribute

- User
  - username: Benutzername des Benutzers
  - password: gehashtes Passwort
  - name: Vorname und Name des Benutzers
  - isAdmin: Flag das anzeigt, ob Benutzer das Recht hat, neue Benutzer zu erstellen
- Role
  - name: Name der Rolle
  - type: Typ der Rolle (Admin, User)

Beziehungen
Ein Benutzer hat genau eine Rolle, wobei eine Rolle beliebig vielen Benutzern zugeteilt werden kann.

6.2.2.2. Profil Konzept

Diagramm

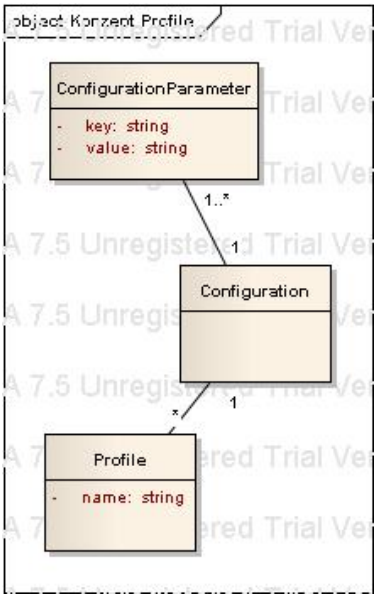


Abbildung 6.3.: Profile Konzept

Beschreibung
Eine einzelne Konfiguration besteht aus einem Schlüssel und Werte Paar. Zusammen werden diese im "ConfigurationParameter" gespeichert. Für eine Konfiguration ("Configuration"), welche direkt auf ein Gerät oder Gerätegruppen angewendet werden kann, benötigt es mindestens einen solchen "ConfigurationParameter". Will man solch eine Konfiguration für die spätere Benutzung speichern, werden diese Informationen als Profil abgelegt.

Attribute

- ConfigurationParameter
  - oid: object identifier, referenziert ein bestimmtes Attribut
  - value: enthält den Wert der für die angegebene oid vorgesehen ist
- Profile
  - name: Name des Profils
- Configuration

Beziehungen
Ein "ConfigurationParameter" gehört genau einer "Configuration" an. Diese kann ihrerseit aber mehrere solche "ConfigurationParameter" enthalten. Eine "Configu-

ration" kann ebenfalls unter Angabe eines Namens als "Profile" gespeichert werden. "Configuration" bzw. "Profiles" können auf einzelne Geräte bzw. Gerätegruppen angewendet werden.

6.2.2.3. Configuration-History Konzept

Diagramm

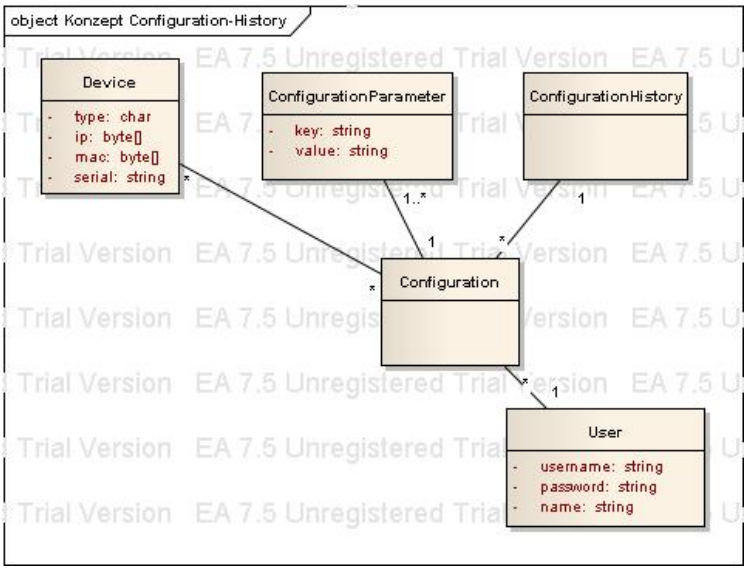


Abbildung 6.4.: Profile Konzept

Beschreibung

Vor der ersten Änderung eines Terminals wird die Grundkonfiguration geladen und in die ConfigurationHistory geschrieben. Nachdem die Änderungen dann auf dem Terminal oder der Terminalgruppe angewendet wurden, werden die von der vorher geladenen Grundkonfiguration abweichenden Werte, in die ConfigurationHistory gesichert. Somit ist gewährleistet, dass zu jedem folgenden Zeitpunkt in den ursprünglichen Zustand gewechselt werden kann.

Attribute

- ConfigurationParameter
  - oid: object identifier, referenziert ein bestimmtes Attribut
  - value: enthält den Wert der für ie angegebene oid vorgesehen ist
- User
  - username: Benutzername des Benutzers
  - password: md5 gehashtes Passwort
  - name: Vorname und Name des Benutzers
- Device = Terminal
  - type: Enthält den Typ des Devices (A/B)

  - ip: Enthält die zugewiesene IP
  - mac: Enthält die physikalische Adresse des Netzwerkadapters
  - serial: Enthält die Seriennummer
- Configuration
- ConfigurationHistory

Beziehungen
In der "ConfigurationHistory" werden von der Grundkonfiguration abweichende Werte gespeichert.

6.2.2.4. Device-Firmware Konzept

Diagramm

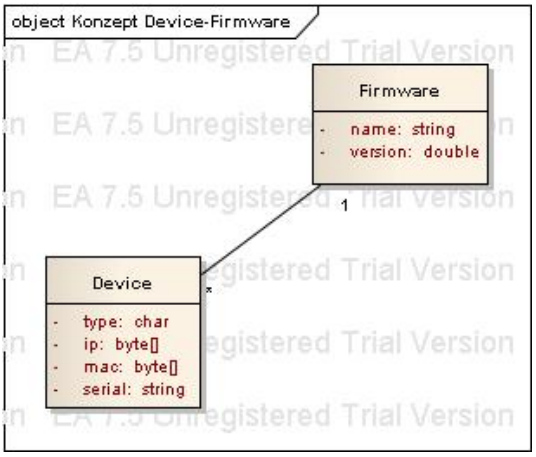


Abbildung 6.5.: Device-Firmware Konzept

Beschreibung
Auf jedem Device läuft eine bestimmte Firmware.

Attribute
- Firmware
  - name: Name der aktuellen Firmware
  - version: Versionsnummer der Firmware
- Device = Terminal
  - type: Enthält den Typ des Devices (A/B)
  - ip: Enthält die zugewiesene IP

- mac: Enthält die physikalische Adresse des Netzwerkadapters
- serial: Enthält die Seriennummer

Beziehungen
Auf jedem Device läuft genau eine Firmware. Diese ist auf beliebig vielen Devices installiert.

6.2.2.5. Device-Group Konzept

Diagramm

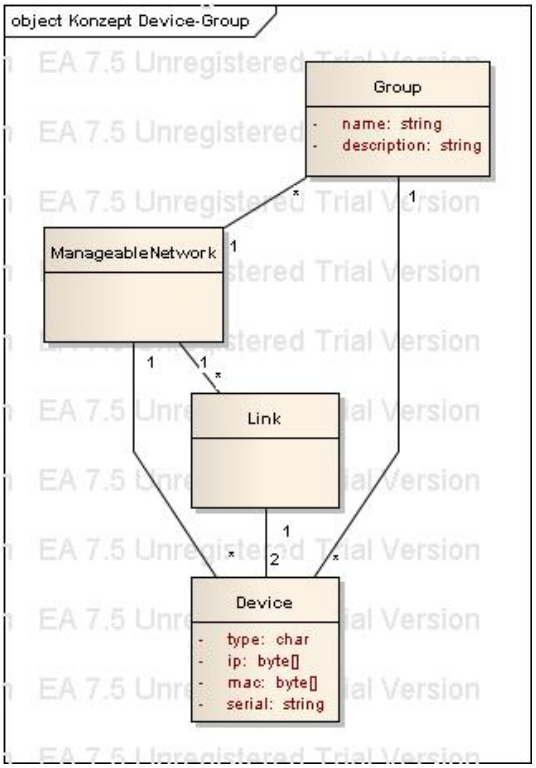


Abbildung 6.6.: Device-Group Konzept

Beschreibung

Jedes Device gehört zu genau einem Link, welcher wiederum zwei Devices beinhaltet. Zudem können Devices genau in einer Gruppe enthalten sein. Näher betrachtet ist ein Link eine Gruppe, welche eine feste Anzahl (zwei) Devices beinhaltet. Alle Links und Gruppen sind in selben Netzwerk. Dabei ist das Netzwerk nicht durch Netzwerkmasken getrennt sondern ist ein dehnbares Konstrukt, welches abhängig von den verwalteten Terminals ist.

Attribute

- Device = Terminal
  - type: Enthält den Typ des Devices (A/B)
  - ip: Enthält die zugewiesene IP
  - mac: Enthält die physikalische Adresse des Netzwerkadapters
  - serial: Enthält die Seriennummer

- Link
- ManageableNetwork
- Group
  - name: Name der Gruppe
  - description: Beschreibung der Gruppe

Beziehungen
Ein Device gehört zu genau einem Link. Ein Link enthält zwei Devices (Typ A und B). Zudem kann ein Device genau zu einer Gruppe gehören. Eine Gruppe kann beliebig viele Devices enthalten. Alle Links und alle Gruppen gehören zum selben Netzwerk ("ManageableNetwork").

6.2.2.6. Statistik Konzept

Diagramm

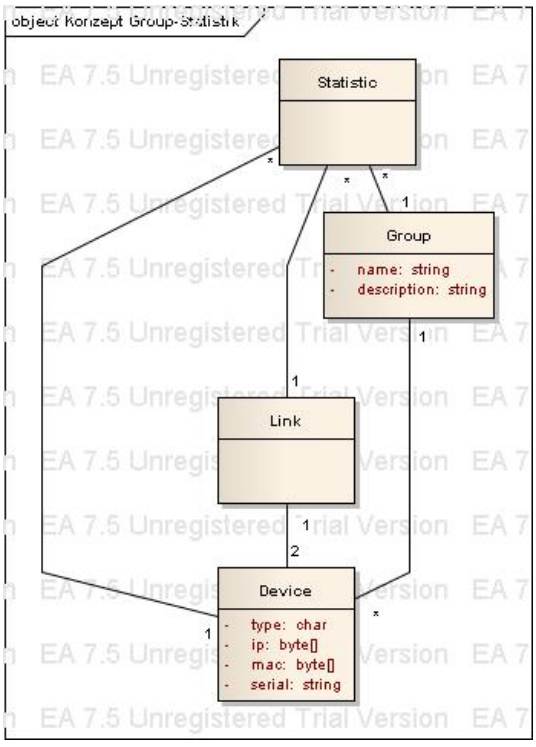


Abbildung 6.7.: Device-Group Konzept

Beschreibung
Statistiken können über Gruppen, Links oder sogar einzelnen Terminals erstellt werden.

Attribute

- Group
  - name: Name der Gruppe
  - description: Beschreibung der Gruppe
- Statistic
- Link
- Device = Terminal
  - type: Enthält den Typ des Devices (A/B)
  - ip: Enthält die zugewiesene IP
  - mac: Enthält die physikalische Adresse des Netzwerkadapters
  - serial: Enthält die Seriennummer

Beziehungen
Eine Gruppe kann verschiedene Statistiken haben (je nach Parametern) und eine Statistik kann für verschiedene Gruppen, Links oder Geräte erstellt werden.

6.2.2.7. Monitoring Konzept

Diagramm

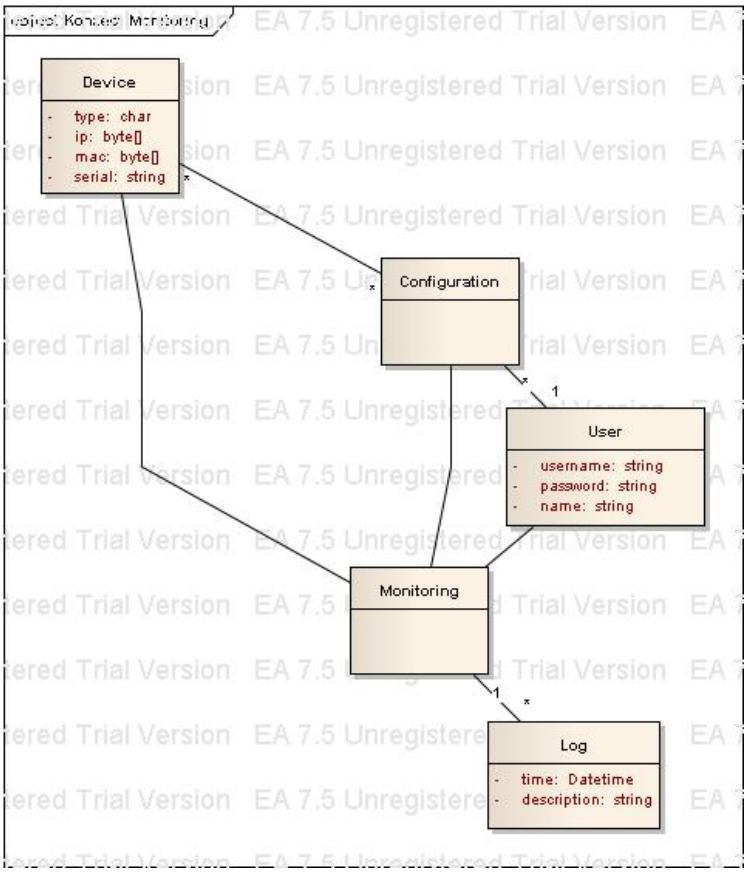


Abbildung 6.8.: Device-Group Konzept

Beschreibung
Das Monitoring überwacht und protokolliert Konfigurationen auf Terminals, Gruppen oder Links. Somit können Nutzungsstatistiken über angesteuerte Geräte und Fehler erstellt werden.

Attribute

- Monitoring
- Log
  - time: Zeit des Log-Eintrags
  - description: Beschreibung der Aktion
- Configuration

- User
  - username: Benutzername des Benutzers
  - password: md5 gehashtes Passwort
  - name: Vorname und Name des Benutzers
- Device = Terminal
  - type: Enthält den Typ des Devices (A/B)
  - ip: Enthält die zugewiesene IP
  - mac: Enthält die physikalische Adresse des Netzwerkadapters
  - serial: Enthält die Seriennummer des Gerätes

Beziehungen

Das Monitoring besitzt mehrere Log-Einträge. Da pro Applikationsinstanz jeweils nur ein Monitoring existiert, gehört jeder Log-Eintrag zu genau einem Monitoring. Für die Logeinträge muss das Monitoring alle Aktionen eines Benutzer Bescheid wissen. Dies bedingt die Beziehungen vom Monitoring zum User, Configuration und Device.

6.3. Systemsequenzdiagramme

6.3.1. UC-01 - Geräte verwalten

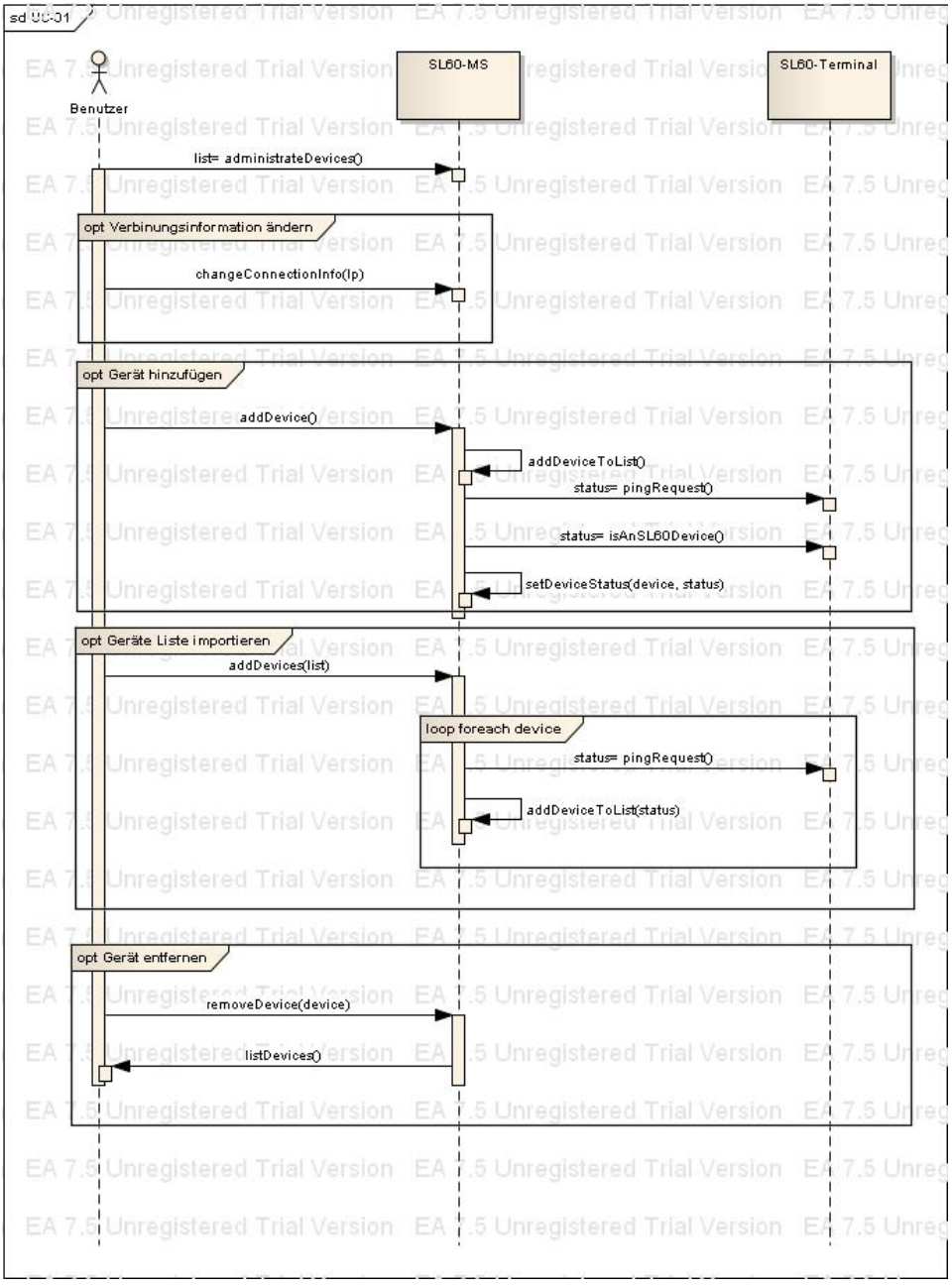


Abbildung 6.9.: Use Case 01 - Geräte verwalten

6.3.2. UC-02 - Gerätekonfiguration durchführen

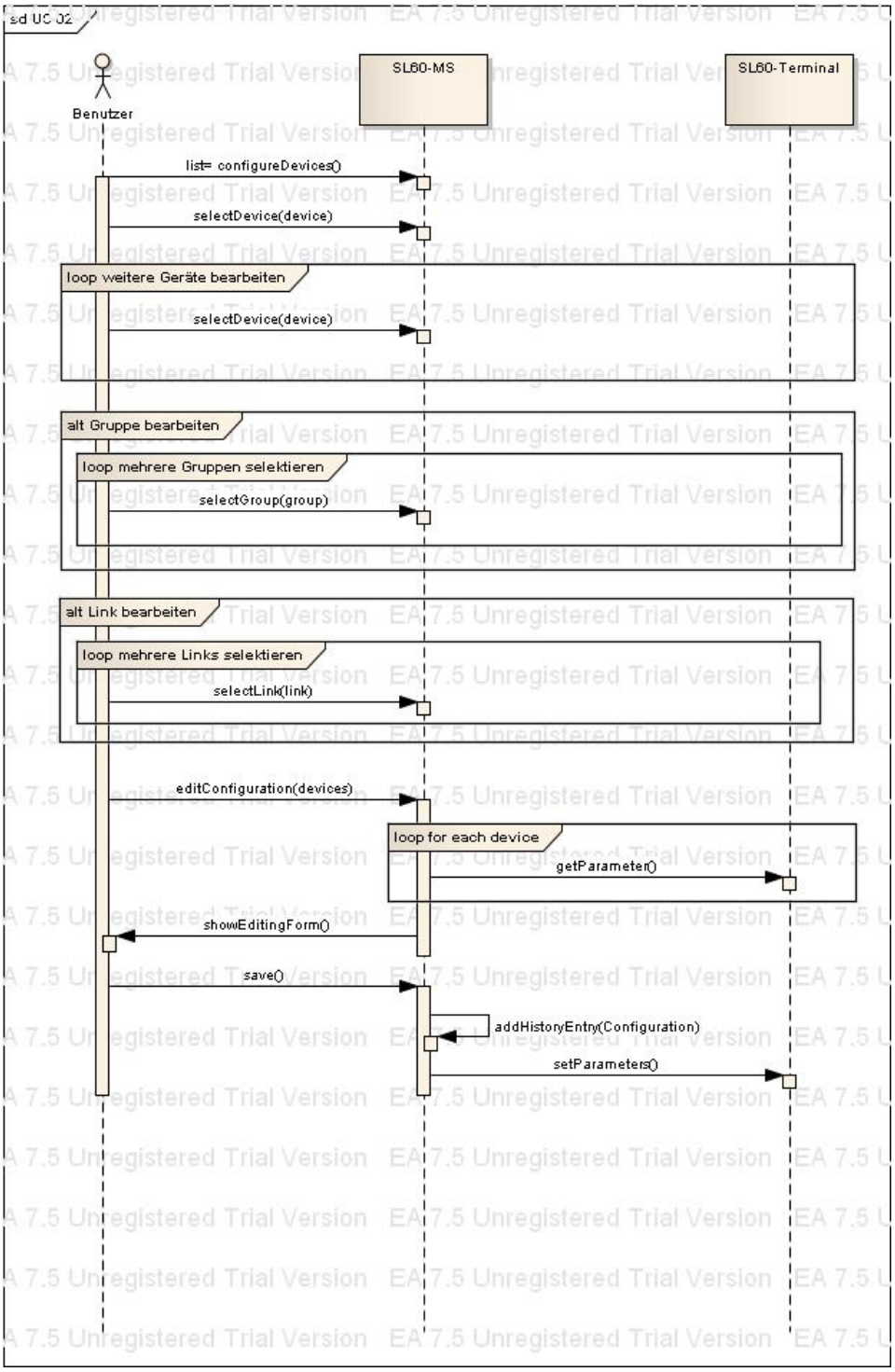


Abbildung 6.10.: Use Case 02 - Gerätekonfiguration durchführen

6.3.3. UC03 - Geräte gruppieren

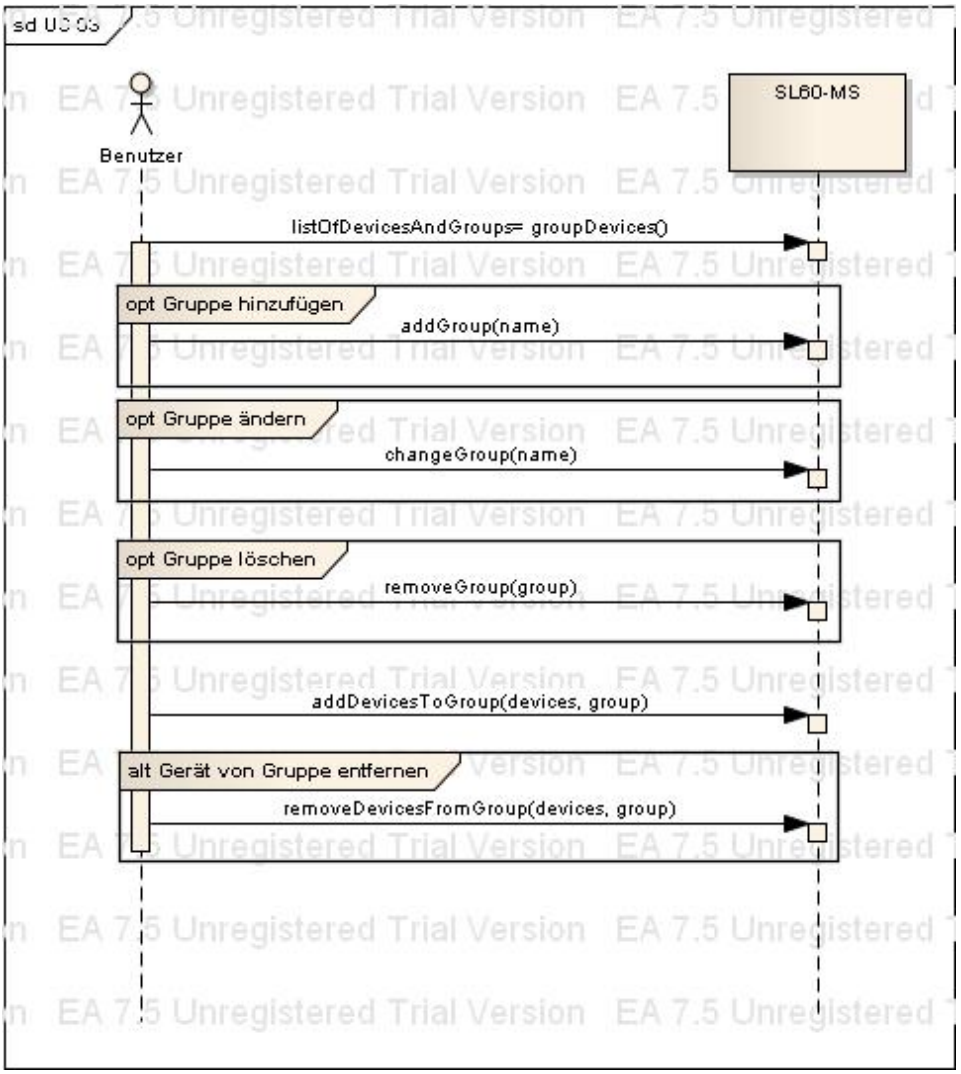


Abbildung 6.11.: Use Case 03 - Geräte gruppieren

6.3.4. UC05 - Geräte darstellen lassen

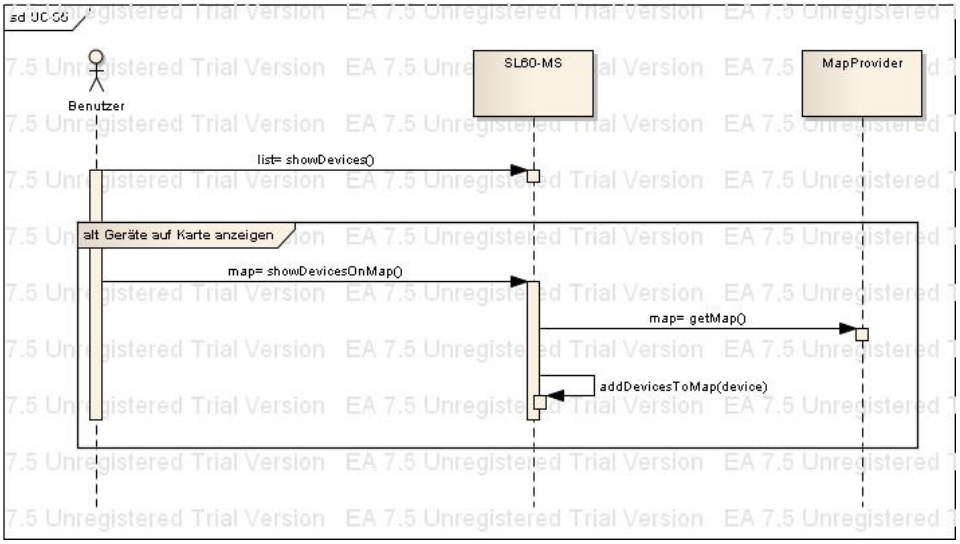


Abbildung 6.12.: Use Case 05 - Geräte darstellen lassen

6.3.5. UC06 - Parameter eines Gerätes detailliert anzeigen

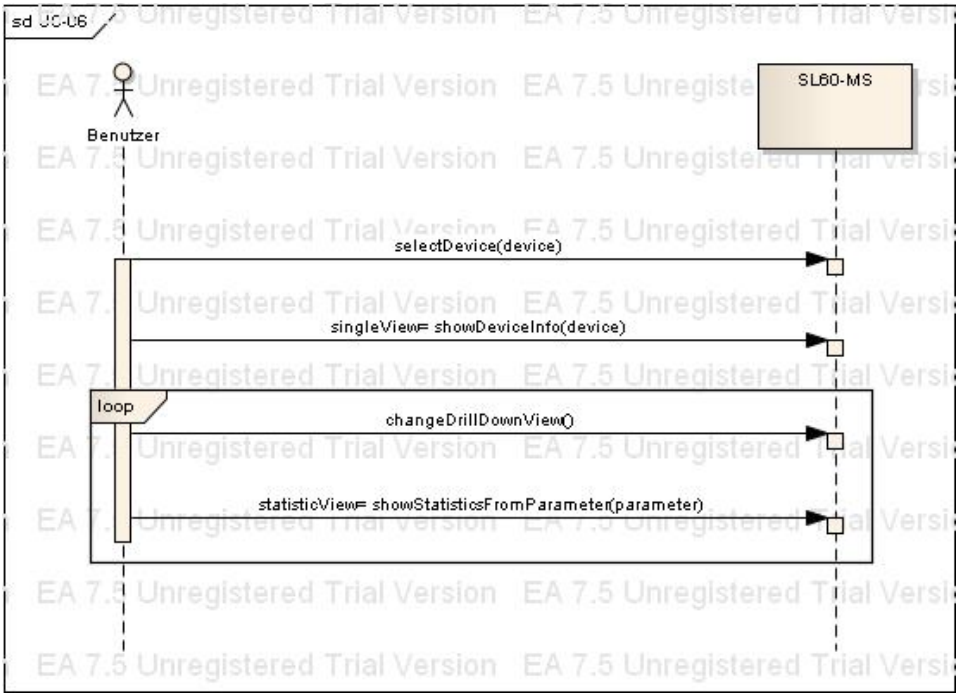


Abbildung 6.13.: Use Case 06 - Parameter eines Gerätes detailliert anzeigen

6.3.6. UC07 - Konfigurationshistory anzeigen lassen

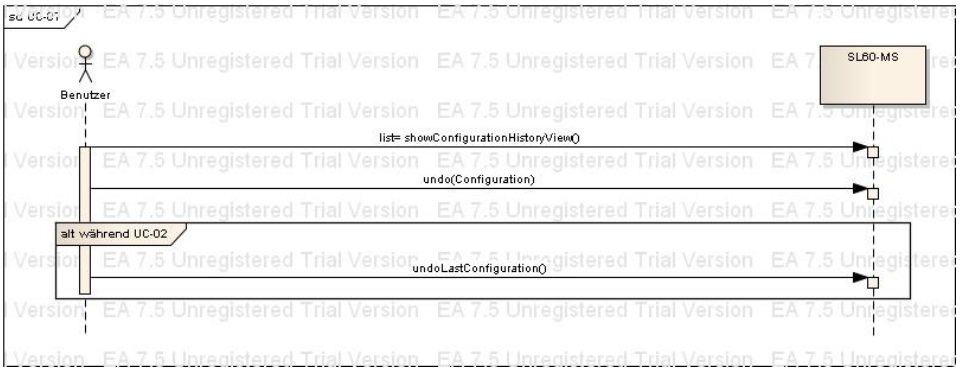


Abbildung 6.14.: Use Case 07 - Konfigurationshistory anzeigen lassen

6.3.7. UC08 - Authentisierung durchführen

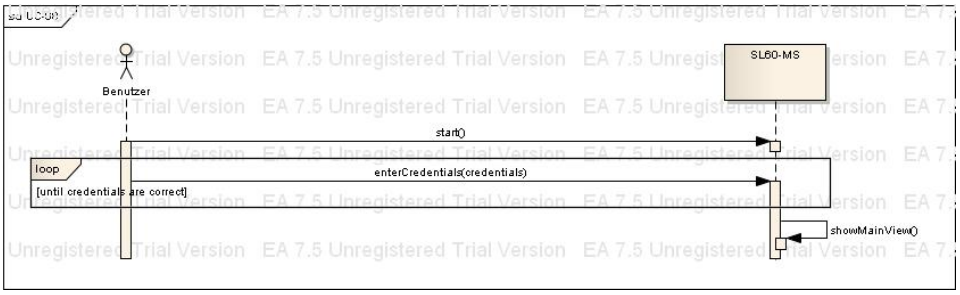


Abbildung 6.15.: Use Case 08 - Authentisierung durchführen

6.4. Systemoperationen

6.4.1. CO-01 - administrateDevices

Operation	administrateDevices()
Querverweis	UC-01 - Geräte verwalten
Vorbedingungen	Benutzer muss eingeloggt sein.
Nachbedingungen	Der Benutzer sieht eine Maske für die Administration von Terminals.

6.4.2. CO-02 - changeConnectionInfo

Operation	changeConnectionInfo(ip: String)
Querverweis	UC-01 - Geräte verwalten
Vorbedingungen	Benutzer muss ein Terminal aus der Liste selektiert haben.
Nachbedingungen	Die ConnectionInfo wurde geändert.

6.4.3. CO-03 - addDevice

Operation	addDevice(ip: String)
Querverweis	UC-01 - Geräte verwalten
Vorbedingungen	Benutzer muss die Maske für das Administrieren von Terminals sehen.
Nachbedingungen	Ein neues Device wurde in die Liste aufgenommen

6.4.4. CO-04 - addDevices

Operation	addDevices(ips: vector¡ip:String¿)
Querverweis	UC-01 - Geräte verwalten
Vorbedingungen	Benutzer muss die Maske für das Administrieren von Terminals sehen.
Nachbedingungen	Alle übergebenen Devices wurden in die Liste übernommen.

6.4.5. CO-05 - configureDevices

Operation	configureDevices()
Querverweis	UC-02 - Gerätekonfiguration durchführen
Vorbedingungen	Benutzer muss sich eingeloggt haben.
Nachbedingungen	Eine Ansicht aller zu Verfügung stehenden Devices wird dem Benutzer angezeigt.

6.4.6. CO-06 - selectDevice

Operation	selectDevice(device: Device)
Querverweis	UC-02 - Gerätekonfiguration durchführen
Vorbedingungen	Benutzer muss die zur Verfügung stehenden Devices sehen können.
Nachbedingungen	Ein Terminal wurde zur Auswahl hinzugefügt.

6.4.7. CO-07 - selectGroup

Operation	selectGroup(group: Group)
Querverweis	UC-02 - Gerätekonfiguration durchführen
Vorbedingungen	Benutzer muss die zu Verfügung stehenden Gruppen sehen können.
Nachbedingungen	Eine oder mehrere Gruppen wurden zur Auswahl hinzugefügt.

6.4.8. CO-08 - selectLink

Operation	selectLink(link: Link)
Querverweis	UC-02 - Gerätekonfiguration durchführen
Vorbedingungen	Benutzer muss die zur Verfügung stehenden Links sehen können.
Nachbedingungen	Eine oder mehrere Links wurden zur Auswahl hinzugefügt.

6.4.9. CO-09 - editConfiguration

Operation	editConfiguraion(devices: vecotr¡device: Device¿)
Querverweis	UC-02 - Gerätekonfiguration durchführen
Vorbedingungen	Benutzer muss Terminals, Links oder Gruppen selektiert haben.
Nachbedingungen	Eine Bearbeitungsmaske für die zu editierende Selektion wird angezeigt.

6.4.10. CO-10 - save

Operation	save()
Querverweis	UC-02 - Gerätekonfiguration durchführen
Vorbedingungen	Benutzer hat Änderungen in der Maske eingegeben.
Nachbedingungen	Die Änderungen für die selektierten Terminals, Links oder Gruppen wurden in der ConfigurationHistory übernommen und auf den Terminals angewandt.

6.4.11. CO-11 - groupDevices

Operation	groupDevices()
Querverweis	UC-03 - Geräte gruppieren
Vorbedingungen	Benutzer hat sich eingeloggt.
Nachbedingungen	Eine Liste der bereits erstellten Gruppen und gefundenen Terminals wird dem Benutzer angezeigt.

6.4.12. CO-12 - addGroup

Operation	addGroup(name: String)
Querverweis	UC-03 - Geräte gruppieren
Vorbedingungen	Benutzer hat sich eingeloggt.
Nachbedingungen	Eine neue Gruppe mit dem mitgegebenen Namen wurde erstellt und erscheint in der Liste der bestehenden Gruppen.

6.4.13. CO-13 - changeGroup

Operation	changeGroup(name: String)
Querverweis	UC-03 - Geräte gruppieren
Vorbedingungen	Benutzer hat eine Gruppe selektiert deren Namen er ändern will
Nachbedingungen	Der Name der Gruppe wurde auf den mitgegebenen neuen Namen geändert. Der neue Gruppenname wird auch in der Liste aktualisiert.

6.4.14. CO-14 - removeGroup

Operation	removeGroup(group: Group)
Querverweis	UC-03 - Geräte gruppieren
Vorbedingungen	Benutzer hat eine Gruppe selektiert welche er löschen will
Nachbedingungen	Die Gruppe wurde gelöscht und auch aus der Liste entfernt.

6.4.15. CO-15 - addDevicesToGroup

Operation	addDevicesToGroup(devices: vector¡device: Device¿, group: Group)
Querverweis	UC-03 - Geräte gruppieren
Vorbedingungen	Benutzer hat eine Auswahl von Terminals getroffen, die er der Gruppe hinzufügen will.
Nachbedingungen	Die ausgewählten Terminals wurden der ausgewählten Gruppe hinzugefügt.

6.4.16. CO-16 - removeDeviceFromGroup

Operation	removeDevicesFromGroup(devices: vector¡device: Device¿, group: Group)
Querverweis	UC-03 - Geräte gruppieren
Vorbedingungen	Benutzer hat eine Auswahl von Terminals getroffen, die er von der Gruppe entfernen will.
Nachbedingungen	Die ausgewählten Terminals wurden von der ausgewählten Gruppe entfernt.

6.4.17. CO-17 - showDevices

Operation	showDevices()
Querverweis	UC-05 - Geräte darstellen lassen
Vorbedingungen	Benutzer hat sich eingeloggt.
Nachbedingungen	Benutzer kann die erkannten Terminals in einer Liste sehen.

6.4.18. CO-18 - showDevicesOnMap

Operation	showDevicesOnMap()
Querverweis	UC-05 - Geräte darstellen lassen
Vorbedingungen	Benutzer hat sich eingeloggt.
Nachbedingungen	Benutzer kann die erkannten Geräte auf einer Karte sehen.

6.4.19. CO-19 - selectDevice

Operation	selectDevice()
Querverweis	UC-06 - Parameter eines Gerätes detailliert anzeigen
Vorbedingungen	Benutzer hat das Terminal, von welchem er die detaillierten Parameter anzeigen will, ausgewählt.
Nachbedingungen	Benutzer sieht eine detaillierte Ansicht der Parameter des ausgewählten Terminals.

6.4.20. CO-20 - changeDrillDownView

Operation	changeDrillDownView()
Querverweis	UC-06 - Parameter eines Gerätes detailliert anzeigen
Vorbedingungen	Benutzer sieht bereits Parameter eines Terminals.
Nachbedingungen	Benutzer hat nun eine tiefere bzw. detailliertere Ansicht des gewünschten Parameter.

6.4.21. CO-21 - showStatisticsOfParameter

Operation	showStatisticsOfParameter(parameter: String)
Querverweis	UC-06 - Parameter eines Gerätes detailliert anzeigen
Vorbedingungen	Benutzer hat ein Parameter ausgewählt.
Nachbedingungen	Benutzer sieht eine statistische Aufstellung der vorherigen Werte des ausgewählten Parameters.

6.4.22. CO-22 - showConfigurationHistory

Operation	showConfigurationHistory()
Querverweis	UC-07 - Konfigurationhistory anzeigen lassen
Vorbedingungen	Benutzer hat sich eingeloggt.
Nachbedingungen	Benutzer sieht eine Aufstellung der letzten zehn gemachten Änderungen.

6.4.23. CO-23 - undo

Operation	undo(configuration: Configuration)
Querverweis	UC-07 - Konfigurationhistory anzeigen lassen
Vorbedingungen	Benutzer hat eine Konfiguration selektiert.
Nachbedingungen	Die Änderungen, die durch die selektierte Konfiguration getätigt wurden, wurden zurückgesetzt.

6.4.24. CO-24 - undoLastConfiguration

Operation	undoLastConfiguration()
Querverweis	UC-07 - Konfigurationhistory anzeigen lassen, UC-02 - Gerätekonfiguration durchführen
Vorbedingungen	Benutzer hat im Rahmen des UC-02 eine Änderung durchgeführt.
Nachbedingungen	Die Änderung für dieses Terminal, die Gruppe oder den Link wurden rückgängig gemacht.

6.4.25. CO-25 - start

Operation	start()
Querverweis	UC-08 - Authentisierung durchführen
Vorbedingungen	Benutzer hat die Software installiert.
Nachbedingungen	Benutzer hat die Software aufgestartet und sieht die Hauptansicht der Applikation.

6.4.26. CO-25 - enterCredentials

Operation	enterCredentials(credentials: Credentials)
Querverweis	UC-08 - Authentisierung durchführen
Vorbedingungen	Benutzer hat die Software auf aufgestartet.
Nachbedingungen	Benutzer hat sich erfolgreich authentisiert und sieht die Hauptansicht der Applikation.

7. Software Architektur Spezifikation

7.1. Einleitung

7.1.1. Zweck

In diesem Kapitel wird die Architektur der SL60-MS, sowie die Entscheide die dazu führten, beschrieben.

7.1.2. Übersicht

Das Dokument zeigt alle wesentlichen architektonischen Entscheidungen in einem Überblick.

7.2. Architektonische Darstellung

Die folgende Tabelle und die Abbildung zeigen die externen Abhängigkeiten der SL60-MS auf.

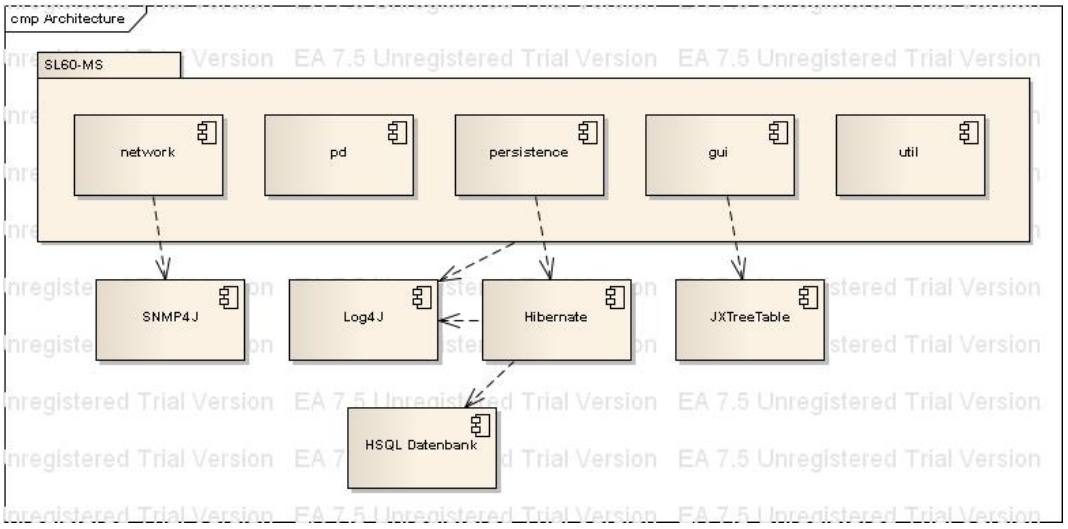


Abbildung 7.1.: externe Abhängigkeiten

Komponente	Version	Beschreibung
SNMP4J	1.10.2	API welche die SNMP Kommunikation übernimmt.
Log4J	1.2.15	Komponente welche das Logging übernimmt
Hibernate	3.3.2.GA	ORM Mapper. Übernimmt die persistente Speicherung der Daten.
HSQL Datenbank	1.8.1.1	Relationale Datenbank
JXTreeTable	1.6	Komponente aus der SwingX Bibliothekt welche die GUI-Kompontente TreeTable bereitstellt.

Tabelle 7.1.: Übersicht der externen Komponenten

7.3. Architektonische Ziele & Einschränkungen

7.3.1. Sicherheit & Privacy

7.3.2. Distribution

Die Distribution der Software erfolgt über ein ZIP-File. Der entpackte Inhalt besteht aus einer ausführbaren Jar-Datei, einem Ordner mit Jar-Bibliotheken sowie einem Ordner der die HSQL Datenbank enthält.

7.4. logische Architektur

7.4.1. Aufbau

Der Aufbau erfolgt nach dem bekannten Model View Controller Konzept (MVC). Die Softwareschichten sehen im Folgenden so aus:

gui
Die View befindet sich in dieser Schicht. Sie wird über das Swing-Framework bereitgestellt.

model
Kapselt die Problem Domain. Sie enthält die Klassendefinitionen der vorhandenen Objekte. Zusätzlich gehört auch die View-Erstellungslogik (Controller) zu dieser Schicht.

network
Die Kommunikation über SNMP und HTTP wird vollständig von dieser Schicht übernommen.

persistence
Das persistente Speichern der Objekte und die Kommunikation mit Hibernate läuft in dieser Schicht ab.

tests
Enthält alle Unit-Tests der Software.

util
Enthält Hilfsklassen die in keines der anderen Packages passen.

7.4.2. interne Abhängigkeiten

Die internen Abhängigkeiten der einzelnen Packages sind in der folgenden Abbildung dargestellt.

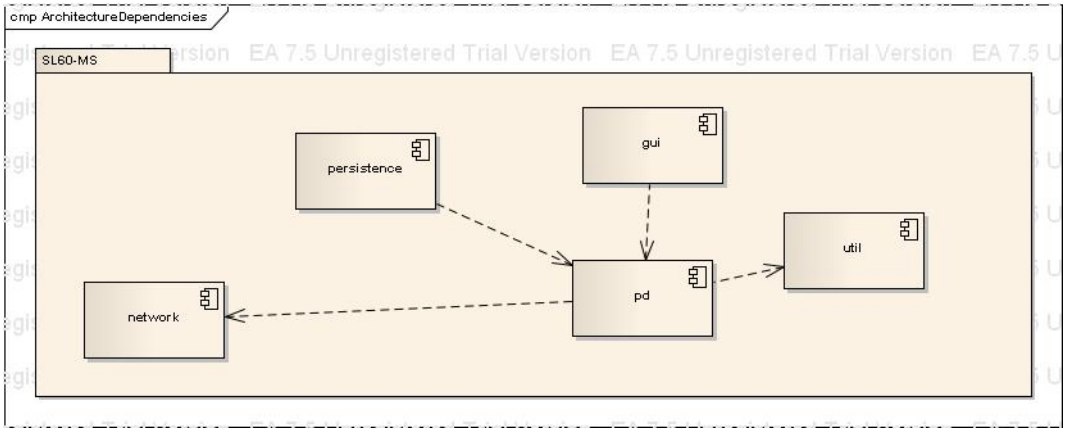


Abbildung 7.2.: Architekturübersicht mit internen Abhängigkeiten

7.4.3. Verwendete Design Patterns

Es ist grundsätzlich schwierig jedes in dieser Software verwendete Pattern aufzulisten. Aus diesem Grund werden hier nur die wichtigsten und grundlegendsten erwähnt.

- Im persistence Package wurde das Pattern "A generic data-access object pattern" [BK06, S. 708] für die Erstellung der Data-Access Objekte verwendet.

- Im network Package wurde für die Erfassung und Abarbeitung der Request das "Half-Sync/Half-Async Pattern" [BMRS96, S. 423] verwendet. Ausschlaggebend für die Wahl dieses Patterns war einerseits das Bestehen eines asynchronen und eines synchronen Layers (die zusammen arbeiten) und andererseits die Möglichkeit gewisse Requests zu priorisieren.

7.4.4. Design Pakete

7.4.4.1. Package gui

Das gui Package enthält die Klassen welche die Benutzerschnittstelle bereitstellen. Grösstenteils werden gewöhnliche Swing Komponenten verwendet. Für die mehrstufigen Anzeigen der Gruppen und Geräte, sowie der Konfigurationseingabemaske konnte keine geeignete Variante innerhalb der Swing Bibliothek gefunden werden. Darum wird für diese Anzeigen auf die TreeTable-Komponente der externen Bibliothek SwingX zurückgegriffen.
Die darzustellenden Benutzerschaltflächen wurden in Panels aufgeteilt, welche ineinander verschachtelt sind. Dies hat den Vorteil, dass solche "Unterpanels" mehrfach in verschiedenen Kontexten gebraucht werden können.

Diagramme

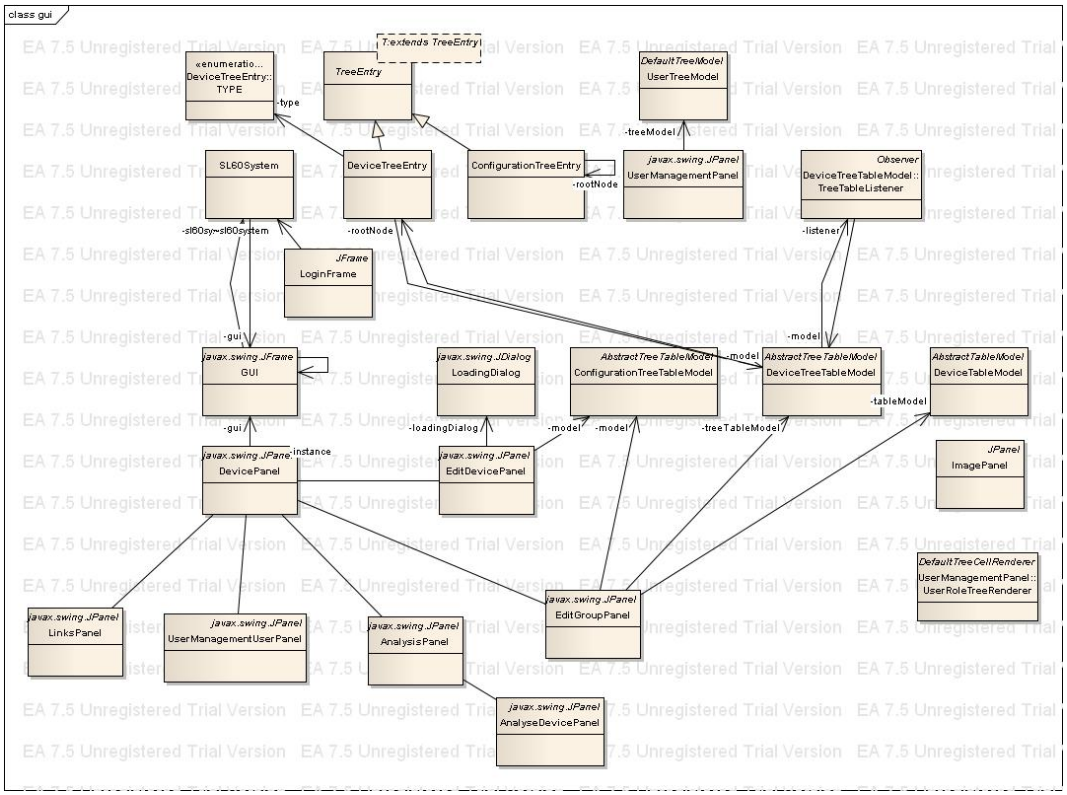


Abbildung 7.3.: Architektur des GUI Packages

Schnittstellen

Jede Klasse im GUI Package bietet Get-Methoden für einzelne Swing Komponenten an. Diese Methoden existieren für Komponenten die ihren Zustand ändern oder bei einem Ereignis eine Aktion auslösen sollen (Tabellen, Buttons).

Operationen

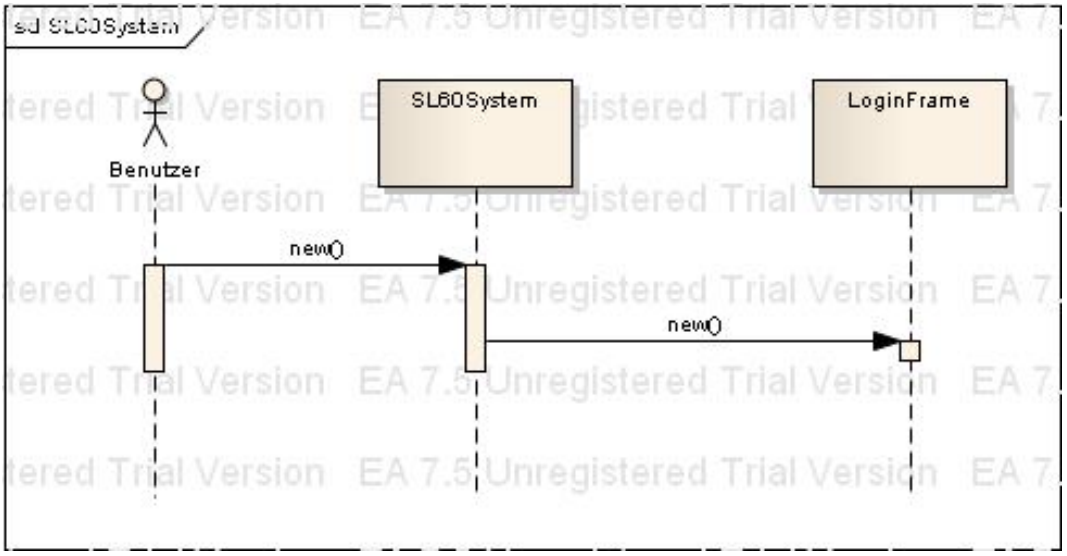


Abbildung 7.4.: Starten der Applikation

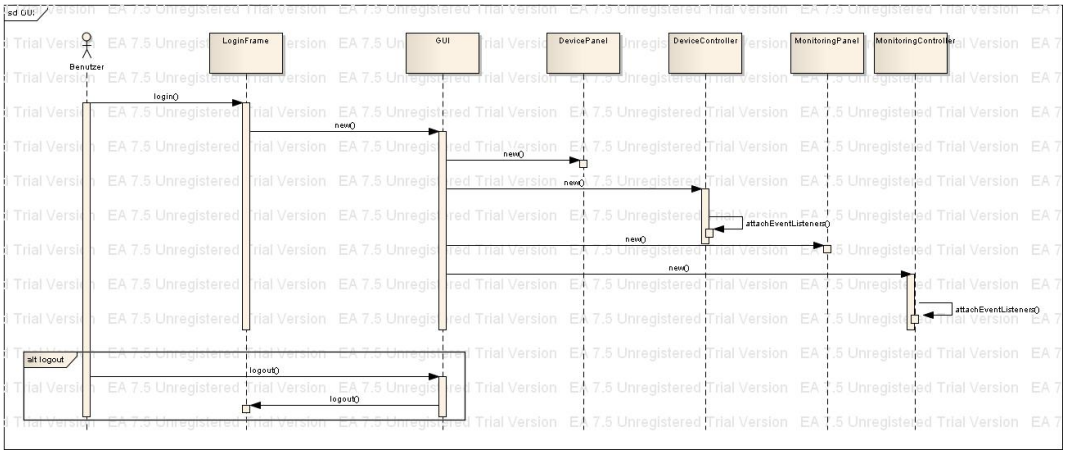


Abbildung 7.5.: Login eines Benutzers

7.4.4.2. Package pd

Das Problem Domain Package beinhaltet die Applikationslogik der SL60-MS. Klassen welche in der Datenbank gespeichert werden, sind über JPA Syntax entsprechend

annotiert. Die Klasse NetworkManager ist der eigentliche Kern dieses Packages. Sie verwaltet Objekt-Listen die in der ganzen Applikation benutzt werden. Beispielsweise die lokale Liste der Geräte und der Gruppen. Sobald Datenbank-relevante Daten verändert werden, werden diese Änderungen automatisch persistiert.

Diagramme

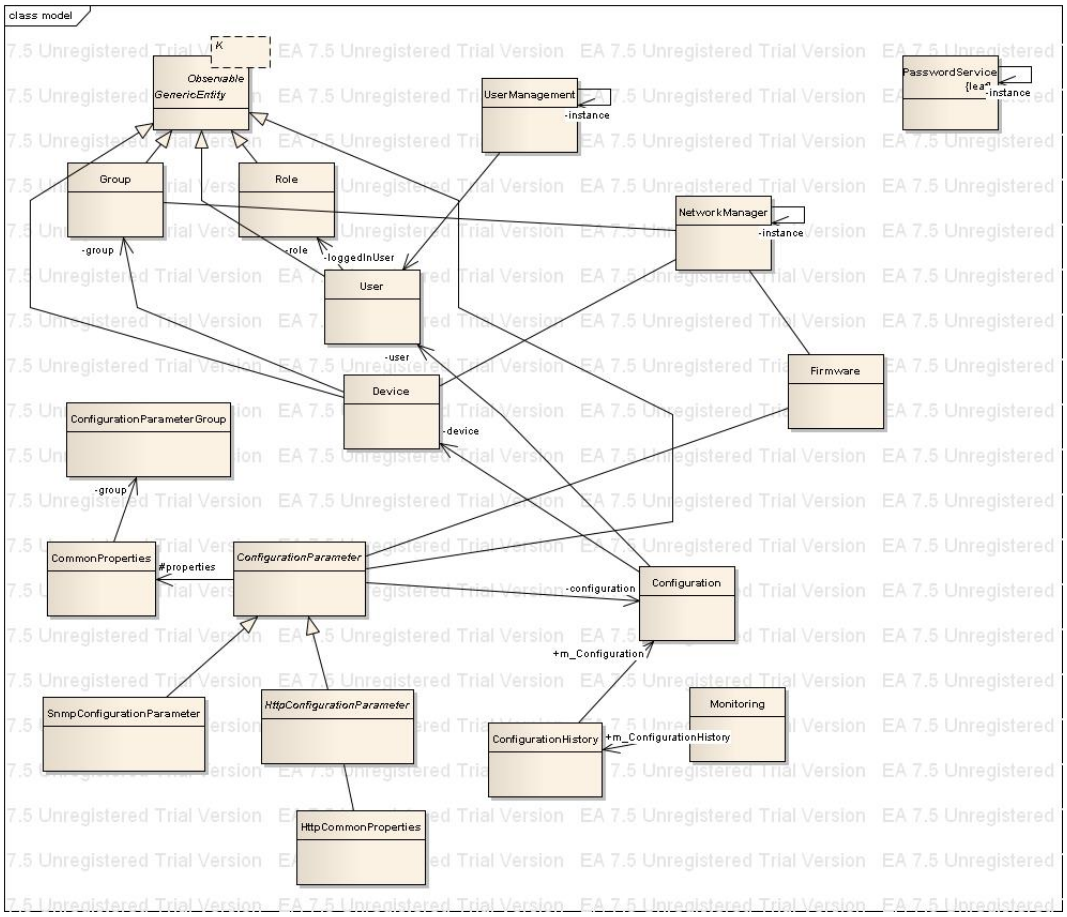


Abbildung 7.6.: Architektur der Problem Domain

Schnittstellen

Die Klassen die vom Persistence Layer in der Datenbank gespeichert werden, sind in JPA und Hibernate Syntax annotiert. Diese Annotations bestimmen die Aufteilung der Problem Domain Klassen auf Datenbank Tabellen.

Operationen

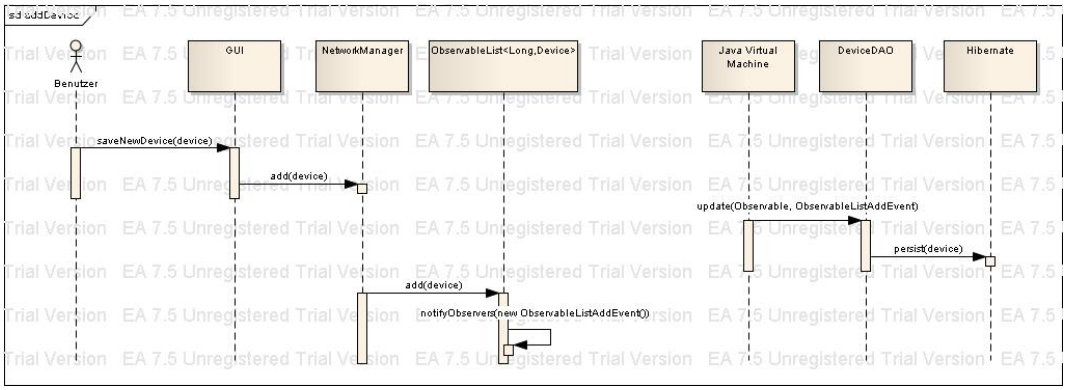


Abbildung 7.7.: Hinzufügen eines Geräts

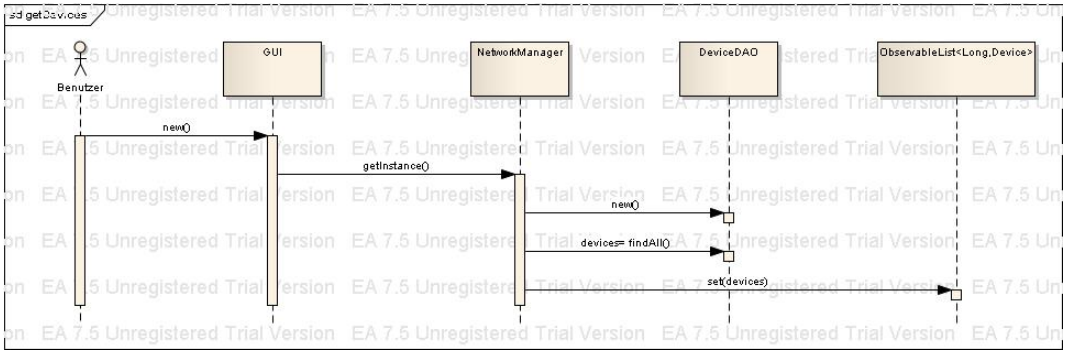


Abbildung 7.8.: Auslesen von Geräten

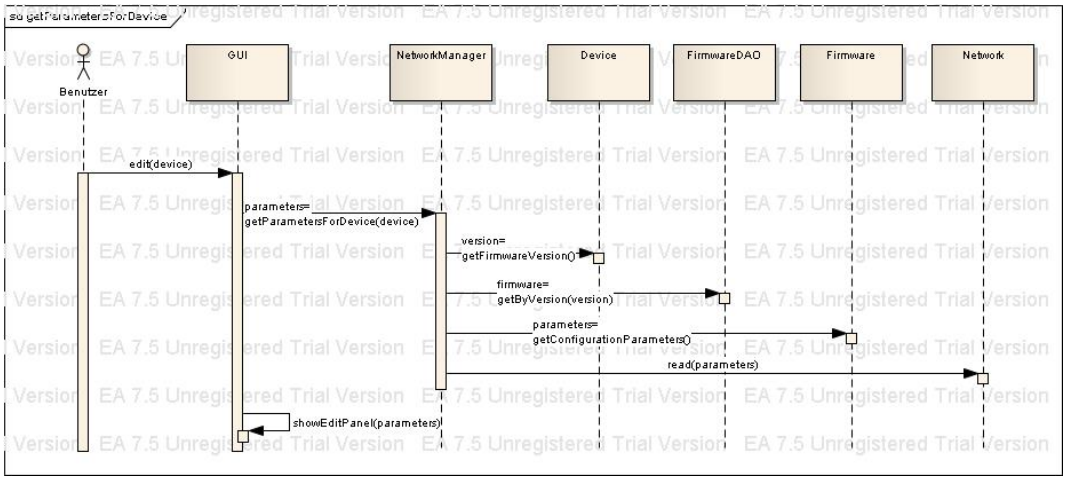


Abbildung 7.9.: Auslesen von Parametern eines Gerätes um eine Konfigurationsmaske darzustellen

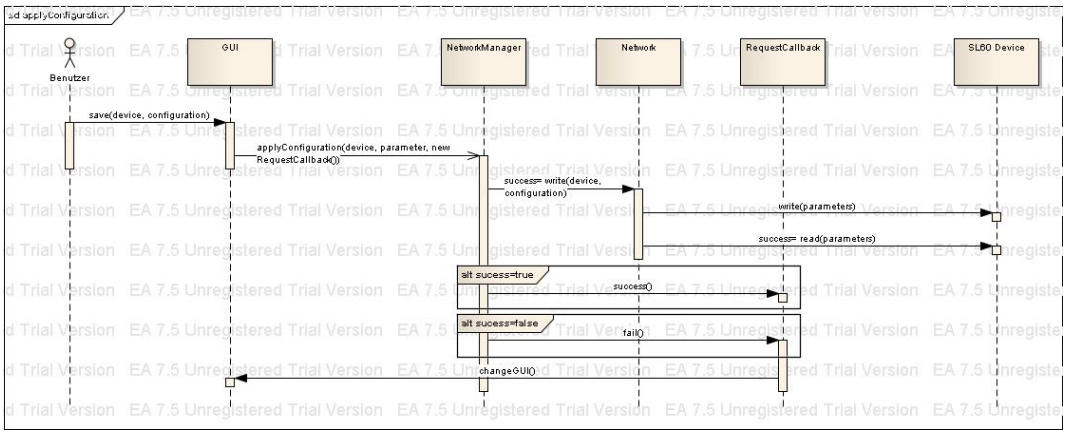


Abbildung 7.10.: Konfiguration auf ein Gerät anwenden

7.4.4.3. Package network

Das network Package regelt jegliche Kommunikation mit den Geräten. Abhängig vom Typ des Requests behandelt das network Package die Kommunikation anders. Aus der Sicht anderer Packages wirkt das network Package wie eine Blackbox.
Das Kernstück des network Package ist die NetworkCommunicator-Klasse. Sie initialisiert die verschiedenen Queues und die dazugehörigen Thread-Pools. In die Queue werden hereinkommende Requests abgelegt. Die zu den Queues gehörenden Thread-Pools lesen die Requests aus den Queues wieder aus und bearbeiten diese. Sobald der Request abgehandelt wurde, wird das Resultat in die ReceiverRequestQueue geschrieben.
Die NetworkCommunicator-Klasse wird mit Objekten instanziiert welche die Interfaces IRead und IWrite implementieren. Das bedeutet das sie das Lesen bzw. Schreiben mit einem bestimmten Protokoll implementieren. Somit lassen sich mit geringem Aufwand neue Protokolle hinzufügen. Im jetzigen Zustand gibt es die Möglichkeit über SNMP (mittels SnmpService) zu lesen und über HTTP (mittels HttpService) zu lesen und zu schreiben. Die im vorherigen Abschnitt erwähnten Threads bzw. der Thread-Pool, benutzen diese IRead- bzw. IWrite-Objekte um den Request zu behandeln.
Die RequestManger-Klasse ist die eigentliche Schnittstelle nach aussen. Wie diese funktioniert wird im Kapitel 7.4.4.3 erläutert.

Diagramme

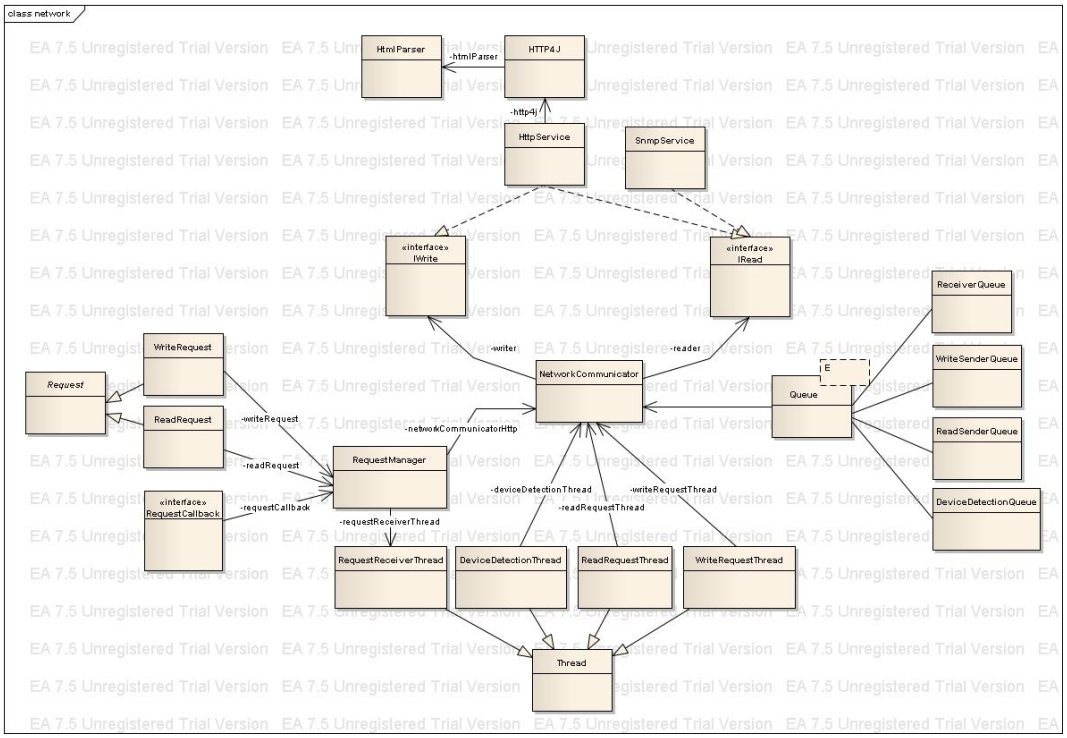


Abbildung 7.11.: Architektur des Network Packages

Schnittstellen

Die RequestManager-Klasse ist die Schnittstelle für andere Packages, die den Dienst des Netzwerks nutzen wollen. Die RequestManager-Klasse instanziiert die NetworkCommunicator-Klassen für die verschiedenen Kommunikationsarten. In aktuellen Fall, einen für das Lesen und Schreiben über HTTP und einen für das Lesen über SNMP.
Weiter startet die RequestManager-Klasse für jede Kommunikationsart einen Thread, welcher dessen ReceiverRequestQueue abhört und abgehandelte Requests empfängt.
Um über den RequestMangager Requests absetzen zu können bedarf es eines Objekts, welches das Interface RequestCallback implementiert. Das RequestCallback-Objekt wird zusammen mit dem Request in einer Liste gespeichert. Sobald dieser Request bearbeitet bzw. vom gestarteten Thread empfangen wurde, wird die Callback-Funktion des dazugehörigen RequestCallback-Objekts aufgerufen. Dieser führt dann in den meisten Fällen ein Änderung am GUI durch.

Operationen

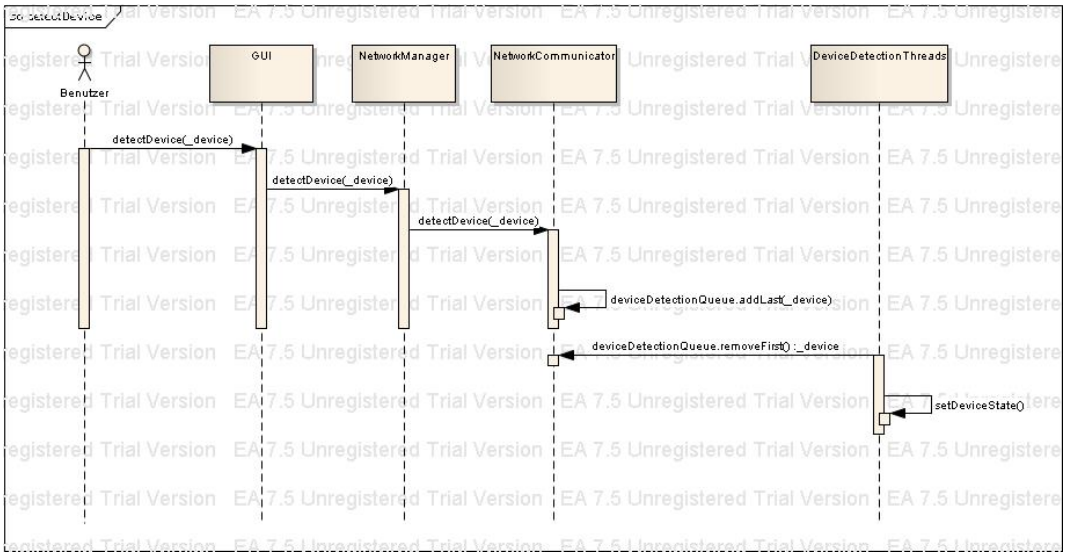


Abbildung 7.12.: Gerät detektieren

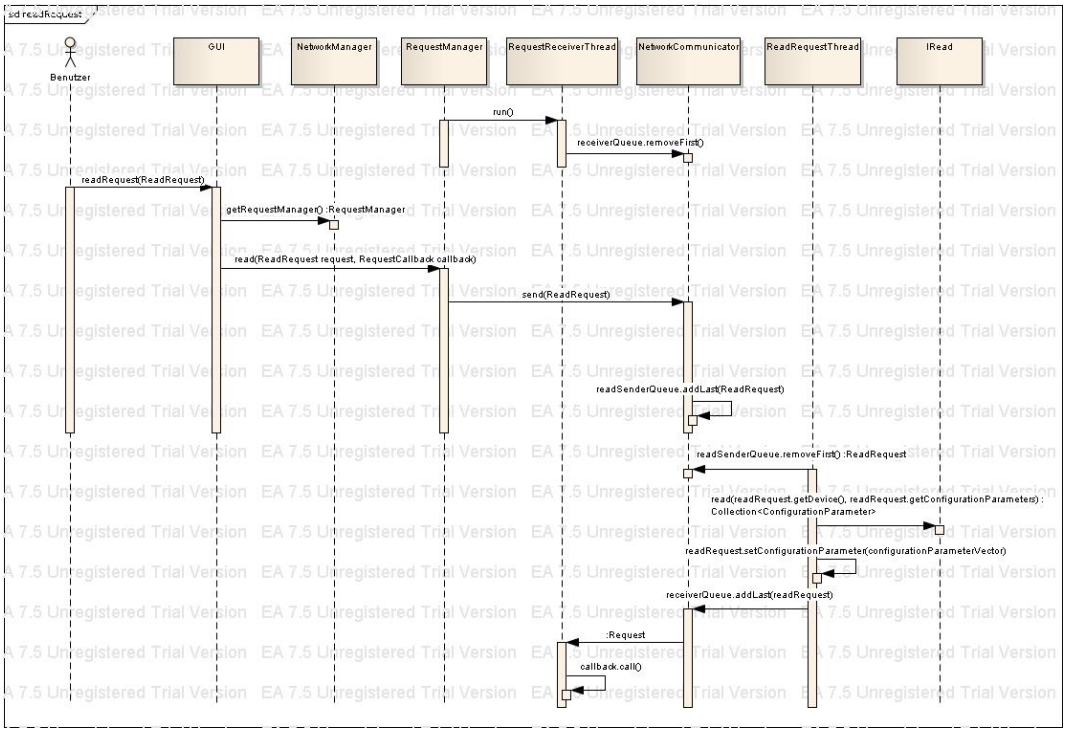


Abbildung 7.13.: Parameter von einem Gerät lesen

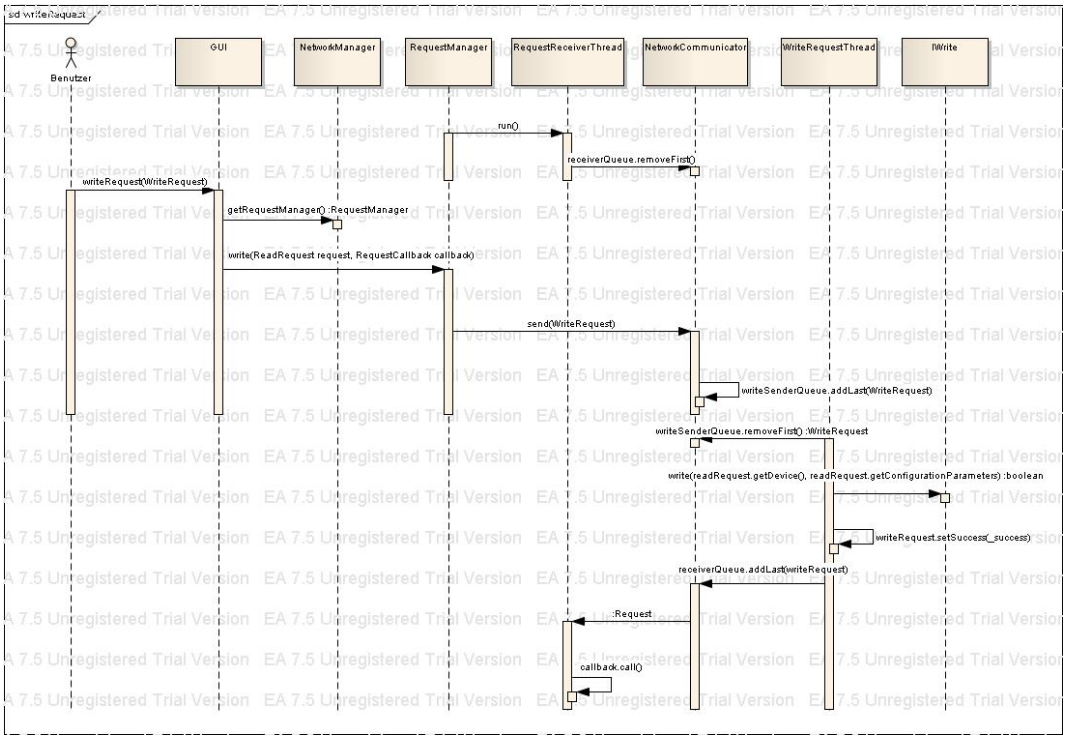


Abbildung 7.14.: Parameter auf einem Gerät schreiben

7.4.4.4. Package persistence

Das persistence Package kapselt die Zugriffe auf die Datenbank. Es bietet Data-Access Objekte an, über die Objekte anhand spezifischer Parameter ausgelesen werden können. Die Data-Access Objekte werden bei den Objekt-Listen des Network-Manager als Observer registriert. Damit wird gewährleistet dass Änderungen an den Daten automatisch persistiert werden.

Diagramme

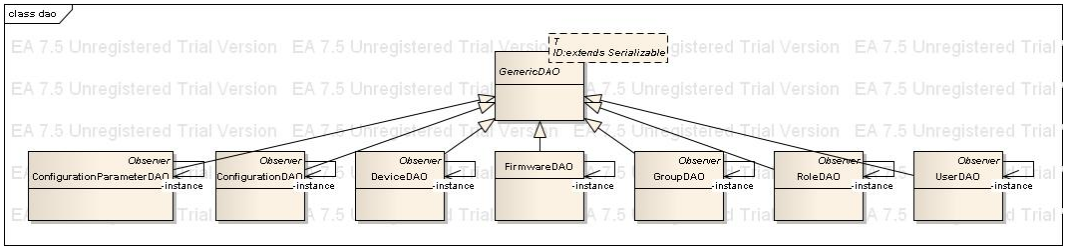


Abbildung 7.15.: Architektur des Persistence Packages

Schnittstellen

Für Objekte die aus der Datenbank gelesen werden müssen, werden Data-Access Objekte (sogenannte DAO's) bereitgestellt. Diese bieten Funktionen an, die Objekte anhand spezifierter Einschränkungen zurückgeben. Es gilt aber die Einschränkung, dass für Objekte wie bsp. ConfigurationParameter keine eigenen DAO's bereitgestellt werden, da diese nur indirekt über ein anderes Objekt ausgelesen werden können.

Operationen

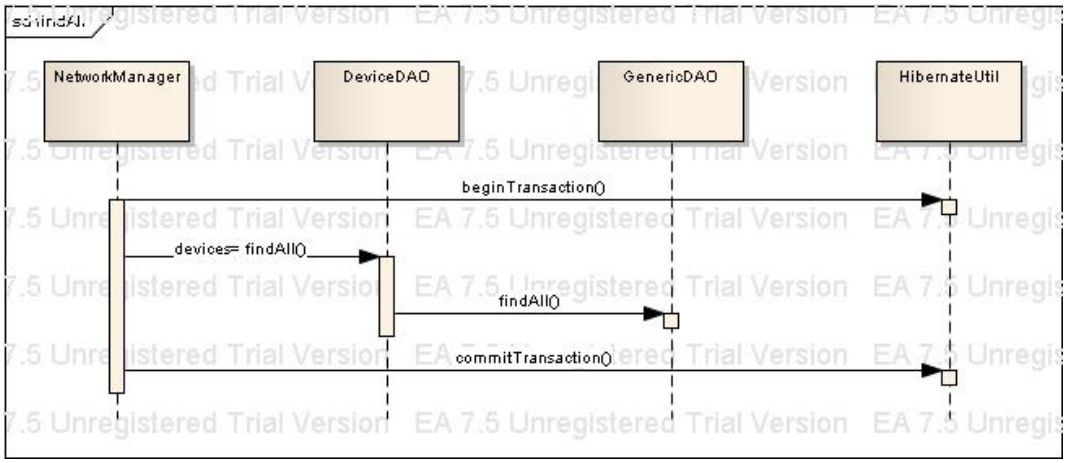


Abbildung 7.16.: Auslesen von Domainklassen (bsp. Device) aus der Datenbank

7.4.4.5. Package util

In diesem Package sind Klassen beinhaltet, welche Hilfsfunktionen anbieten und in keines der anderen Packages gehören.

Diagramme

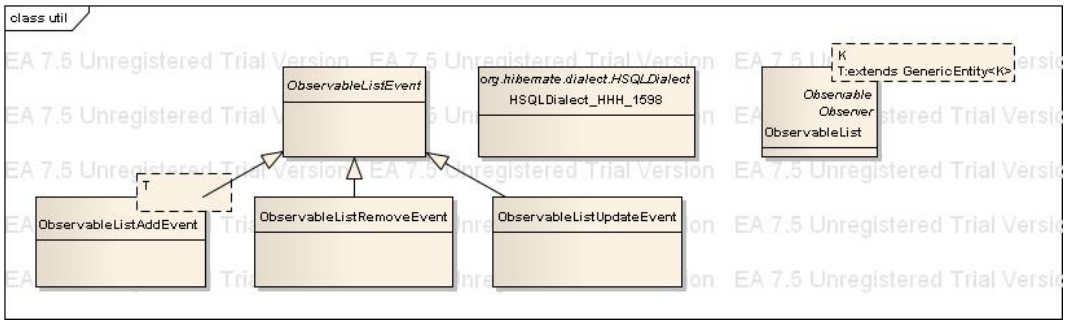


Abbildung 7.17.: Architektur des Util Packages

Schnittstellen

Die ObservableList Klasse wird im Problem Domain Package verwendet. Sie stellt einen Wrapper um eine Liste dar, welcher die Umsetzung des Observer Patterns ermöglicht. Die ObservableList Events werden geworfen um den registrierten Observern anzuzeigen, welche Objekte verändert wurden.

7.4.4.6. Package tests

Das tests Package beinhaltet die automatisierten JUnit Tests, sowie einige Mock-Objekte. Die Mock-Objekte sind abgeleitete Problem Domain Klassen die nur für Tests erstellt wurden. Sie realisieren die Testbarkeit einer Klasse wenn Tests mit den vorhandenen Problem Domain Klasse nicht durchgeführt werden können weil bsp. kein physisches Gerät angeschlossen ist.

Diagramme

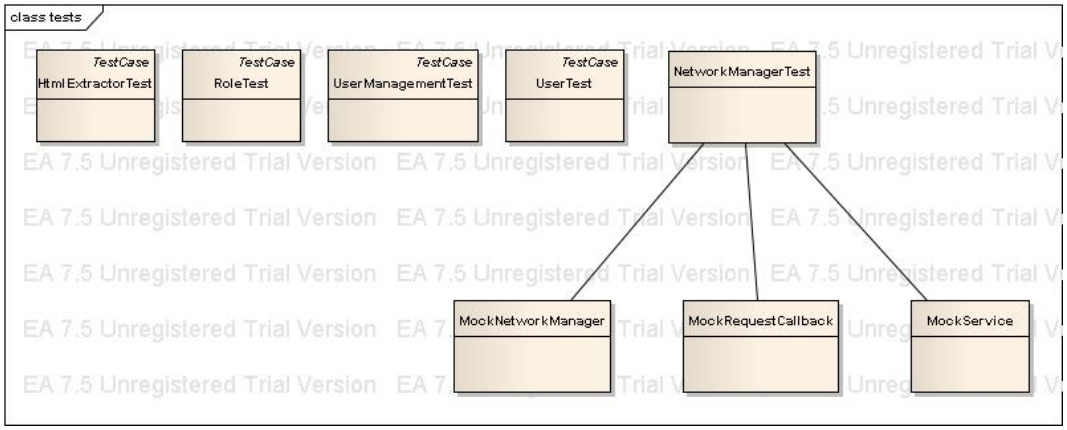


Abbildung 7.18.: Architektur des tests Packages

7.5. Prozesse und Threads

7.5.1. GUI und Controller

Wenn nach einer Benutzeraktion Prozesse im Hintergrund ausgeführt werden, erfolgt die Abarbeitung in einem separaten Hintergrundprozess. In diesem Bereich der Applikation ist neben dem GUI nur maximal ein zusätzlicher Thread gleichzeitig in Abarbeitung.

7.5.2. Netzwerk

Die Schnittstelle zum Netzwerk-Package besteht aus der bereitgestellten RequestManager Klasse oder aus den vier synchronisierten Queues, in welche Requests abgelegt werden können. Die verschiedenen Queues sind:

- DeviceDetectionQueue (5 Threads)
- ReadSenderQueue (5 Threads)
- WriteSenderQueue (5 Threads)
- ReceiverQueue

Jede Queue hat ihren eigenen Thread-Pool. Deren Threads behandeln die auf die Queue geschriebenen Requests. Speziell ist dabei, dass die Threads der ReadSenderQueue mit einer höheren Priorität gestartet werden als die der anderen Queues. Die Idee dahinter ist, dass Leseaufgaben auf die der Benutzer aktiv wartet, so schnell wie möglich abgehandelt werden sollen. Dadurch wird die Wartezeit für den Benutzer minimiert. Die ReceiverQueue hält keinen Thread-Pool, da die Threads der verschiedenen Thread-Pools ihre Ergebnisse darauf schreiben. Wie aus der ReceiverQueue gelesen wird, wird im Kapitel 7.4.4.3 beschrieben.

7.5.3. Problem Domain Klassen

Zugriffe verschiedener Threads auf Problem Domain Klassen werden über synchronized Methoden synchronisiert.

7.6. Datenspeicherung

Die persistente Speicherung der Daten erfolgt über Objekt-Relationales Mapping mittels Hibernate. Hibernate legt die Daten in einer HSQL Datenbank auf dem SL60-MS Host System ab.

7.6.1. Verfolgung des JPA Standards

Die Java Klassen welche in der Datenbank gespeichert werden sollen, werden über Annotations nach dem JPA Standard definiert. Die Kommunikation mit Hibernate erfolgt aber nicht mehr nach dem JPA Standard, sondern mit den spezifischen

Hibernate Hilfsmitteln. Dies aus dem Grund, dass Hibernate den Standard JPA 2.0 noch nicht unterstützt, und sich Datenabfragen mit den spezifischen Hibernate Hilfsmitteln einfacher definieren lassen.

7.7. Grössen und Leistung

Die SL60-MS ist auf eine unbegrenzte Zahl von Daten ausgelegt. Dies bedeutet, dass grundsätzlich beliebig viele Geräte, Gruppen, Links und Benutzer erstellt werden können. Allerdings ist eine visuelle Begrenzung durch das GUI vorhanden. Wenn zu viele Objekte in einer Liste angezeigt werden, verliert der Benutzer die Übersicht. Eine flüssige Bedienung wird erheblich erschwert. Durch diese Begrenzung werden folgende Empfehlungen abgegeben:

- Anzahl der Gruppen, Links: ca. 5 - 20
- Anzahl Geräte (wenn Sie in Gruppen verteilt sind): max. 500
- Anzahl Benutzer: max. 20

8. Schlussfolgerungen und Ausblick

Es wurde eine Applikation entwickelt welche die Anforderungen die in diesem Projekt gestellt wurden, zum grössten Teil erfüllt. Die wichtigsten Anforderungen wurden implementiert und laufen stabil. Dazu konnten zusätzlich bereits tiefer priorisierte Funktionalitäten umgesetzt werden. Wobei aber wegen der zeitlichen Begrenzung einige der spezifizierten Anforderungen aus Kapitel 5.3 ausgelassen werden mussten. Der genaue Entwicklungsstand ist im Kapitel 11.4 festgehalten. Die vorliegende Version kann als Prototyp-Version betrachtet werden. Das Konzept und die Implementation wurden für die Unterstützung mehrerer Firmware Versionen erstellt. Vollständig implementiert und getestet wurde die Applikation aber nur für die Firmware 1.46. Darum kann sie in homogenen Umgebungen mit der Firmware Version 1.46, d.h. in Konfigurationen, bei welchen alle Geräte die Firmware Version 1.46 besitzen, bereits produktiv eingesetzt werden. In inhomogenen Umgebungen wäre die Applikation aber nur für den Teil der Geräte mit der entsprechenden Firmware Version nutzbar. In einer weiteren Phase könnte man die Unterstützung weiterer Firmware Versionen vervollständigen.

Leider muss man mit der vorliegenden Lösung die SL60-Geräte in der "richtigen Reihenfolge" anwählen, damit garantiert werden kann, dass alle Geräte konfiguriert werden können. Durch eine geschicktere Abarbeitungsstrategie könnte man den Nachteil wettmachen, dass ein zu konfigurierendes Gerät durch ein gleichzeitig zu konfigurierendes Gerät unerreichbar wird. In einer Weiterentwicklung könnte man den Fokus auf die Performance und die automatische Fehlerbehebung bei Überlastung des Netzwerks legen. Dies könnte nach den Patterns im Buch [Han07] erfolgen, für deren Umsetzung uns die Zeit leider nicht mehr reichte.

Als Kritikpunkt könnte die zu grosse Fokussierung auf Sekundärfunktionalitäten angebracht werden. In einem nächsten Projekt müsste man mehr Wert auf die Primärfunktionalitäten legen, bevor mit anderen Funktionalitäten begonnen wird.

Das erzielte Ergebnis kann von Huber+Suhner einerseits als Demonstrator verwendete werden und andererseits für die Festlegung von detaillierten Anforderungen an ein Endprodukt. Zudem zeigt dieses Endprodukt Schwachstellen der Geräte auf.

Teil III.

Projekt Management

9. Projektplan

9.1. Einleitung

9.1.1. Zweck

Dieser Projektplan ist das zentrale Dokument für das Projektmanagement im Projekt SL60-MS und soll die vorhandenen Ressourcen koordinieren, Aktionen festlegen und den Projektverlauf dokumentieren.

9.2. Übersicht

Die folgenden Kapitel beschreiben detailliert die Projektorganisation, die Management Abläufe sowie das Risikomanagement. Desweiteren wird das Projekt SL60-MS in diesem Dokument in Arbeitspakete aufgeteilt.

9.3. Projektübersicht

9.3.1. Zweck und Ziel

In diesem System sollen Benutzer der SL60-MS (typischerweise ein Netzwerkmanager) die Möglichkeit haben, mehrere SL60-Geräte auf ein Mal zu konfigurieren und sich so einen grossen Arbeitsaufwand zu ersparen. Das Kapitel 5.2 gibt einen genaueren Überblick.

9.3.2. Annahmen und Einschränkungen

Der Aufwand für das Softwareprojekt wird pro Teammitglied auf zirka 26 Stunden pro Woche geschätzt. Da die Teammitglieder zusätzlich mit anderen Modulen belastet sind, gehen wir davon aus dass die zeitlichen Ressourcen pro Teammitglied auf maximal 34 Stunden pro Woche begrenzt sind. Daher wurde versucht genügend Reserven in den Zeitplan einzubauen.

9.4. Projektorganisation

Die Verantwortlichkeiten sollen entsprechend den Fähigkeiten und Motivationen der einzelnen Teammitglieder aufgeteilt werden. Es sollen alle Teammitglieder an der Dokumenation sowie an der Programmierung beteiligt sein, wobei jeder für einen Teil die Verantwortlichkeit übernimmt.

9.4.1. Organisationsstruktur

Name	Kürzel	Verantwortlichkeiten
Lukas Wilhelm	lw / lwilhelm	Zeitplan, Design, Datenhaltung, GUI
Damien Vouillamoz	dv / dvouilla	Anforderungsspezifikation, Domainanalyse, Netzwerk

9.4.2. Externe Schnittstellen

Name	Firma / Organisation	Kürzel	Email
Prof. Dr. Peter Heinzmann	HSR, Cnlab	hei / pheinzma	pheinzma@hsr.ch
Stephané Racine	Huber+Suhner	sra / sracine	stephane.racine@hubersuhner.com
Prof. Hans Rudin	HSR	srudin	hrudin@hsr.ch

9.5. Management Abläufe

9.5.1. Projekt Kostenvoranschlag

Der Zeitrahmen für die Umsetzung des Projekts beträgt 14 Wochen. Es ist kaum realistisch, dass das Projekt bereits wesentlich früher abgeschlossen werden kann. Der Grund dafür ist der zu erwartende wöchentliche Aufwand. Der Zeitplan geht von einer möglichst realistischen Annahme über den zu erwartenden Aufwand aus. Es wurde im Projektplan darauf geachtet, dass mit riskanten Arbeitspaketen früh genug begonnen wird. Dadurch sollen davon abhängige Pakete keine Verzögerungen erfahren. Es wurden einige Funktionen eingeplant, welche im Notfall weggelassen werden können ohne dass dadurch die Grundanforderungen an die Applikation beeinträchtigt werden. Verzögerungen sollen frühzeitig erkannt werden.

9.5.2. Projektplan

Siehe Dokument “Zeitplan.xls”.

9.5.2.1. Zeitplan

9.5.2.2. Iterationsplanung / Meilensteine

Iteration	von	bis	Beschreibung
Inception	14.09.2009	27.09.2009	Anforderungsanalyse aus Kundensicht
Elaboration	28.09.2009	11.10.2009	Einarbeitung in Problemstellung, Erstellen eines Proof Of Concept
Construction 1	12.10.2009	01.11.2009	Erreichen von MS3. Doku und Implementation wichtigster Priorität A Anforderungen
Construction 2	02.11.2009	22.11.2009	Erreichen von MS4. Doku und Implementation aller Priorität A Anforderungen.
Construction 3	23.11.2009	29.11.2009	Sofern Zeit reicht, Doku und Implementation Auswahl Priorität B, C Anforderungen.
Construction 4	30.11.2009	06.12.2009	Software auslieferungsfähig und ausgiebiges Testen
Transition	07.12.2009	18.12.2009	Reservezeit sowie CD mit SW und Dokumentation brennen

Milestone	Datum	Beschreibung
MS1 - Projektplan abgeschlossen	28.09.2009	Projektplan abgeschlossen (siehe Kapitel 10.1)
MS2 - Design & Proof Of Concept	11.10.2009	Design und Proof-of-concept abgeschlossen und mit Betreuer bzw. Fachleuten besprochen (siehe Kapitel 10.2)
MS3 - Alpha Version - Prototyp	01.11.2009	Anforderungs-, Designreview, Prototyp (wichtigste Priorität A Funktionalitäten fertig) für Präsentation Zwischenstand; Präsentation Koreferent
MS4 - Beta Version - Prototyp erweitert	29.11.2009	alle Priorität A-, sowie Auswahl B,C Funktionalitäten implementiert
MS5 - Release Candidate 1 Version (RC1) - Vollversion	06.12.2009	SW Auslieferungsfähig, getestet auf Fremdrechner
MS6 - Final Version - Auslieferungsfähige Software inkl. Dokumentation fertig	18.12.2009	Dokumentation fertig (inkl. Korrektur, Reviews und Ausdruck) und CD mit SW und Dokumentation brennen

9.5.2.3. Besprechungen

Wann	Wer	Ziel / Grund
22.09.2009	pheinzma, sracine, dvouilla, lwilhelm	Abklärung Anforderungen
03.11.2009	srudin, pheinzma, dvouilla, lwilhelm	Zwischenpräsentation des Prototypen
17.11.2009	sracine, pheinzma, dvouilla, lwilhelm	Zwischenpräsentation des Entwicklungsstandes
24.11.2009	sracine, dvouilla, lwilhelm	Besprechung GUI

Ansonsten finden wöchentliche Meetings zwischen den Entwicklern und Peter Heinzmann statt.

9.5.2.4. Releases

Version	Features	Typ	Datum	Iteration
0.1	Anforderungen wichtigste Priorität A	Alpha	01.11.2009	Ende Construction 1
0.2	Anforderungen alle Priorität A, evtl. B	Beta	29.11.2009	Ende Construction 3
0.3	fertige Version	RC1	06.12.2009	Ende Construction 4
1	nach erneutem Bugfixing	Final	18.12.2009	Ende Transition

9.6. Risiko Management

ID	Risiko	Massnahme	WK	Zeit	Risiko (WK * Zeit)
R01	Die Kommunikation mit den Geräten ist nicht wie geplant über eine SNMP API möglich	Frühzeitiges Testen innerhalb eines Proof-Of-Concept	0.2	20h	4h
R02	Datenverlust	SVN verwenden	0.05	20h	1 h
R03	Die gestellten Anforderungen sprengen den zeitlichen Rahmen des Projekts	Nur einige der Anforderungen der B Prioritäten realisieren und für ein Weiterentwicklung sauber dokumentieren	0.5	40h	20h
R04	Die erstellte Java-Applikation oder importierten Libraries laufen aufgrund unbekannten Faktoren nicht auf den vorgesehenen Plattformen.	Bei Beginn der Phase Construction 1: Tests durchführen	0.3	15h	5h
R05	Die Datenhaltung mit purem JDBC erweist sich als untauglich.	Suchen einer Alternative, Einarbeitung in neue Technologie.	0.5	24h	12h
R06	Absenzen bringen den Zeitplan durcheinander und verzögern die Entwicklung	Während Absenzen trotzdem Minimalaufwand an Arbeit leisten	0.5	16h	8h
Gesamtrisiko des Projekts					**50h**

9.6.1. Eingeplante Reserven

Falls nötig können die wöchentlichen Arbeitsstunden von 26 auf 34 Stunden erhöht werden.

9.7. Arbeitspakete

Siehe Dokument “Zeitplan.xls”.

9.8. Infrastruktur

Da die Entwickler bereits mit der Java Entwicklungsumgebung Eclipse vertraut sind, wird diese als Entwicklungsumgebung gewählt. Für die Erstellung des User Interfaces wird auf die Hilfe eines GUI Builders zurückgegriffen. Hier wird der GUI Builder der Open Source Software Netbeans gewählt. Netbeans wurde dem Eclipse Plugin Jigloo vorgezogen, weil der darin enthaltene GUI Builder weniger aufwändig zu bedienen ist. Um das Projekt mit dem Subversion-Server zu synchronisieren, wird das Eclipse-Plugin Subclipse verwendet. Der Zeitplan wird mit Excel aus dem Microsoft Office Paket erstellt. Die Dokumentation des Source Codes wird mit JavaDoc erstellt.

9.8.1. Tools

Applikation	**Version**	**Beschreibung**
Eclipse	Galileo (3.5.1)	Java Entwicklungsumgebung
Netbeans	6.7.1	GUI Builder
Tortoise SVN	1.6.6	Versionierung
Microsoft Office Excel	07	Tabellenkalkulation
Javadoc	1.5	Source Dokumentation

9.8.2. Zeiterfassung

Die Arbeitszeit wird von den einzelnen Mitarbeitern direkt in den Zeitplan eingetragen.

9.9. Qualitätsmassnahmen

9.9.1. Coding Guidelines

Um den Source-Code konsistent zu halten wird nach den folgenden Richtlinien entwickelt:

- Klassenkommentar (JavaDoc)
  Klassen werden im folgenden Format dokumentiert:

```
/**
* This class represents a Device.
*
* @author Lukas Wilhelm
*/
public class Device
```

- Methodenkommentar (JavaDoc)
  Methoden werden im folgenden Format dokumentiert:

```
/**
* Reads the passed field from the passed script on the passed ip address
* (Single Get/Read)
*
* @param _ip IP address of the device on which the field has to be read
```

```
* @param _readScriptPath Script Path where the field can be found on the device
* @param _field Field containing the requested information
* @return Value of the requested field in the passed script on the device with the passed ip address
* @throws IOException Throwed when errors occured during the read process
*/
public String getFieldValue(String _ip, String _readScriptPath, String _field) throws IOException {
```

- Methodennamen
  Methodennamen sollen aussagekräftig sein und beschreiben was die Methode macht und nicht wie oder warum diese einen Arbeitsprozess durchführt. Desweiteren sollen sollen Informationen zu Rückgabewerten oder Parametern nicht im Methodennamen integriert sein. (Bsp: getFieldValueString(..) oder getFieldValueByIpReadScriptPathField(...)).

- Methodenparameter
  Methodenparametern soll immer ein "_" (Underscore) vorangestellt werden. Somit ist im Code sofort ersichtlich woher eine Variable stammt.

- Kommentare
  Grundsätzlich sollten Kommentare durch gut leserlichen und verständlichen Code ersetzt werden. Übersichtshalber können Codeabschnitte kommentiert werden. Es ist aber darauf zu achten, dass der Kommentar erklärt "Was" gemacht wird und nicht "Wie".

9.9.2. Teststrategie

Alle erstellten Java-Klassen werden systematisch vom verantwortlichen Entwickler getestet und dokumentiert. Für jede wichtige Komponente - vorwiegend Klassen aus der Problem Domain - wird ausserdem eine Unit-Test Klasse geschrieben, damit die anderen Entwickler die Integrität ihrer neuen Module im gesamten Projekt testen können. Unit-Tests werden, soweit möglich, gemäss den Requirements erstellt und abgearbeitet. Am Ende jeder Iteration wird das GUI auf Einhaltung der Anforderungsspezifikation und der spezifizierten Use Cases getestet.
Für die Klassen welche die Netzwerkkommunikation übernehmen, werden Mock-Objekte bereitgestellt. Diese simulieren das Lesen und Schreiben von Informationen auf physische Geräte.

9.9.2.1. Reviews

Zeit- und sicherheitskritische Codeblöcke werden bei der Planung und Programmierung im Zweier-Team besprochen und bei Änderungen an den Entwickler zur Überarbeitung zurückgegeben.

10. Meilensteine

10.1. MS1 - Projektplan abgeschlossen

Der Meilenstein 1 wurde anfangs der dritten Projektwoche erreicht. Das Projekt begann während einer Sitzung zwischen Huber+Suhner, dem Betreuer sowie den Entwicklern. Während dieser Sitzung wurden von Entwicklerseite Vorschläge für funktionelle Anforderungen gemacht und besprochen. Danach wurden die Anforderungen gemeinsam besprochen. Die wesentlichen Korrekturen und Erweiterungen seitens des Betreuers und Huber+Suhner waren vorgenommene Einschränkungen der vorgeschlagenen Funktionalitäten. Dies geschah aus Gründen der technischen Machbarkeit oder einer durchgeführten Aufwandsschätzung. In einem zweiten Schritt wurden diese aufgenommenen Anforderungen priorisiert und nochmals dem Industriepartner vorgelegt. Den Meilenstein schlossen Prioritätsanpassungen und die Erstellung eines Zeitplans ab.

10.2. MS2 - Proof Of Concept

10.2.1. Überblick

Zu Testzwecken wurde eine erste kleine Applikation entwickelt. Sie hat den Zweck die Möglichkeit der Realisierung des geplanten Konzepts aufzuzeigen. Geplant war die Kommunikation vollständig über SNMP durchzuführen. Da die SL60-Geräte aber nur das Lesen von SNMP Parametern gestatten, muss der Schreibvorgang über das HTTP-Protokoll durchgeführt werden. Folglich realisiert die Applikation das simple Schreiben und Lesen eines Wertes auf einem Gerät. Das Schreiben wird über HTTP, das Lesen über SNMP v2c realisiert.

10.2.2. Screenshots

Abbildung 10.1.: Screenshots des Proof Of Concept

Zuerst wird der Wert des Systemnamens auf das Gerät geschrieben. Danach wird die erfolgreiche Übermittlung der Daten, über das Web-Interface, überprüft. Zum Schluss wird der Namen wieder vom Gerät gelesen.

10.2.3. Technische Schwierigkeiten

Das Schreiben über HTTP benötigt bislang noch zwei Verbindungen für ein Gerät. Die erste Verbindung um die Authentisierung durchzuführen. Die zweite Verbindung um dann den Wert zu übertragen.

10.3. MS3 - Alpha Version

10.3.1. Überblick

Die Anforderungsspezifikation wurde abgeschlossen sowie eine Domainanalyse durchgeführt. Nach der anschliessend durchgeführten Planung der Priorität A Funktionen und des Erstellens eines detaillierten Klassendiagramms, wurde mit der Implementation begonnen. Geplant war ursprünglich die vollständige Implementation der Priorität A Funktionalitäten. Dieses Ziel konnte wegen des engen zeitlichen Rahmens nicht erreicht werden. Der Stand der Entwicklung bis zum Zeitpunkt des Meilenstein 3 ist im Kapitel 11.1 genauer festgehalten.

10.4. MS4 - Beta Version

10.4.1. Überblick

Das Design der Applikation wurde weiter verfeinert. Daneben wurde die Implementation fortgesetzt. Wegen der leichten Verzögerung der Entwicklung wurde eine Neufokussierung auf die Kernfunktionalitäten vorgenommen. Mehr zum Stand der Entwicklung beim Meilenstein 4 ist im Kapitel 11.2 festgehalten.

10.5. MS5 - Release Candidate 1 Version

10.5.1. Überblick

Neben der ständigen Anpassung der Dokumentation wurde einiges an Code refactored. Der Stand der Entwicklung bis zum Zeitpunkt des Meilenstein 5 ist im Kapitel 11.3 genauer festgehalten.

10.6. MS6 - Final Version

10.6.1. Überblick

Neben der Endversion wurde eine ausführliche Projektdokumentation fertiggestellt. Der Stand der Entwicklung bis zum Zeitpunkt des Meilenstein 6 ist im Kapitel 11.4 genauer festgehalten. Die Ergebnisse aus diesem Projekt werden im Kapitel 8 reflektiert.

11. System-Testdokumentation

11.1. Release 1 - Version 0.1

11.1.1. Release Überblick

Von - Bis	Woche 5 - 7 (12.10 - 01.11.2009)
Zeitdauer	3 Wochen

11.1.2. Differenzen zur Planung

In der ersten Version der SL60-MS sollten die Priorität A Funktionen vollständig implementiert werden. Dieses Ziel konnte nur teilweise erreicht werden. Der Grund liegt darin dass die Anforderungsspezifikation später als geplant fertiggestellt wurde, und somit erst später mit der Domain-Analyse begonnen werden konnte. Für die Designanalyse sowie die Implementation blieben somit nur noch 1.5 Wochen der Construction Phase 1 übrig. Geplant war alle Anforderungen mit Priorität A fertig zu implementieren. Der tatsächliche Entwicklungsstand ist in der folgenden Tabelle festgehalten. Es konnten noch keine Anforderungen vollständig implementiert werden.

11.1.3. Tests nach funktionalen Anforderungen

Priorität	Anforderung	Status	Unschönheit
A	FR-01 Authentisierung	in Entwicklung	Benutzerwechsel funktioniert nicht
A	FR-10 Konfiguration durchführen	in Entwicklung	Abhängigkeiten nicht schön dargestellt
A	FR-12 Konfigurations-History verwalten	noch nicht begonnen	
A	FR-05 listenbasierte Geräteerkennung	in Entwicklung	
A	FR-18 Geräte gruppieren	in Entwicklung	Nach Aktualisierungen wird Baum fehlerhaft dargestellt.
A	FR-06 Darstellung der Geräte	in Entwicklung	benutzte GUI Komponente nicht optimal.
A	FR-07 Detailansicht eines Geräts	noch nicht begonnen	

Tabelle 11.1.: Test zum Zeitpunkt von Release 1, 01.11.09

11.1.4. weitere Tests

Das Projekt wurde gemäss den Anforderungen als ausführbare Jar-Datei exportiert und auf einem Windows System (XP) und einer Linux Distribution (Fedora 10) getestet. Dabei wurde festgestellt dass keinerlei Fehler auftraten, und die SL60-MS auf beiden Systemen fehlerfrei lief. Es ist anzumerken dass die Jar-Datei auf dem Linux System nicht per Doppelklick gestartet werden konnte, sondern nur über die Konsole.

11.1.5. Technische Schwierigkeiten

Es musste eine geeignete Variante zur persistenten Datenhaltung gefunden werden. Der Entscheid fiel auf Objekt-Relationales Mapping mit Hibernate. Die Einarbeitung in Hibernate bedeutete einen Mehraufwand gegenüber der Planung, da zu dieser Zeit die Implementation der Datenhaltung mittels der den Entwicklern bereits bekannten SQLite Datenbank geplant wurde.

11.2. Release 2 - Version 0.2

11.2.1. Release Überblick

Von - Bis Woche 8 - 11 (02.11 - 29.11.2009)
Zeitdauer 4 Wochen

11.2.2. Differenzen zur Planung

Das ursprüngliche Ziel war die Fertigstellung aller B-Priorität Funktionen. Da die Entwicklung für den Release 1 schon verzögert fortgeschritten ist, war dieses Ziel nicht mehr realistisch. Während der Sitzung vom 17.11.09 wurde darum eine Fokussierung auf die A-Priorität Funktionalitäten festgelegt. B- und C-Priorität Funktionalitäten werden nur teilweise umgesetzt. Aufgrund Kundenwunschs werden die B-Prioritäten "FR-08 Darstellung der Geräte auf einer Landkarte" und "FR-13 Netzwerkstatistiken anzeigen" bevorzugt behandelt. Die restlichen Funktionalitäten werden je nach Aufwand und verbleibender Zeit umgesetzt. Der Entwicklungsstand ist in der folgenden Tabelle festgehalten.

11.2.3. Tests nach funktionalen Anforderungen

Priorität	Anforderung	Status	Unschönheiten
A	FR-01 Authentisierung	fertig gestellt	
A	FR-10 Konfiguration durchführen	in Entwicklung	Beim ersten Klick auf Konfigurationsparameter wird Feld nicht ausgewählt
A	FR-12 Konfigurations-History verwalten	noch nicht begonnen	
A	FR-05 listenbasierte Geräteerkennung	fertig gestellt	
A	FR-18 Geräte gruppieren	fertig gestellt	
A	FR-06 Darstellung der Geräte	fertig gestellt	
A	FR-07 Detailansicht eines Geräts	fertig gestellt	
B	FR-02 Benutzer verwalten inkl. Rollen zuweisen	fertig gestellt	
B	FR-19 Import/Export der Konfiguration	noch nicht begonnen	
B	FR-11 vordefinierte Templates verwalten und anwenden	noch nicht begonnen	
B	FR-08 Darstellung der Geräte auf einer Landkarte	noch nicht begonnen	
B	FR-16 Systemlog eines Geräts anzeigen	noch nicht begonnen	
B	FR-13 Netzwerkstatistiken anzeigen	noch nicht begonnen	
B	FR-22 Hilfe zu den Konfigurationsparametern anbieten	fertig gestellt	

Tabelle 11.2.: Test zum Zeitpunkt von Release 2, 29.11.09

11.3. Release 3 - Version 0.3

11.3.1. Release Überblick

Von - Bis Woche 12 (30.11 - 06.12.2009)
Zeitdauer 1 Woche

11.3.2. Differenzen zur Planung

In diesem Release wurde Programm-Code refactored sowie zusätzliche Funktionalitäten implementiert. Der Fortschritt war etwas langsamer als geplant, so dass die Entwicklung noch nicht abgeschlossen werden konnte. Einige wenige Funktionalitäten müssen für die Endversion noch implementiert werden. Der Entwicklungsstand ist in der folgenden Tabelle festgehalten.

11.3.3. Tests nach funktionalen Anforderungen

Priorität	Anforderung	Status	Unschönheiten
A	FR-01 Authentisierung	fertig gestellt	
A	FR-10 Konfiguration durchführen	fertig gestellt	
A	FR-12 Konfigurations-History verwalten	noch nicht begonnen	
A	FR-05 listenbasierte Geräteerkennung	fertig gestellt	
A	FR-18 Geräte gruppieren	fertig gestellt	
A	FR-06 Darstellung der Geräte	fertig gestellt	
A	FR-07 Detailansicht eines Geräts	fertig gestellt	
B	FR-02 Benutzer verwalten inkl. Rollen zuweisen	fertig gestellt	
B	FR-19 Import/Export der Konfiguration	noch nicht begonnen	
B	FR-11 vordefinierte Templates verwalten und anwenden	noch nicht begonnen	
B	FR-08 Darstellung der Geräte auf einer Landkarte	in Entwicklung	Pfeile nicht eingefärbt, Gerätedaten erst Beispieldaten
B	FR-16 Systemlog eines Geräts anzeigen	noch nicht begonnen	
B	FR-13 Netzwerkstatistiken anzeigen	in Entwicklung	Skalierung nicht angepasst
B	FR-22 Hilfe zu den Konfigurationsparametern anbieten	fertig gestellt	
C	FR-14 Geräte- und Verbindungsstatistiken exportieren und drucken	fertig gestellt	

Tabelle 11.3.: Test zum Zeitpunkt von Release 3, 06.12.09

11.4. Release 4 - Version 1

11.4.1. Release Überblick

Von - Bis Woche 13 - 14 (07.12 - 18.12.2009)
Zeitdauer 2 Wochen

11.4.2. Differenzen zur Planung

Es konnten alle Priorität A Funktionalitäten fertig implementiert werden. Ausserdem konnten auch die aus Kundensicht wichtigsten B und C Prioritäten implementiert werden. In der folgenden Tabelle ist der Entwicklungsstand bei Abschluss des Projekts zu sehen.

11.4.3. Tests nach funktionalen Anforderungen

Priorität	Anforderung	Status	Unschönheiten
A	FR-01 Authentisierung	fertig gestellt	
A	FR-10 Konfiguration durchführen	fertig gestellt	
A	FR-12 Konfigurations-History verwalten	fertig gestellt	Es kann nur immer genau zum letzten Schritt zurückgesprungen werden.
A	FR-05 listenbasierte Geräteerkennung	fertig gestellt	
A	FR-18 Geräte gruppieren	fertig gestellt	
A	FR-06 Darstellung der Geräte	fertig gestellt	
A	FR-07 Detailansicht eines Geräts	fertig gestellt	
B	FR-02 Benutzer verwalten inkl. Rollen zuweisen	fertig gestellt	
B	FR-19 Import/Export der Konfiguration	nicht begonnen	
B	FR-11 vordefinierte Templates verwalten und anwenden	nicht begonnen	
B	FR-08 Darstellung der Geräte auf einer Landkarte	fertig gestellt	
B	FR-16 Systemlog eines Geräts anzeigen	fertig gestellt	
B	FR-13 Netzwerkstatistiken anzeigen	nicht begonnen	
B	FR-22 Hilfe zu den Konfigurationsparametern anbieten	fertig gestellt	
C	FR-14 Geräte- und Verbindungsstatistiken exportieren und drucken	fertig gestellt	
C	FR-15 logische Fehlkonfiguration erkennen	fertig gestellt	
C	FR-09 schematische Darstellung der Geräte und Verbindungen	fertig gestellt	

Teil IV.

Anhang

A. Externe Dokumente des Berichts

Dokument	**Dateipfad**
Zeitplan	doc/Zeitplan.xls
Sitzungsprotokolle	doc/protokolle/
Benutzeranleitung der Applikation	doc/Benutzeranleitung.pdf

B. Kommentare zum Zeitplan

Der Zeitplan wurde zu Beginn des Projekts aufgestellt. Danach wurden während dem Projektverlauf die Arbeitszeiten eingetragen und wenn nötig Korrekturen an der Planung gemacht. Das Projekt verlief grob in 3 Phasen. In der ersten Phase war die Arbeit sehr theoretisch, während in der zweiten Phase fast nur noch implementiert wurde. In der Schlussphase wurde dann die Dokumentation und die Implementation fertig gestellt. Die tatsächliche Arbeitszeit liegt bei beiden Entwicklern über der ursprünglich geplanten Soll-Zeit. In der Entwicklungsphase wurde bemerkt, dass vor allem die Implementation des GUI's einiges mehr an Zeit kostet als geplant. Mit dem Kunden und dem Betreuer wurde während der Sitzung vom 24.11.2009 abgesprochen, dass diese Mehrarbeit bewilligt wird.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Management-Software für Richtfunk-Systeme


Lukas Wilhelm


Damien Vouillamoz

Bachelorarbeit Herbstsemester 09/10
Internet-Technologien und -Anwendungen

Betreuer: Prof. Dr. Peter Heinzmann
Experte: Dr. Th. Siegenthaler
Projektpartner: Huber+Suhner AG, Herisau AR

Ausgangslage

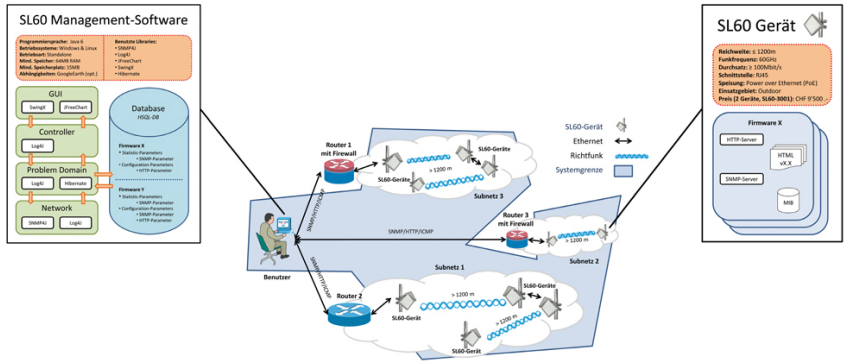
- Huber+Suhner wünscht Management Anwendung für ihre neuen SL60-Geräte
- Geräte sollen nicht nur einzeln und über Web Interface konfigurierbar sein

Herausforderungen

- Mehrere Firmware Versionen mit unterschiedlichen Parametern
- Umgang mit der hohen Fehleranfälligkeit der Geräte
- Entwurf einer übersichtlichen Benutzerschnittstelle trotz zahlreicher Funktionalitäten
- Parallele und priorisierbare Kommunikation mit den Geräten

Resultat

- Einzel- und Massenkonfiguration von Geräten (inkl. Firmware Upload) und Step-Back Funktion für bis zu 1000 Geräte
- Gerätestatistiken in einem Diagramm darstellen
- Gruppieren von Geräten in Gruppen und Links
- Links und dazugehörige Geräte auf einer Karte (GoogleEarth) anzeigen lassen
- User-Management und Authentisierung der Benutzer



D. Persönliche Berichte

D.1. Damien Vouillamoz

Schon bei der Bewerbung um diese Arbeit wusste ich, dass es sich hier um ein sehr interessantes Projekt handelt. In diesem Projekt sah ich die Möglichkeit den in den letzten drei Jahren erlernten Stoff in der Praxis anzuwenden. Dazu kam, dass dieses Projekt meine bevorzugten Themengebiete (Internettechnologien und -Anwendungen) beinhaltete. Die Motivation war von Anfang an hoch. Als wir endlich nach sechs Wochen mit der Implementation beginnen konnten war die Motivation allerdings schon ein bisschen angeschlagen. Von diesem Zustand erholte ich mich aber ziemlich schnell mit den ersten Code-Zeilen und Klassen. Der zweite Teil der Arbeit verlief ziemlich stressig und aufwändig. Gewillt nicht nur das Projekt voranzutreiben, sondern ein stabiles und nutzbares Produkt zu entwickeln, lasteten bzw. liessen wir uns zusätzlich Wünsche vom Betreuer und vom Kunden auflasten. Doch so sieht nun mal die Praxis aus. So kam es, dass wir am Schluss in Zeitdruck gerieten. Rückblickend bin ich froh, dass wir zu Beginn sechs Wochen damit verbracht haben die Dokumentation zu erstellen. Es hat uns sicherlich viel Arbeit und Ärger erspart. Trotzdem hätten wir dort genauer spezifizieren sollen was, wer und wie auslöst und verarbeitet. Zudem hätten wir im zweiten Teil des Projektes verstärkt auf die Hauptfunktionalitäten eingehen sollen. Auf jeden Fall freue ich mich dem Kunden das Resultat unserer Arbeit auf Weihnachten schenken zu können.

D.2. Lukas Wilhelm

Die Arbeit am SL60-MS Projekt war sehr spannend und herausfordernd. Es galt die Anforderungen des Kundens, sowie des Betreuers zu erfüllen. Es gelang uns nicht immer alles gleichzeitig unter einen Hut zu bringen. Da neben dem funktionierenden Produkt, mit möglichst vielen realisierten Features, noch eine saubere Software-Dokumentation zu erstellen war. Beim Zeitaufwand für die Implementation hatten wir uns verschätzt, darum mussten zum Schluss des Projekts einige Anforderungen gestrichen werden. Trotzdem wurde es am Schluss knapp, das Projekt noch termingerecht abzuschliessen. Für die Zukunft merke ich mir, dass ich frühzeitig über die Bücher gehen sollte und den Zeitplan laufend, entsprechend der “Implementationsgeschwindigkeit”, überarbeite. Ebenfalls würde ich dem Kunden frühzeitig mitteilen, dass gewisse Anforderungen nicht in dieser Zeit realisierbar sind. Falls zu viele Anforderungen in geringer Zeit realisiert werden müssen, besteht die Gefahr dass die Qualität des Codes darunter leidet. Nun zum Schluss des Projekts bin ich überzeugt

dass wir eine saubere Projektarbeit vorweisen können. Ausserdem freut mich dass wir für den Kunden eine gute Lösung entwickeln konnten, welche er unter gewissen Voraussetzungen bereits produktiv nutzen kann.

E. Metriken

Eine Auswertung der Code Metriken hat die folgenden Zahlen zur SL60-MS ergeben.

Anzahl Codezeilen	14425
Klassen	175

F. HSR Vereinbarung Rechte

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

Vereinbarung über Urheber- und Nutzungsrechte

1. Gegenstand der Vereinbarung

Mit dieser Vereinbarung werden die Rechte über die Verwendung und die Weiterentwicklung der Ergebnisse der Bachelorarbeit „Management-Software für Richtfunk-Systeme“ von Lukas Wilhelm und Damien Vouillamoz unter der Betreuung von Dr. Prof. Peter Heinzmann geregelt.

2. Urheberrecht

Die Urheberrechte stehen der Studentin / dem Student zu.

3. Verwendung

Die Ergebnisse der Arbeit dürfen sowohl von der Studentin / dem Student, von der HSR wie von der Huber+Suhner AG nach Abschluss der Arbeit verwendet und weiter entwickelt werden

Rapperswil, den..... .....
Die Studentin/der Student

Rapperswil, den..... .....
Der Betreuer / die Betreuerin der Bachelorarbeit

Rapperswil, den..... .....
Der Studiengangleiter / die Studiengangleiterin

Vereinbarung Schutzrechte

Ohne anderslautende Vereinbarungen stehen die Schutzrechte und das Know-how an der Studienarbeit oder Bachelorarbeit (nachfolgend ‚Arbeit' genannt) und an der in diesem Rahmen geschaffenen Güter, wie Software, sowohl dem Rechtsträger der HSR Hochschule für Technik, dem für die Arbeit verantwortlichen Professoren sowie dem Verfasser der Arbeit resp. Entwickler der in diesem Rahmen geschaffenen Güter, wie Software, zu.

Die genannten Parteien übertragen sich gegenseitig nicht exklusiv, jedoch unentgeltlich, weltweit, sachlich und zeitlich unbeschränkt die jeweiligen Schutzrechte und das Know-how an der Arbeit und an der in diesem Rahmen geschaffenen Güter, wie Software, einschliesslich dem Recht zur Weiterübertragung, ab. Entsprechend steht es jeder Partei zu, sämtliche Schutzrechte an der Arbeit resp. an der in diesem Rahmen geschaffenen Güter, wie Software, beliebig weltweit, zeitlich und sachlich unbeschränkt zu verwerten. Darunter fällt namentlich aber nicht abschliessend das Recht zur Lizenzierung in jeder Art, Umfang und Form, das Recht zur Bearbeitung und damit zur Nutzung z. B. der Software oder Komponenten hiervon als Grundlage eines neuen schutzfähigen Guts. Die Parteien erklären sich gegenseitig den Verzicht auf Namensnennung bei der Verwertung der Schutzrechte und des Know-how durch eine oder mehrere Parteien gemeinsam und stimmen namentlich zu, dass jede Partei allein unter ihrem eigenem Namen die Schutzrechte resp. das Know-how verwertet. Die vorliegende gegenseitige unentgeltliche Übertragung der Schutzrechte resp. des Know-how bezieht sich auch auf Verwertungsarten, welche heute noch nicht bekannt sind.

Rapperswil, den..... .....
Die Studentin/der Student

Rapperswil, den..... .....
Der Betreuer / die Betreuerin der Bachelorarbeit

Rapperswil, den..... .....
Der Studiengangleiter / die Studiengangleiterin

G. Sitzungsprotokolle

In der folgenden Tabelle sind die erfolgten Sitzungen festgehalten. Die detaillierten Sitzungsprotokolle finden sich in separaten Dokumenten gemäss Kapitel A.

Datum	**Wer**	**Bemerkungen**
15.09.2009	pheinzma, dvouilla, lwilhelm	Kick-Off Meeting
22.09.2009	pheinzma, dvouilla, lwilhelm	Besprechung Anforderungen aus Kundensicht
23.09.2009	sracine, pheinzma, dvouilla, lwilhelm	Besprechung Anforderungen mit Huber+Suhner
06.10.2009	pheinzma, dvouilla, lwilhelm	Besprechung Anforderungsspezifikation
13.10.2009	pheinzma, dvouilla, lwilhelm	Vorstellung Proof-Of-Concept. Suche einer Lösung auf der man aufbauen kann.
20.10.2009	pheinzma, dvouilla, lwilhelm	Besprechung Domain Model, Anforderungsspezifikation
04.11.2009	srudin, pheinzma, dvouilla, lwilhelm	Zwischenpräsentation vor Koreferenten
17.11.2009	pheinzma, dvouilla, lwilhelm	Besprechung des aktuellen Entwicklungsstandes
24.11.2009	sracine, dvouilla, lwilhelm	Besprechung des GUIś
30.11.2009	pheinzma, dvouilla, lwilhelm	Besprechung Abgabedokumente, Entwicklungsstand
14.12.2009	pheinzma, dvouilla, lwilhelm	Besprechung Management-Summary, Poster

H. Inhaltsverzeichnis CDROM

- *doc*: Dokumentation
  - *Bericht.pdf*: Dokumentation des Projekts
  - *Zeitplan.xls*: Zeitplan
  - *Zeitplan_urspruenglich.xls*: erste Version des Zeitplans
  - *Benutzeranleitung.pdf*: Benutzeranleitung der Applikation
  - *protokolle*: Sitzungsprotokolle
  - *javadoc*: Javadoc Entwicklerdokumentation SL60-MS
  - *quelldateien*: Quelldateien der Dokumentation
- *src*: Source Code
  - *src*: Source Code SL60-MS
  - *proofofconcept*: Source Code Proof Of Concept
  - *eclipse*: Eclipse Projekt SL60-MS
  - *netbeans*: Netbeans Gui Builder Projekt SL60-MS
- *bin*: Ausführbare Applikation

Glossar

Admin-Rolle Benutzer-Rolle in der alle Funktionen der Software nutzbar sind., 42, 46

Administrator Benutzer der das Recht hat neue Benutzer zu erstellen. Nicht zu verwechseln mit der Admin-Rolle, 46

Benutzer Benutzer der SL60-MS Software. Benutzergruppe ist im Kapitel 5.2.1 genauer definiert., 43, 125

Elementary Business Process Eine Aufgabe, die von einer Person an einem Ort zu einem Zeitpunkt als Reaktion auf ein Geschäftsereignis ausgeführt wird (siehe [Lar05, S. 124])., 45

FCAPS Abkürzung für unterschiedliche Bereiche des Netzwerkmanagement nach ISO. Dazu gehören Fault-, Configuration-, Accounting-, Management-, und Security Management., 45

Kunde Als Kunde wird der Auftraggeber der SL60-MS bezeichnet, namentlich die Firma Huber+Suhner, bezeichnet., 31, 38, 42, 47

SL60-Link Bezeichnet die physikalische Verbindung zwischen zwei SL60-Terminals, 48, 61

SL60-MS Die sich im Entwicklungsprozess befindende Software für das Management von mehreren SL60 Geräten., 6, 43, 45, 49–51, 53, 55, 73, 74, 76, 103, 107, 120, 125, 137, 138, 157, 167, 171

SL60-MS Host System Bezeichnung für den Computer auf die SL60-MS Software läuft., 43, 51, 119

SL60-MS System Bezeichnung des Einsatzgebietes, der Geräte und der Akteure in deren Zusammenhang die SL60-MS eingesetzt werden soll., 43, 55

SL60-Terminal Dieser Begriff wird im Kontext dieser Dokumentation verwendet für das Produkt SL-60 der Firma Huber+Suhner., 40, 41, 43, 46–51, 56, 58, 59, 61, 63, 68–72, 74, 125, 167

User-Rolle Benutzer-Rolle in der Software nur eingeschränkt genutzt werden kann., 42, 46

Literaturverzeichnis

[BK06] Christian Bauer and Gavin King. *Java Persistence with Hibernate*. Manning, revised edition, 2006.

[BMRS96] Frank Buschmann, Regine Meunier, Hans Rohnert, and Peter Sommerlad. *Pattern - Oriented Software Architecture*. John Wiley & Sons Inc, 1996.

[Dal05] Matthias Kalle Dalheimer. *Latex kurz und gut*. O'Reilly, 2 edition, 2005.

[fS01] International Organization for Standardization. Iso/iec 9126-1:2001, software engineering - product quality, 2001.

[Han07] Robert S. Hanmer. *Patterns for Fault Tolerant Software*. John Wiley & Sons, Ltd, 2007.

[Hel96] Gilber Held. *LAN Management with SNMP and Rmon*. John Wiley & Sons Inc, 1 edition, 1996.

[Hub08] Huber+Suhner. *Installation and User Manual SL60-3001/SL60-4001*. Huber+Suhner, revision d edition, 2008.

[KR08] James F. Kurose and Keith W. Ross. *Computernetzwerke: Der Top-Down-Ansatz*. Pearson Studium, 4 edition, 2008.

[Kun] Alexander Kunkel. Jpa mit hibernate. PDF verfügbar unter `http://www.kunkelgmbh.de/jpa/jpa.html`.

[Lar05] Craig Larman. *UML2 und Patterns angewendet*. mitp, 1 edition, 2005.

[NMSa] Open NMS. Benutzerdokumenation open nms. Verfügbar unter `http://www.opennms.org/wiki/Main_Page`.

[NMSb] Open NMS. Entwicklerdokumentation im javadoc stil. Verfügbar unter `http://www.opennms.org/documentation/java-apidocs-stable/`.

[SUN] SUN. Java seite von sun. Verfügbar unter `http://www.java.com/en/download/help/6000011000.xml`.

Abbildungsverzeichnis