

**Fachhochschule Köln
&
Fachhochschule Dortmund**

Verbundstudiengang Wirtschaftsinformatik Master

Ausarbeitung zur Erlangung
des Leistungsnachweises im Modul

„Fortgeschrittene Software Entwicklung“

Thema

„Simulation von Siemens-DU05-Prozedur Telegrammen“

von	Michael Burkhardt	7062922
	Sven Münzner	11047554
	Michael Rongen	11044347
	Thomas Baier	7071757
	Stefan Albert	7042791

vorgelegt am: 30.07.2010

Dozent: Prof. Dr. Mario Winter

Inhaltsverzeichnis

I. Einleitung	9
1. Vorwort	10
1.1. Einstieg	10
1.2. Lastenheft	10
1.3. Pflichtenheft	11
1.4. Grobentwurf	11
1.5. Feinentwurf	12
1.6. Implementation	12
1.7. Ausblick	12
2. Einsatzumgebung	13
3. Aufteilung	14
4. Projektplan	15
5. Vorgehensmodell SCRUM	23
5.1. SCRUM-Framework	23
5.2. Rollen	25
5.3. Projektablauf	26
6. Erfahrungsbericht	29
II. Lastenheft	30
1. Vorwort	32

2. Zielbestimmung und Zielgruppen	33
2.1. Produktperspektive	33
2.2. Einsatzkontext	35
2.3. Zielgruppe	37
3. Produktdaten (Programm-Spezifische Daten)	39
3.1. Telegrammdefinition	39
3.2. Verbindungsparameter	39
3.3. Email	39
3.4. Ein- / Ausgabedateien	39
4. Funktionale Anforderung	40
4.1. Produktanforderungen	40
4.2. Muss-Anforderung	42
4.3. Allgemeine Beschreibung der Anforderungen	42
4.4. Präzise Beschreibung der Anforderungen	45
4.4.1. Grafische Benutzeroberfläche	45
4.4.2. Kommunikation zum Kommunikationspartner	47
4.4.3. Telegrammdefinition	49
4.4.4. Dauerhafte Speicherung von Projekt-Parametern	50
4.4.5. Kommunikation	50
4.4.6. Informationsweitergabe	51
4.4.7. Programm-Handling	52
4.5. Kann-Anforderung	54
4.5.1. AF70 Validierung des Telegramms	54
4.5.2. AF80 Validierung IP Adresse	54
4.5.3. AF90 Validierung des Ports	54
4.5.4. AF100 Senden und empfangen mit verschiedenen - Partnern über das gleiche Programm	55
4.5.5. AF110 Telegramm Analyse (Nettodaten verifizieren)	55

4.5.6. Nähere Beschreibung	55
5. Nicht-Funktionale Anforderung	56
5.1. NF10 Einfachheit des Programms	58
5.2. NF20 Programmierung in CSharp	58
5.3. NF30 Qualitätsanforderung	58
5.3.1. Softwarequalität	58
5.3.2. Übertragbarkeit	60
5.3.3. Anwendbarkeit	60
5.3.4. Effizienz	60
5.3.5. Wartbarkeit	61
5.3.6. Zuverlässigkeit	61
5.3.7. Sicherheit	62
5.4. Technische Anforderung und Einschränkungen	62
5.4.1. Lizenzen	63
6. Architektur des Gesamtsystems	64
7. Skizzen der Benutzeroberfläche	66
8. Abnahmekriterien	69
9. Erfahrungsbericht (Fazit)	70
 III. Pflichtenheft	 72
1. Allgemeines	74
1.1. Sinn und Zweck des Dokumentes	74
1.2. Zielbestimmung	74
1.3. Zielgruppen	75

2. Produktdaten	76
2.1. Telegrammdefinition	76
2.2. Verbindungsparameter	76
2.3. Email	77
2.4. Ein- / Ausgabedateien	78
3. Funktionale Anforderung	79
3.1. Anwendungsfälle	79
3.1.1. Grafische Benutzeroberfläche	79
3.1.2. Kommunikation zum Kommunikationspartner	84
3.1.3. Telegrammdefinition	86
3.1.4. Dauerhafte Speicherung von Projekt-Parametern	88
3.1.5. Kommunikation	88
3.1.6. Informationsweitergabe	90
3.1.7. Programm-Einschränkungen	91
3.1.8. Use Case: Definition Telegramme	93
3.2. Use Case: Definition Verbindungsparameter	97
3.2.1. Use Case: Definition E-Mail-Adressen	100
3.2.2. Use Case: Definition Speichern und Laden der Projektdatei	103
3.2.3. Szenarien	106
3.3. Akteure	106
3.3.1. internen Inbetriebnehmer	107
3.3.2. internen Softwareentwickler	108
3.3.3. externen Inbetriebnehmer	108
3.3.4. externe Softwareentwickler	109
3.3.5. Leitstand-Personals	110
3.4. Anwendungsfalldiagramm	110
3.5. Klassen	113
3.6. Verhaltensdiagramme Anforderungsanalyse	113

3.7. Benutzungs- und Systemschnittstellen	113
4. Nicht-Funktionale Anforderung	113
4.1. NF10 Einfachheit des Programms	113
4.2. NF20 Programmierung in CSharp	113
4.3. NF30 Qualitätsanforderung	114
4.3.1. Softwarequalität	114
4.3.2. Übertragbarkeit	115
4.3.3. Anwendbarkeit	115
4.3.4. Effizienz	116
4.3.5. Wartbarkeit	116
4.3.6. Stabilität	117
4.3.7. Sicherheit	117
4.4. Technische Anforderung und Einschränkungen	118
4.4.1. Installation	118
4.4.2. Betriebssystem	118
4.4.3. Hardware	119
4.4.4. Netzinfrastruktur	120
4.4.5. Lizenzen	120
5. Benutzeroberflächen	121
5.1. Architektur	123
5.2. Technische Anforderungen und Einschränkungen	124
6. Fachliche Architektur	124
6.1. Klassendiagramme Softwarespezifikation	124
7. Lieferumfang	127
8. Abnahmekriterien	127
8.1. Vorgaben des Lastenheftes	127
8.2. Abnahmetest	127

9. Erfahrungsbericht (Fazit)	130
 IV. Grobentwurf	 134
1. Einleitung Grobentwurf	136
2. Telegrammdefinition	138
2.1. Dialog	138
2.2. Aktivitätsdiagramm	139
2.3. Sequenzdiagramm	140
3. Verbindungsparameter	141
3.1. Dialog	141
3.2. Aktivitätsdiagramm	142
3.3. Sequenzdiagramm	143
4. Speichern des Projektes	144
4.1. Dialog	144
4.2. Aktivitätsdiagramm	145
4.3. Sequenzdiagramm	146
5. Email Adressen definieren	147
5.1. Dialog	147
5.2. Aktivitätsdiagramm	148
5.3. Sequenzdiagramm	149
6. Email Senden	150
6.1. Dialog	150
6.2. Aktivitätsdiagramm	151
6.3. Sequenzdiagramm	152
7. Erfahrungsbericht	153

V. Feinentwurf	154
1. Einleitung Feinentwurf	156
2. Programmübersicht	156
2.1. Hauptfenster	156
3. Telegrammdefinition	158
3.1. Dialog	158
3.2. Aktivitätsdiagramm	159
3.3. Sequenzdiagramm	160
4. Verbindungsparameter	161
4.1. Dialog	161
5. Speichern des Projektes	162
5.1. Dialog	162
6. E-Mail Adressen definieren	163
6.1. Dialog	163
7. E-Mail senden	164
7.1. Dialog	164
8. Erfahrungsbericht	165
VI. Implementierung	168
1. Einleitung Implementierung	170
1.1. Produkt-Backlog	171
1.2. Systemumgebung	172
1.3. Netzwerkumgebung	172
1.4. Kommunikation	172

1.5. Verbindungsaufbau Testumgebung	173
1.6. Partnersysteme	173
1.7. Installation	174
 VII. Prototyp	 175
1. Beschreibung Prototyp	176
2. Installationsvoraussetzung	177
3. Start des Programms	179
3.1. Dialog E-Mail Adressen	183
3.2. Dialog Verbindungsparameter	183
3.3. Dialog sende Mail	184
3.4. Dialog Telegramm definition	184
4. Deinstallation	185
 VIII.Abschluss	 186
1. Fazit	187
2. Ausblick	190
 IX. Anhang	 191
1. Anhang	192
1.1. Meeting-Protokolle	192

X. Abkürzungsverzeichnis	201
Literaturverzeichnis	203

Abbildungsverzeichnis

1.	Der alte Projektplan	17
2.	Der neue Projektplan	19
3.	Meeting Planung	22
4.	SCRUM Modellübersicht	26
5.	Übersicht Stakeholder	37
6.	Übersicht Gesamtsystem	65
7.	Skizze des Hauptfensters	66
8.	Skizze IP Kommunikations Dialog	66
9.	Skizze Telegrammdefinitions-Dialog	67
10.	Skizze E-Mail Adressen Dialog	67
11.	Skizze E-Mail senden-Dialog	68
12.	Sequenz-Diagramm Gesamtübersicht	111
13.	Sequenz-Diagramm Gesamtübersicht	111
14.	Sequenz-Diagramm Gesamtübersicht	112
15.	Sequenz-Diagramm Gesamtübersicht	112
16.	Spezifizierte Benutzeroberfläche Hauptfenster	121
17.	Dialog Zeichnung Mail Adressen definition	121
18.	Dialog Telegrammdefinition	122
19.	Dialog Verbindungsparameter	122
20.	Dialog Mail senden	123
21.	MVC Architektur	123
22.	Klassen-Diagramm DU05 Simulator	125
23.	Klassen-Diagramm Kommunikation	132
24.	Klassen-Diagramm Gesamtübersicht	133
25.	Dialog Telegrammdefinition	138
26.	Aktivitätsdiagramm Telegrammdefinition	139
27.	Sequenzdiagramm Telegrammdefinition	140
28.	Dialog Verbindungsparameter	141

29. Aktivitätsdiagramm Verbindungsparameter	142
30. Sequenzdiagramm Verbindungsparameter	143
31. Dialog Speichern des Projektes	144
32. Aktivitätsdiagramm Speichern des Projektes	145
33. Sequenzdiagramm Speichern des Projektes	146
34. Dialog Email-Adressen definieren	147
35. Aktivitätsdiagramm Email-Adressen definieren	148
36. Sequenzdiagramm Email-Adressen definieren	149
37. Dialog Email senden	150
38. Aktivitätsdiagramm Email senden	151
39. Sequenzdiagramm Email Senden	152
40. Darstellung Simulator Hauptfenster	157
41. Laufender DU05 Simulator	158
42. Telegrammdefinitions-Dialog	158
43. Aktivitätsdiagramm Telegrammdefinition	159
44. Sequenzdiagramm Telegrammdefinition	160
45. Verbindungsparameter-Dialog	161
46. Speichern des Projektes	162
47. E-Mail Adressen Dialog	163
48. E-Mail Adressen senden Dialog	164
49. Übersicht Product-Backlog	171
50. Prototyp: Information Startordner	179
51. Prototyp: Programm DU05 Simulator	179
52. Prototyp: Aussehen der Symbolleiste	181
53. Prototyp: Dialog Eingabe E-Mail Adressen	183
54. Prototyp: Dialog Verbindungsparameter	183
55. Prototyp: Dialog E-Mail Versand	184
56. Prototyp: Dialog Telegramm definition	184
57. Meeting Protokoll 1	192

58.	Meeting Protokoll 2	193
59.	Meeting Protokoll 3	194
60.	Meeting Protokoll 4	195
61.	Meeting Protokoll 5	196
62.	Meeting Protokoll 6	197
63.	Meeting Protokoll 7	198
64.	Meeting Protokoll 8	199

Tabellenverzeichnis

1.	Übersicht über die Aufteilung der Tätigkeiten	1
2.	Übersicht über die Aufteilung der Tätigkeiten	2
3.	Übersicht über die Aufteilung der Tätigkeiten	3
4.	Übersicht über die Aufteilung der Tätigkeiten	4
5.	Übersicht über die Aufteilung der Tätigkeiten	5
6.	Übersicht über die Aufteilung der Tätigkeiten	6
7.	Übersicht über die Aufteilung der Tätigkeiten	7
8.	Übersicht über die Aufteilung der Tätigkeiten	8
9.	Übersicht über die Aufteilung der Tätigkeiten	14
10.	Versionsübersicht Lastenheft	31
11.	Versionsübersicht Pflichtenheft	73
12.	Versionsübersicht Grobentwurf	135
13.	Versionsübersicht Feinentwurf	155
14.	Versionsübersicht Implementierung	169

Individuelle Anteile

Einleitung

Kapitel	Hauptautor	Nebenantor
1	Michael Burkhardt	Michael Rongen, Stefan Albert, Sven Münzner
2	Michael Burkhardt	Michael Rongen, Stefan Albert, Sven Münzner, Thomas Baier
3	Michael Burkhardt	Michael Rongen, Stefan Albert, Sven Münzner, Thomas Baier
4	Sven Münzner	Thomas Baier, Stefan Albert
5	Michael Burkhardt	Sven Münzner, Stefan Albert
6	Michael Burkhardt	Sven Münzner Michael Rongen

Tabelle 1: Übersicht über die Aufteilung der Tätigkeiten

Lastenheft

Kapitel	Hauptautor	Nebenautor
1	Michael Burkhardt	Sven Münzner, Michael Rongen, Stefan Albert
2	Sven Münzner	Stefan Albert, Michael Rongen
3	Michael Burkhardt	Sven Münzner, Michael Rongen, Stefan Albert
4	Michael Burkhardt	Sven Münzner, Michael Rongen, Stefan Albert
5	Michael Burkhardt	Stefan Albert, Michael Rongen
6	Michael Burkhardt	Sven Münzner, Thomas Baier
7	Stefan Albert	Sven Münzner, Michael Burkhardt
8	Michael Burkhardt	Michael Rongen, Sven Münzner
9	Michael Burkhardt	Sven Münzner, Thomas Baier, Stefan Albert

Tabelle 2: Übersicht über die Aufteilung der Tätigkeiten

Pflichtenheft

Kapitel	Hauptautor	Nebenautor
1	Sven Münzner	Michael Burkhardt, Thomas Baier, Stefan Albert, Sven Münzner
2	Michael Rongen	Michael Burkhardt, Thomas Baier, Stefan Albert, Sven Münzner
3	Michael Rongen	Michael Burkhardt, Thomas Baier, Stefan Albert
4	Thomas Baier	Michael Burkhardt, Michael Rongen, Stefan Albert
5	Thomas Baier	Michael Burkhardt, Michael Rongen, Stefan Albert, Sven Münzner
6	Sven Münzner	Michael Burkhardt, Thomas Baier, Stefan Albert
7	Sven Münzner	Michael Burkhardt, Thomas Baier, Stefan Albert
8	Sven Münzner	Michael Burkhardt, Thomas Baier, Stefan Albert
9	Thomas Baier	Michael Burkhardt, Michael Rongen, Stefan Albert

Tabelle 3: Übersicht über die Aufteilung der Tätigkeiten

Grobentwurf

Kapitel	Hauptautor	Nebenantor
1	Michael Burkhardt	Thomas Baier, Stefan Albert, Sven Münzner, Michael Rongen
2	Stefan Albert	Michael Burkhardt, Thomas Baier, Sven Münzner
3	Sven Münzner	Michael Burkhardt, Michael Rongen, Sven Münzner
4	Thomas Baier	Michael Burkhardt, Stefan Albert, Sven Münzner
5	Thomas Baier	Michael Burkhardt, Stefan Albert, Sven Münzner
6	Thomas Baier	Michael Burkhardt, Stefan Albert, Sven Münzner
7	Thomas Baier	Michael Burkhardt, Stefan Albert, Sven Münzner

Tabelle 4: Übersicht über die Aufteilung der Tätigkeiten

Feinentwurf

Kapitel	Hauptautor	Nebenautor
1	Thomas Baier	Michael Burkhardt, Sven Münzner, Stefan Albert
2	Thomas Baier	Michael Burkhardt Sven Münzner, Stefan Albert
3	Michael Burkhardt	Sven Münzner, Michael Rongen, Stefan Albert, Thomas Baier
4	Michael Rongen	Sven Münzner, Stefan Albert, Thomas Baier
5	Stefan Albert	Michael Burkhardt, Michael Rongen, Sven Münzner
6	Sven Münzner	Stefan Albert, Michael Burkhardt, Thomas Baier
7	Sven Münzner	Stefan Albert, Michael Burkhardt, Thomas Baier
8	Thomas Baier	

Tabelle 5: Übersicht über die Aufteilung der Tätigkeiten

Implementierung

Kapitel	Hauptautor	Nebenantor
Alle	Michael Burkhardt	Sven Münzner, Michael Rongen, Stefan Albert, Thomas Baier

Tabelle 6: Übersicht über die Aufteilung der Tätigkeiten

Beschreibung Prototyp

Kapitel	Hauptautor	Nebenautor
Alle	Michael Burkhardt	Sven Münzner, Thomas Baier, Michael Rongen, Stefan Albert

Tabelle 7: Übersicht über die Aufteilung der Tätigkeiten

Abschluss

Kapitel	Hauptautor	Nebenautor
Alle	Michael Burkhardt	Sven Münzner, Thomas Baier, Michael Rongen, Stefan Albert

Tabelle 8: Übersicht über die Aufteilung der Tätigkeiten

Teil I.

Einleitung

1. Vorwort

Das Folgend beschriebene Projekt wurde gewählt, da es in der Praxis zum Einsatz kommt. Die hier zusammengefasste Artefakte sind im Verbundstudiums-Modul *Erweiterte Software Technologie* entstanden und zeigen die Vorgehensweise in der Software-Entwicklung.

1.1. Einstieg

Der Einstieg bietet einen Überblick über den Aufbau des gesamten Dokumentes und beschreibt in welcher Form die Projektplanung und Realisierung durchgeführt wurde. Des Weiteren wird aufgezeigt wie das Gesamtprojekt der Hausarbeit unter den Teammitgliedern aufgeteilt wurde.

1.2. Lastenheft

Das Lastenheft definiert die Anforderungen der Stakeholder an das Gesamtsystem DU05 Telegramm Simulator. Hier werden die unterschiedlichen Funktionen des Systems vorgestellt und funktionale, nicht funktionale Anforderungen sowie die Zielgruppen definiert. Der Aufbau des Hauptfensters (GUI) wird an-

hand eines papierbasierten Prototypen vorgestellt.

1.3. Pflichtenheft

Die Anforderungen an das Programm DU05 Telegramm Simulator werden im Rahmen des Pflichtenheftes aus Sicht des Entwicklers beschrieben. Die genaue Qualifizierung der nicht-funktionalen Anforderungen, das Architekturkonzept, die Softwarespezifikation und die Definition von Testfällen werden im Pflichtenheft genauer beschrieben. Im Rahmen der Hausarbeit wurde zwar das gesamte System entwickelt, im Pflichtenheft wird aber nur auf Kernfunktionen der Anforderungen eingegangen. Der vollständige Softwareentwicklungsprozess wird nachvollziehbar dargestellt.

1.4. Grobentwurf

Das Kapitel Grobentwurf beinhaltet neben den Domänenklassendiagrammen auch sämtliche Grobentwurfsdiagramme. Jeder im Pflichtenheft beschriebener Anwendungsfall wurde auch im Grobentwurf berücksichtigt und umgesetzt.

1.5. Feinentwurf

Der Feinentwurf beinhaltet den Anwendungsfall „senden, empfangen und anzeigen von Telegrammen“, der in einem Feinentwurfsverhaltensdiagramm umgesetzt wurde.

1.6. Implementation

Die Implementation beschreibt die Entwicklung des Programms *DU05 Telegramm Simulation* bzw. der Bereich die im Rahmen des Pflichtenheftes sowie im Grob- und Feinentwurfs umgesetzt wurde.

1.7. Ausblick

In diesem Abschnitt werden Hürden während des Projektverlaufes beschrieben sowie Reflexionen von den Teammitgliedern während Ihrer Hausarbeitstätigkeiten dargestellt. Die im Pflichtenheft nicht genauer beschriebenen Anforderungen der Software *DU05 Telegramm Simulator* wurden in den Anhang der Hausarbeit verschoben. Falls beim Durcharbeiten des Dokumentes bzw. des Pflichtenheftes offene Fragen auftauchen kann ein Blick in den Anhang hilfreich sein.

2. Einsatzumgebung

Die ThyssenKrupp Steel Europe AG (TKSE) ist ein stahlproduzierendes Unternehmen, in dem unterschiedliche Stahlsorten mit unterschiedlichster Qualität hergestellt werden. Das Unternehmen produziert im 24-Stunden-Dauerbetrieb Stähle für die Automobil-, Hausgeräte- und Bauindustrie sowie Energietechnik.

Zur Berechnung von Materialzugaben durch computergesteuerte Modelle für metallurgische Verfahren wird bei der TKSE fortwährend betriebsspezifische Software entwickelt. Die Ausfallsicherheit dieser Software bzw. Softwaresysteme muss daher extrem hoch sein. Produktionsstörungen sowie Produktionsausfälle sind in diesem betrieblichen Umfeld zwangsläufig mit einem hohen finanziellen Mehraufwand verbunden.

In naher Zukunft werden Software-Teile aus den alten Rechner-Anlagen (Level 2 Systeme) in neue externe Rechnersysteme (Level 1 Systeme) umgesiedelt und re-enginiert müssen. Auch hier ist die Ausfallsicherheit der neuen Rechner-Anlagen von großer Wichtigkeit. Um die Kommunikation im TKSE-(Siemens)-Standard zum Level 1 Partner-System bis ins kleinste Detail vorab testen zu können muss eine Software entwickelt werden, die eine Verbindung zu den neuen Partner-Systemen aufbau-

en, Test-Telegramme erzeugen und senden kann.

3. Aufteilung

Die Aufgabenstellung der TKSE innerhalb der Gruppe wurde anhand des fachlichen Kenntnisstandes in folgende Teilaufgaben unterteilt:

Person	Tätigkeit
Michael Burkhardt	Grob- und Feinentwurf der Kommunikationsschnittstelle. Programmierung der DU05 Siemens Prozedur.
Stefan Albert	Programmierung der Druckerschnittstelle mit Crystal Reports.
Sven Münzner	E-Mail Versand und Entwicklung der dauerhaften Speichermöglichkeit der Kommunikationsergebnisse.
Michael Rongen	Grob- und Feinentwurf Entwicklung der Graphical User Interfaces des Hauptfensters.
Thomas Baier	Grob- und Feinentwurf der Dialogschnittstellen für Kommunikationsparameter und Telegrammdefinitionen.
Gemeinsame Tätigkeiten	Erstellung der Ausarbeitung zu gleichen Teilen

Tabelle 9: Übersicht über die Aufteilung der Tätigkeiten

4. Projektplan

Im Folgenden der Projektplan: Auf Grund der Größe des Projektes und der Teamstärke von fünf Personen war allen Teammitgliedern von vornherein klar, dass das Projekt nur mit Hilfe eines angemessenen Projektmanagements und einer Zerlegung in Projektteilaufgaben durchgeführt werden kann. Der erste Ansatz zur Bearbeitung des Projektes bestand darin, eine Aufgabenteilung zwischen den einzelnen Teammitgliedern durchzuführen.

Die Teilaufgaben wurden so geplant, dass durch Schnittstellen das Programm wieder zu einem Ganzen zusammengefügt werden kann, um somit ein unabhängiges Arbeiten ermöglichen zu können und den Kommunikationsaufwand untereinander zu reduzieren. In der Abbildung 1 dargestellten ersten Version des Projektplans, sind die ersten Überlegungen des gemeinsamen Vorgehens aufgezeigt. Hier wurde ein sequenzielles Vorgehen, wie beim Wasserfall Modell gewählt.

Wie in dem Projektplan ersichtlich, gliederte sich das Projekt in 4 Phasen wobei die erste Phase mit dem Projektstart zwischen dem 27.02.2010 und dem 15.03.2010 stattfand. Hier fanden die ersten Überlegungen zum neuen Programm statt und Ziele wurden definiert. In der zweiten Phase, dem Grobent-

wurf, sollten vertiefende Gespräche mit dem Auftraggeber zu einer Festlegung der Anforderungen und dem ersten Entwurf von Pflichten- und Lastenheft führen. Zusätzlich sollten zu allen relevanten Prozessen User-Stories definiert und gesammelt werden. Ergänzend dazu sollten schon erste Diagramme und eine grobe Architektur erzeugt werden. Danach sollte in der Phase des Feinentwurfs die erzeugten Diagramme spezifiziert werden und an die Zielprogrammiersprache angepasst werden. Nachdem alles angepasst wurde sollte die Programmierung des DU05 Simulators beginnen. In der letzten Phase sollten dann die Erarbeitung der Präsentation und der, im Zuge der Vorlesung fortgeschrittene Softwareentwicklung geforderte, Hausarbeit stattfinden.

Hier der alte Projektplan.

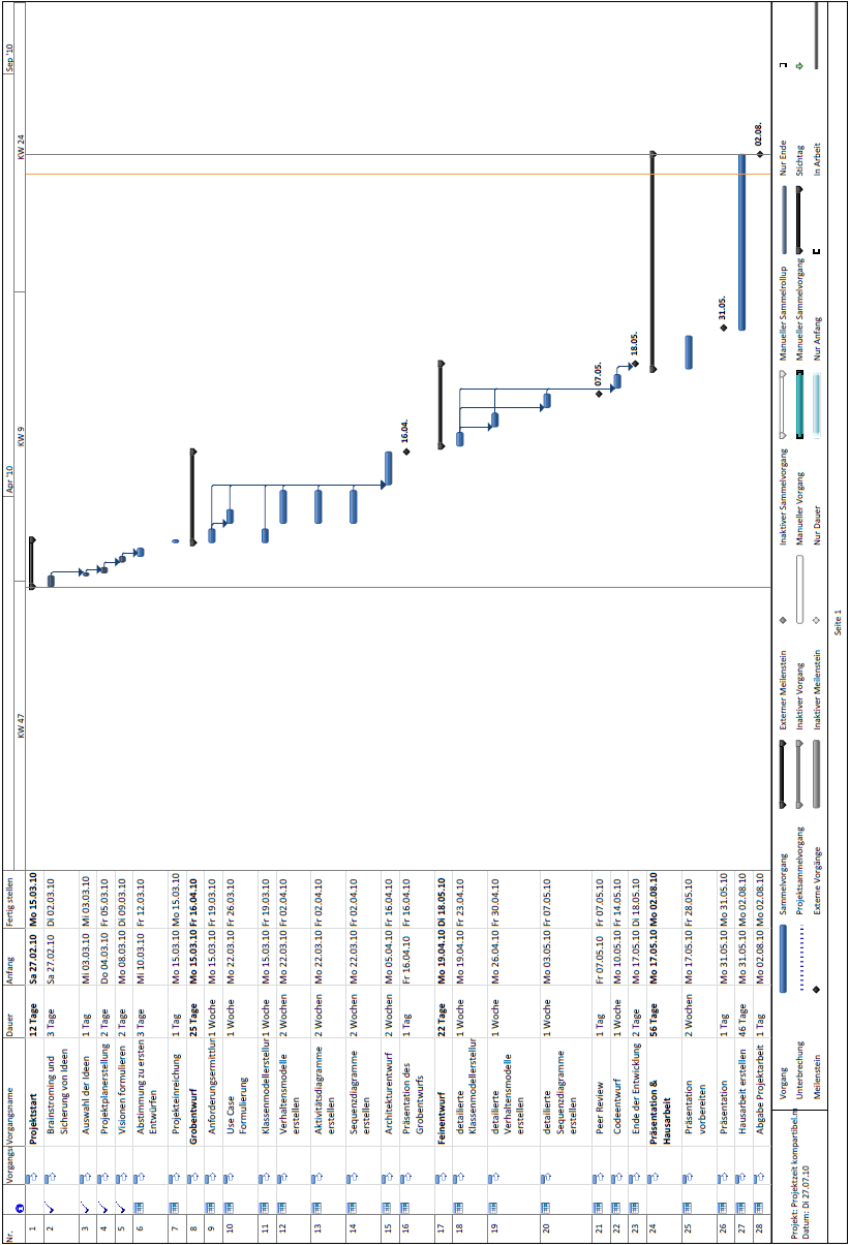
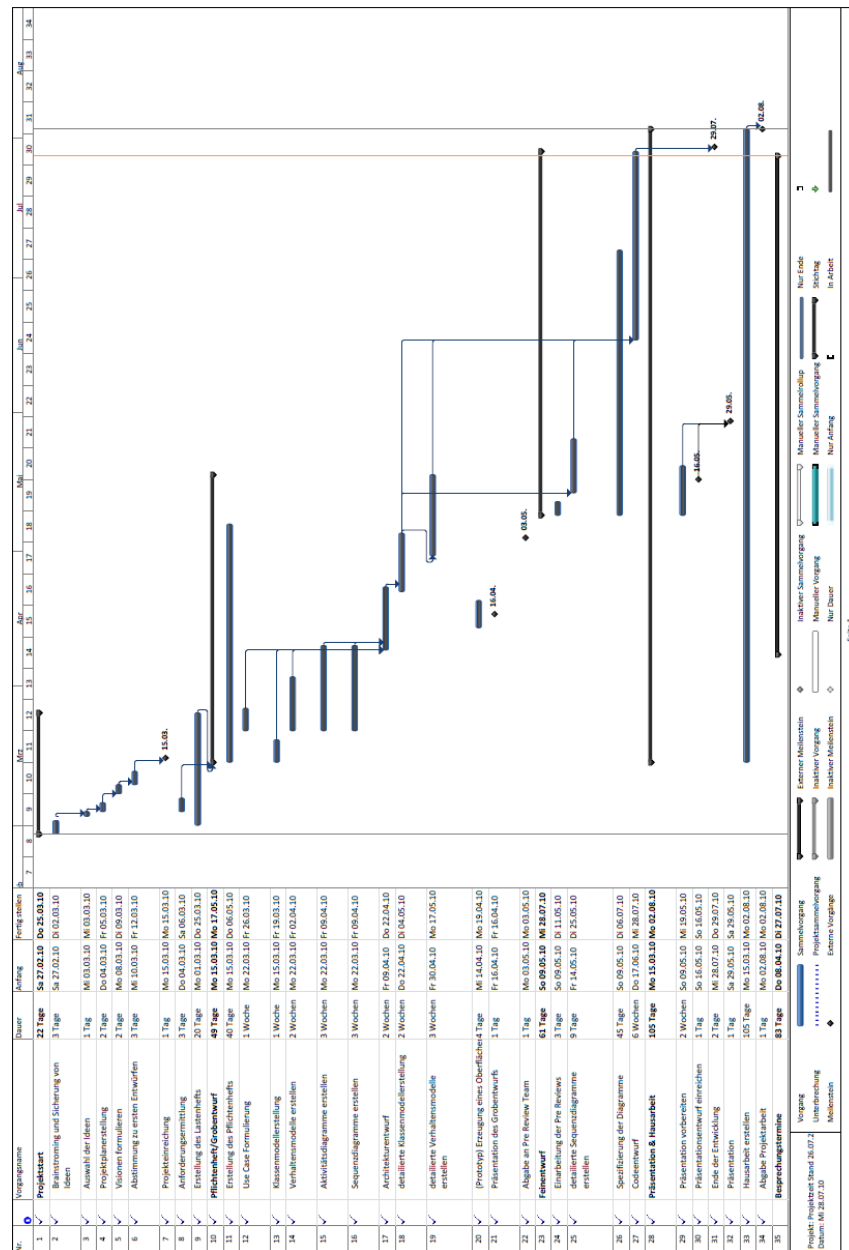


Abbildung 1: Der alte Projektplan

Schon kurz nach Beginn des Projekts und mit Absprache des Professors, stellte sich der Projektplan durch das sequenzielle und starre Vorgehen, als nicht geeignet heraus. Es musste ein Vorgehensmodell gefunden werden, was den Anforderungen unseres Projektteams mehr entsprach und vor allem die unterschiedlichen Wissensstände und Fähigkeiten mehr berücksichtigte. Deshalb wurde SCRUM als Vorgehensmodell gewählt, welches in Kapitel 5 vorgestellt wird. Es bietet den Vorteil einer leicht zu erlernenden agilen Vorgehensweise, in dem die geographisch verstreuten Entwicklerteams eigenständig arbeiten konnten. Des Weiteren ein klar definiertes Rollenkonzept, welches einen flexiblen Entwicklungsprozess ermöglicht. Dies kam unserem Vorgehen entgegen. Da die im Modell vorkommenden Daily Meetings durch andere Kommunikationswege ersetzt wurden, tauchen diese nicht im Projektplan auf. Die Sprint-Einträge wurden eins zu eins genutzt, jedoch auf Sprint-Meetings im Sinne von SCRUM wurde verzichtet. Diese wurden auf Grund des hohen kommunikativen Aufwands und dem immensen Zeitverzug, durch Nutzung von asynchroner Kommunikation wie z.B. E-Mail, durch wöchentliche Meetings ersetzt.

Programm DU05 Simulator



Aufgrund dieser veränderten Struktur konnte das Projekt innerhalb des Projektzeitraums fertig gestellt werden. Der Projektplan musste aber vor allem in der Grob- und Feinentwurfsphase zeitlich stark nach hinten korrigiert werden. Auch die Erstellung der Hausarbeit wurde als ein stetiger Prozess über den gesamten Projektzeitraum festgelegt, was den Zeitdruck zum Ende des Projekts reduzierte. Die Schwierigkeiten, die zu diesem Zeitverzug führten, waren die Einarbeitung in das komplexe Projektumfeld des Auftraggebers, sowie die festgelegten Vorgaben (z.B. die zu verwendende Programmiersprache). Hinzu kamen im Laufe des Projektes immer wieder ergänzende oder anpassende Anforderungen des Auftraggebers, die ebenfalls ein zügiges Fortschreiten erschwerten. Neben diesen externen Einflüssen kamen aber auch Fehlentscheidungen des Projektteams hinzu (Notwendiger Wechsel des Gestaltungswerkzeuges), die ebenfalls zu Verzögerungen führten. Dies ist im neuen Projektplan ersichtlich Abbildung 2. Durch diese Faktoren konnte der frühere Projektplan nicht eingehalten werden, so dass wir zum Zeitpunkt der Präsentation, Ende Mai, im Anfang der Phase des Feinentwurfs und nicht wie geplant mit der Entwicklung fertig waren. Diese Verzögerungen begleiteten uns während unseres kompletten Projektes, konnten aber im Laufe der Zeit wieder aufgeholt werden, da in der Schlussphase parallel am Feinentwurf und der Programmierung gearbeitet wurde.

So konnte das Projekt von der Entwicklung her, Ende Juli beendet werden. Eine frühere Terminierung fiel den Vorbereitungen auf verschiedene Klausuren der Teammitglieder zum Opfer. Abschließend lässt sich festhalten, dass die zeitliche Planung in der Umsetzung des Grob- und Feinentwurfs zu optimistisch definiert wurde. Aber durch die intensive Kommunikation, besonders während der wöchentlichen Meetings eine intensive und schnelle Zusammenarbeit ermöglicht wurde. Dies kann als zentraler Faktor für den Erfolg des Projekts gesehen werden kann.

Meeting-Plan:










	<input type="checkbox"/> Besprechungstermine	63 Tage	Do 08.04.10	Di 29.06.10
	Meeting	1 Tag	Do 08.04.10	Do 08.04.10
	Meeting	1 Tag	Do 29.04.10	Do 29.04.10
	Meeting (Peer Review)	1 Tag	Di 04.05.10	Di 04.05.10
	Meeting (Einarbeitung Peer Review A	1 Tag	Mi 12.05.10	Mi 12.05.10
	Meeting	1 Tag	Do 27.05.10	Do 27.05.10
	Meeting	1 Tag	Do 10.06.10	Do 10.06.10
	Meeting	1 Tag	Do 17.06.10	Do 17.06.10
	Meeting	1 Tag	Di 29.06.10	Di 29.06.10

Abbildung 3: Meetingplan

Dieser Meetingplan zeigt die Geplanten Treffen in denen Meetingprotokolle erzeugt wurden. Diese befinden sich auf der abzuliefernden CD-ROM.

5. Vorgehensmodell SCRUM

SCRUM ist ein agiles Vorgehensmodell und beinhaltet Prozesse, Rollen und Methoden. Grundidee von SCRUM ist die Selbstorganisation des Entwicklerteams und die eigenverantwortliche Auswahl der eingesetzten Mittel. Es handelt sich bei SCRUM um einen allgemeinen Managementrahmen für beliebige Projekte. SCRUM lässt den Teammitgliedern sehr viele Freiräume in der Ausgestaltung wie z.B. die Anforderungen formuliert und geschätzt werden. [?] und [?]

5.1. SCRUM-Framework

SCRUM lässt sehr viel Freiraum in der Ausgestaltung wie z.B. Anforderungen formuliert und geschätzt werden. Das SCRUM-Framework umfasst folgende Punkte [?]:

- Rollen (Produkt-Owner, SCRUM-Master, Team)
- Product-Backlog
- Daily-SCRUM-Meeting
- Sprints

- Sprint-Backlog
- Sprint-Planing-Meetings

Zu den Rollen vergleiche Kapitel 5.2. Die Anforderungen für das Produkt, welches das Team entwickelt, sind im Product-Backlog aufgelistet. Der Product-Owner ist für den Inhalt, die Verfügbarkeit, und die Priorisierung verantwortlich. Das **Product-Backlog** enthält die Features des zu entwickelnden Produkts. Es ist niemals komplett, da der Inhalt laufend verfeinert und angepasst wird, es entwickelt sich mit dem Produkt und der Umgebung in der entwickelt wird mit. Das Product-Backlog ist sortiert nach Prioritäten und enthält alle aktuell bekannten und festgelegten Anforderungen für die erfolgreiche Entwicklung und Inbetriebnahme des Software-Produkts.

Sprints sind zeitgebundene Iterationen mit Anforderungen aus dem Product-Backlog mit einer empfohlenen Maximaldauer von 30 Tagen. Nach Durchlauf eines Sprints stellt das Team die Ergebnisse im Sprint-Review vor. Der *Sprint-Backlog* beinhaltet Aufgaben, welche das Team abarbeitet und mit *erledigt* dem Product-Backlog zurückmeldet. Die Sprint-Backlog Einträge müssen abgearbeitet werden.

Das *Sprint-Planing-Meeting* findet statt, um Sprints (Iteration) zu planen. Die Dauer sollte 8 Stunden pro Planungs-Monat

nicht überschreiten. Kürzere Sprints sollten 5 % Prozent der gesamten Sprint-Länge nicht überschreiten.

5.2. Rollen

SCRUM unterscheidet bei der Aufgabenverteilung drei Arten von Mitarbeiter- Rollen:

- SCRUM-Master
- Product-Owner
- Team

Der Product-Owner ist für die Formulierung, Auswahl und Priorisierung des Softwareentwicklungsprojekts zuständig. Des Weiteren legt er in der *Definition of Done* mit dem Team fest, was *fertig* bedeutet. Dies dient der Transparenz des Projektfortschritts und definiert die Qualität jeder Iteration. Ziel soll sein, dass am Ende einer jeden Sprints (Iteration) ein lauffähiges Software-(Teil-)Produkt dem Kunden ausgeliefert werden kann. [?, vergl.] In einem Softwareprojekt vertritt der Product-Owner den Kunden und den Anwender. Das Team besteht vorwiegend aus Entwicklern die für die Umsetzung der Anforderungen verantwortlich sind. Mit dem Product-Owner werden der Zeitplan und die Priorisierung abgestimmt. Der SCRUM-Master unterstützt das

Team und den Product-Owner in schwierigen Situationen und bei Problemen. Er achtet auf die Einhaltung des Entwicklungs-Prozesses und darf weder Teil des Teams sein noch zum Product-Owner gehören werden, da dies seinen objektiven Standpunkt als neutralen Mittler gefährden würde.

5.3. Projektablauf

Übersicht über die Vorgehensweise von SCRUM:

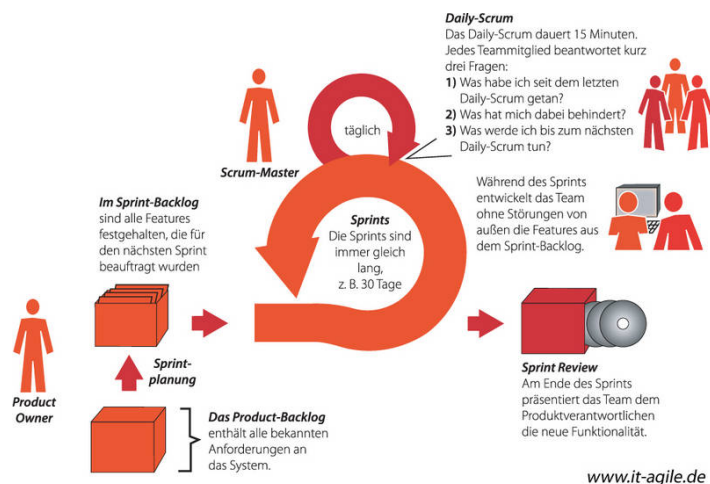


Abbildung 4: SCRUM Übersicht

Produkt-Anforderungen können von jedem Projektbeteiligten jederzeit erstellt werden. Diese Produktanforderungen werden in dem sogenannten Product-Backlog gesammelt und mit einer Priorität versehen. Der Product-Owner bestimmt selbst wie in diesem Projekt mit dem Product-Backlog umgegangen wird. Aus der gegebenenfalls großen Menge an Anforderungen aus

dem Product-Backlog wählt der Product-Owner eine angemessene Menge an Anforderungen für jedes Sprint-Backlog aus, somit beinhaltet dieses eine Ansammlungen von Anforderungen, die in einem festgelegten Zeitraum erledigt werden sollen. Ein solcher, gleich bleibende und vorab definierter Zeitraum wird bei SCRUM Sprint genannt. In der Sprint-Planung legt der Product-Owner in Zusammenarbeit mit dem Entwicklungsteam den Aufwand für die Anforderungen fest. Gemeinsam legen Product-Owner und Entwicklungsteam die Inhalte für den nächsten Sprint fest. Solche Vereinbarungen werden bei dem sogenannten Sprint-Planning-Meeting getroffen, indem auch die Aufwandsschätzung und der Auswahlprozess stattfinden.

Das Team arbeitet während eines Sprints die im Sprint-Backlog liegenden Anforderungen ungestört von äußeren Einflüssen, eigenverantwortlich und selbstorganisiert ab. Während eines Sprint werden bestehende Anforderungen nicht geändert oder neue aufgenommen, dies ist nur zwischen den einzelnen Sprints möglich.

Zur Abstimmung innerhalb eines Entwicklungsteams treffen sich alle Teammitglieder einmal an jedem Tag am gleichen Ort für maximal 15 Minuten. In diesen Daily-SCRUM-Meetings sollte jeder 3 Fragen beantworten:

1. Was habe ich seit dem letzten Daily-SCRUM getan?
2. Was werde ich bis zum nächsten Daily-SCRUM tun?
3. Was behindert mich?

Der SCRUM-Master ist im Idealfall bei jedem Daily-SCRUMs anwesend und achtet auf die strenge Einhaltung der 15 Minuten. In diesem Meeting sind aus Zeitgründen nur Verständnisfragen erlaubt, Diskussionen können getrennt davon im kleinen Kreis abgehalten werden. Am Ende eines Sprints präsentiert das Entwicklungsteam dem Product-Owner das Sprint-Ergebnis im Rahmen eines Sprint-Review. Aus diesem ergeben sich gegebenenfalls neue oder geänderte Anforderungen für das Product-Backlog. Nicht erledigte Sprint-Anforderungen wandern wieder zurück in den Product-Backlog-Pool. Das Sprint-Review erlaubt dem Entwicklungsteam eine Reflektion über die aufgetretenen Probleme während der Entwicklung, beispielsweise zeitlichen Fehleinschätzungen, um diese Erkenntnisse bei der Planung des nächsten Sprints einfließen zu lassen. Im Anschluss an das Sprint-Review erfolgt in der Regel unmittelbar das nächste Sprint-Planning-Meeting.

6. Erfahrungsbericht

Die Erfahrungsberichte der einzelnen Dokumente, die die Gruppenmitglieder gemacht haben, stehen jeweils am Ende eines Artefaktes. Alle Dokumente wurden von allen Teammitgliedern erstellt. Die Verteilung der Aufgaben unterschieden sich hinsichtlich der zu programmierenden Komponenten, aber nicht in den Meeting-, Protokoll-, Artefakt Erstellungs- und Projektierungsaufgaben.

Um die Artefakte zu erstellen wurden unterschiedliche Tools benötigt (UML Editoren, Programmierumgebung, Dokumentenerstellung), deren Wahl und Einarbeitung oft sehr zeitaufwendig war. Nicht alle Gruppenmitgliedern war die, durch ThyssenKrupp Europe vorgegebene, Programmiersprache C# (CSharp) geläufig und somit mussten sich die Mehrzahl der Teammitglieder darin erst zurecht finden.

Das von uns gewählte Vorgehensmodell SCRUM ist nicht Dokumenten getrieben, und somit fehlte oft der Antrieb, diese zu erstellen. Nicht allen Gruppenmitgliedern war das Vorgehen nach dem SCRUM Modell geläufig. Somit war eine gewisse Einarbeitung und Eigendisziplin erforderlich.

Teil II.

Lastenheft

Versionshistorie:

Version	Datum	Was?
0.1	20.03.2010	Angelegt
0.2	27.03.2010	Anforderungen ausformuliert
0.3	30.03.2010	Muss-Anforderungen Verfeinert
0.4	08.04.2010	Erste Überarbeitung des gesamt Textes
0.5	12.05.2010	Überarbeitet nach Peereview
0.6	15.05..2010	Letzte Mängel aus dem PeerReview beseitigt

Tabelle 10: Versionsübersicht Lastenheft

Vorgelegt am: 02.08.2010

Durch: Firma MakingTools GmbH

1. Vorwort

Das nachfolgend dargestellte Lastenheft wurde durch die fünf Mitglieder der Gruppenarbeit Simulation von Siemens DU05 Telegrammen erstellt. Im Rahmen des Lastenheftes werden die Ziele für die zu entwickelnde Software (DU05 Simulator) bestimmt, die im Pflichtenheft konkretisiert werden. Hierbei handelt es sich lediglich um die Darstellung der Programmfunktionalität, nicht aber um deren technische Umsetzung. Durch das Lastenheft soll die Komplexität des Produktes aus fachlicher Sicht des Auftraggebers widergespiegelt werden. Beim skizzierten Programm handelt es sich um ein Projekt aus der TKSE, welches als Studienprojekt im Rahmen des Master-Verbundstudie Wirtschaftsinformatik der Fachhochschulen Dortmund / Köln umgesetzt wurde. Da dieses Projekt losgelöst von der Umgebung der TKSE entwickelt werden kann eignet es sich gut für die Anforderungen einer gemeinsamen Programmentwicklung im Rahmen des Moduls „Fortgeschrittene Softwaretechnik“, da Komplexität sowie Umfang hoch sind um die gelehrt Vorgehensweisen und Techniken dieses Modules anwenden zu können. Da Auftraggeber und Auftragnehmer einer gemeinsamen Abteilung angehören, die Entwicklung somit ein abteilungsinterne Softwareprogrammierung darstellt, ist eine Abgrenzung dieser beiden Rollen praktisch nur schwer möglich. Die Haupt-

motivation von TKSE für diese Herangehensweise ist die Absicht, vorhandene Anlagen in ihrer Software-Version zu belassen auch falls Neu-Anlagen dazu kommen, um Produktionsstörungen zu vermeiden, was in die rein abteilungsinterne Zuständigkeit fällt. Auch wenn in der Praxis in diesem Fall keine strikte Trennung zwischen den Lastenheft und dem Pflichtenheft gemacht werden musste, so wurde dieses zur Anwendung eines realen, generischen Entwicklungsprozesses doch vorgenommen. In Projekten mit klar definierten Rollen für Auftraggeber und Auftragnehmer wären Lasten- und Pflichtenheft jedoch deutlich einfacher zu trennen gewesen als im Rahmen dieses Studienprojekt.

Im Folgenden werden die einzelnen Kapitel des Lastenheftes vorgestellt.

2. Zielbestimmung und Zielgruppen

2.1. Produktperspektive

In der Inbetriebnahme-Phase von Industrieanlagen wird in der Regel die TCP/IP-Kommunikation zu den Partner-Kommunikationssystemen immer dann getestet wenn ihre Entwicklung zur Neu-Anlage ebenfalls abgeschlossen ist. Dies bedeutet, dass die Entwick-

lungsabteilung der Anlagen-Lieferanten und die interne Entwicklungsabteilung der TKSE, in einem Team arbeiten müssen, dessen Koordination sich in der häufig als schwierig heraus stellt, da die Mitglieder in den meisten Fällen an verschiedenen Standorten getrennt von einander arbeiten. Dies erschwert und verkompliziert die Kommunikation und die Zusammenarbeit teilweise und kann den Projektverlauf negativ beeinflusst.

Des Weiteren werden in den Büros der Anlagenlieferanten durch die Entwicklungsabteilung bereits Kommunikationstests durchgeführt auf der Basis von Pflichtenheften der jeweiligen Industrieprojekte. Diese Vorabtests könnten deutlich verbessert und zeitlich optimiert werden, falls es eine Software gäbe, die den Verbindungsaufbau bzw. Verbindungsabbau in der Entwicklungsabteilung des Lieferanten simulieren könnte. Ist das gemeinsame Ziel einmal erreicht, müssen funktionierende Systeme geändert werden um Telegrammverläufe simulieren und testen zu können. Das Simulieren von Telegrammen in laufenden Anlagen birgt immer Risiken im Produktionsumfeld, denn bereits laufende und fehlerfrei funktionierende Software-Systeme müssen für diese Tests geändert werden. Ein Produktionsstillstand für Inbetriebnahmen werden aus wirtschaftlichen Gründen in den seltensten Fällen geplant, sondern das entstehende Risiko bewusst in Kauf genommen. Es ist bekannt, dass

auftretende Fehler sich im Produktionsbetrieb auswirken und finanziellen Schaden verursachen können. Im Extremfall kann es durch fehlerhafte oder falsch deklarierte Telegramme dazu führen das z.B. flüssiger Stahl durch beblasen mit zu hohem Druck (falsche Soll-Vorgabe) aus der Stahlpfanne spritzt und somit eine Gefahr für Leib kommen kann.

2.2. Einsatzkontext

Ziel ist es eine Software zu entwickeln die unabhängig vom Entwicklungsstand der Partner-Kommunikationssysteme (der vorhandenen Anlagen), Telegramme senden und empfangen kann. Sie muss die vorhandene Software in den produzierenden Anlagen unberührt lassen.

Hauptmotivation war es eine Software zu entwickeln, die die Möglichkeit bietet, Anlagenkomponenten oder Anlageninbetriebnahmen durchzuführen ohne vorhandene oder in Betrieb befindliche Software zu Ändern und Ortsunabhängig zu sein. So könnten die externen Entwickler die Kommunikation in ihren Büro's einsetzen und mit der noch in der Entwicklung befindlichen Anlage (z.B. Siemens SPS oder einem Kommunikationsgateway) testen. Die Kommunikationsgateways sind Rechner mit einer Software die es ermöglichen unterschiedliche Kom-

munikationsprozeduren mit einander zu verbinden.

Für eine Inbetriebnahme sollen die Anforderungen an die Software für Kommunikationstests abgedeckt werden. Dabei können Fehler oder Neu-Anforderungen durch diese Software erkannt oder analysiert werden. Auftretende Fehler, nicht umgesetzte Anforderungen oder Neu-Anforderungen können durch die Art der Kommunikationsverläufe dokumentiert werden. Kommunikationsergebnisse können gespeichert oder ausgedruckt werden.

Hauptbenutzergruppen sind die Lieferanten und die Entwicklungsabteilung der TKSE.

2.3. Zielgruppe

Die unterschiedlichen Zielgruppen (Stakeholder) setzen sich zusammen aus Anlagenbauern (Auftragnehmer), externen Entwicklern (Auftragnehmer), internen Entwicklern (TKSE), Anlagenbedienern und Produktionsabteilungen (Auftraggeber).

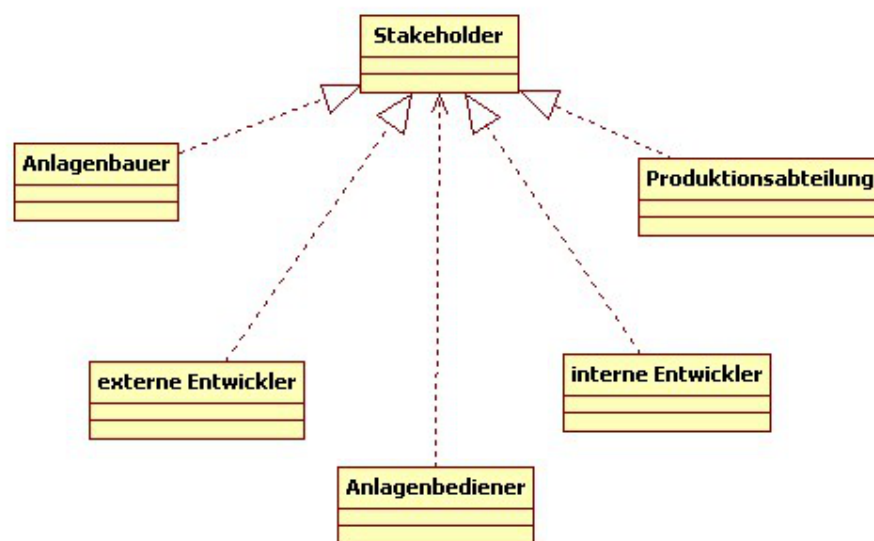


Abbildung 5: Übersicht Stakeholder

Zweck dieser Software ist folgendes:

- Vorhandene Software in produzierenden Anlagen soll nicht mehr für Inbetriebnahmezwecke geändert werden müssen, da Änderungen zu schwer wiegenden Fehlern führen können.
- eine schnelle und unkomplizierte Telegramminbetriebnahme und Tests sollen ermöglicht werden

- fehlerhafte Kommunikationsverläufe sollen dokumentiert werden können (drucken)
- Kommunikationsverläufe allgemein sollen speicherbar sein und per Mail verschickt werden
- schnelle Änderungen an Telegrammen zu ermöglichen ohne Software zu ändern

Als Stakeholder kommen alle Personen in Frage die in einem Inbetriebnahme Prozess mitwirken. Das können die Entwickler der Lieferanten sein sowie interne Entwickler der TKSE. Im Folgenden wird davon ausgegangen dass die Hauptzielgruppe Entwickler und Inbetriebnehmer sind.

Eine Nebengruppe sind die Bediener dieser neuen oder eventuell erweiterten Anlagen, auch sie werden im Schichtbetrieb sicherlich dazu aufgefordert, vordefinierte Telegramme zur Simulation auszulösen.

Durch das Grundprinzip der einfachen Bedienung soll die Hemmschwelle der nicht Inbetriebnehmer (Anlagen-Bediener) gesenkt werden und zur Bedienung nach Aufforderung animiert werden.

Innerhalb der Zielgruppe muss niemand große Erfahrung mit der Software sammeln, um diese zu bedienen zu können. Dieses Prinzip ist für Inbetriebnahmen von großer Wichtigkeit.

3. Produktdaten (Programm-Spezifische Daten)

3.1. Telegrammdefinition

LD10 Telegrammdefinition

3.2. Verbindungsparameter

LD20 IP-Adresse

LD30 Portnummer

LD40 Verbindungs-Art (aktiv/passiv)

3.3. Email

LD50 Emailadresse

3.4. Ein- / Ausgabedateien

LD60 Projekt-Parameter-Datei

Datei zu Speicherung von projektspezifischen Parametern, wie

IP-Adresse, Port und Telegrammdefinitionen. Der Inhalt muss nicht in menschen lesbarer Form vorliegen.

LD70 Kommunikationsverlauf-Datei

Datei zur Speicherung des Kommunikationsverlaufes. Die Inhalt muss in menschen lesbarer Form vorliegen.

4. Funktionale Anforderung

Die folgenden Anforderungen stellen die Basisanforderungen an die Software in Stichpunkten dar.

4.1. Produktanforderungen

- Produktdaten (Software-Spezifische Daten)
- Grafische Benutzeroberfläche
- einfache Bedienung auch für fachfremdes Personal
- Kommunikationspartner müssen frei wählbar sein
- freie Wahl der Verbindungsart (Aktiv / Passiv)
- frei definierbare Telegramme in Länge und Inhalt
- projekt- und anlagenbezogene Parameter und Definitionen

müssen dauerhaft speicherbar sein

- senden von Telegrammen zu Kommunikationspartnern
- Empfang von Telegrammen von Kommunikationspartnern
- Protokollierung des Kommunikationsverlaufes
- drucken des Kommunikationsverlaufes
- speichern des Kommunikationsverlaufes
- E-Mail-Versand des Kommunikationsverlaufes

Ohne Kommunikations-Gegenpartner ist die Software nur bedingt einsetzbar. Hier könnten lediglich, für die Inbetriebnahme, vorbereitenden Tätigkeiten ausgeführt werden, z.B. Vereinbarte Telegramme definieren, E-Mail Adressen einpflegen und/oder IP-Adressen vergeben. Beherrscht der Kommunikations-Gegenpartner nicht die Befehle einer Siemens DU05 Prozedur (wie Telegrammaufbau, LIVE-Telegrammdefinition, verhalten bei Fehlerhaften Telegrammen, usw...) ist die Software nicht einsetzbar.

4.2. Muss-Anforderung

Die in Kapitel 4.1 aufgeführten Anforderungen stellen die wesentlichen Hauptfunktionen (Anforderungen) des Simulators da.

4.3. Allgemeine Beschreibung der Anforderungen

Die Software muss zu Standard-Siemens-DU05-Kommunikationspartnern eine TCP/IP Verbindung aufbauen können. Die Siemens DU05 Prozedur muss mit möglicherweise auftretenden Kommunikationsfehlern weiterhin eine Verbindung zum Partner-System halten oder auch nach schwerwiegenden Fehlern wieder neu aufbauen können.

Die Parameter zum Aufbau einer Verbindung müssen dauerhaft gespeichert werden können. Des Weiteren muss die Art der Verbindung -aktiv oder passiv- parametrierbar sein, d.h. dass die Software bei einer Aktiven Verbindung eigenständig versucht die Verbindung zum, Partner aufzubauen, während eine passive Verbindung wartet, bis eine Verbindungs-Anforderung kommt. Ein einmal definierter Kommunikationsparameter muss jederzeit verändert werden können.

Unterschiedliche Telegramme müssen komfortabel definierbar sein, wobei die Länge eines Telegramms variable definiert wer-

den kann. Jedoch gibt es Anlagen, in denen eine Feste Telegrammlänge vorgegeben ist. Hier ist es von Vorteil, wenn über ein Parameter ein Feste Länge des Telegramms automatisch generiert werden kann, d.h. dass ein Telegramm welches nur einen Teil der Telegrammlänge mit Daten verbraucht hat, die restlichen mit 0'en aufgefüllt bekommt. Die Telegramme müssen in einem ASCII-Format gesendet und empfangen werden. Die Länge eines Telegramms wird in einem zwei Byte Telegrammkopf im Siemens Zwei-Byte (ein Wort) Format voran gestellt. Zu beachten ist, dass das High-Byte nicht an der gleichen Stelle sitzt wie in einem Intel-Format, sondern die Bytes müssen vertauscht (SWAP-Byte) werden.

Der Kommunikationsverlauf muss in einem Fenster in Quasi-Echtzeit auf dem Rechnersystem sichtbar sein um den zeitlichen Verlauf oder Fehler in der Kommunikation jederzeit und ohne Zeitverlust erkennen zu können.

Die definierten Telegramme müssen dauerhaft gespeichert werden können. Einmal definierte Telegramme müssen gelöscht oder bearbeitet werden können um Fehler zu korrigieren oder Erweiterungen der Telegrammdefinitionen vornehmen zu können.

Die gesamten Einstellungen des Programms (im Wesentlichen

Kommunikationsparameter und Telegrammdefinitionen) müssen in unterschiedliche Projekt-Dateien gespeichert werden können. Die ist gerade für die erste Programmversion wichtig, damit durch mehrfach Start des Programms die Verbindung zu unterschiedlichen Kommunikationssystemen gleichzeitig aufgebaut werden kann, indem die unterschiedlichen Projektdateien geladen werden. Spätere Programmversionen sollen in einer Instanz Verbindungen zu mehreren Partnern erlauben.

Die Anzahl der gesendeten und empfangenen Telegramme soll in einer übersichtlichen Art und Weise angezeigt werden, um den Kommunikationsverlauf (Telegrammverkehr) besser verfolgen zu können. Des Weiteren muss die Verbindungszeit überwacht werden, um Kommunikationsabbrüche, bedingt durch fehlerhafte Telegramme, besser erkennen zu können und das zeitliche Verhalten in diesen Fehlersituationen jederzeit zu rekonstruieren.

Die gesendeten und empfangenen Telegramme, die im Fenster angezeigt werden, müssen auf einem Drucker ausgegeben werden können, so dass das Kommunikationsverhalten dauerhaft dokumentieren und an die entsprechenden Entwickler weitergegeben werden kann. Als weitere Option zur Weiterleitung der gesammelten Daten sollen die gesendeten und empfangenen Telegramme als lesbarer ASCII-Text exportiert werden

können, so dass der Telegrammverkehr ohne weitere Kodierung per Mail an die Entwickler verschickt werden kann.

4.4. Präzise Beschreibung der Anforderungen

4.4.1. Grafische Benutzeroberfläche

- **AF10 Grafische Oberfläche zur Benutzung des Programms**

Das Programm soll über eine grafische Benutzeroberfläche (GUI) bedienbar sein und den aktuellen Standards der Window-basierten Programmierung folgen um eine möglichst intuitive Bedienung durch Benutzer zu ermöglichen, die den Umgang im fensterorientierten Anwendungen und Betriebssystemen gewöhnt sind. Der Kommunikationsverlauf muss in einem window-basierten Teil-Fenster sichtbar sein.

- **AF10.1 Bedienung auch für fachfremdes Personal**

Die Software soll generell intuitiv und einfach bedienbar sein damit sie auch von fachfremdem Personal genutzt werden kann. Anlagenbediener sollten durch die einfache Bedienung in die Lage versetzt sein, auch ohne Fachkenntnisse Anweisungen eines Inbetriebnehmers auszuführen.

- **AF10.2 Dialog zur Eingabe der Parameter zum Verbin-**

ungsaufbau

Die Eingabe der IP-Adresse und der TCP-Ports als obligatorische Verbindungsbeschreibung zu Partner-System muss über einen Dialog geschehen. Des Weiteren muss im Dialog die Möglichkeit bestehen die Verbindungs-Art, Aktiv oder Passiv, festzulegen. Für die IP-Adresse und den Port gilt eine Einschränkung: Die Software darf zu einem bestimmten Rechner **keine** Verbindung aufbauen, dessen IP-Adresse kann im Quellcode fest eingetragen sein. Ist sie anpassbar muss ein Schutz vor einer Änderung durch nichtberechtigte Personen vorhanden sein. Es sollten nur Ports benutzt werden die von Microsoft nicht durch Standards belegt sind.

- **AF10.3 Dialog zur Definition von Telegrammen**

Die Eingabe von Telegrammen soll über einen Dialog in einem eigenen Fenster geschehen. Die Länge sowie der Inhalt müssen frei wählbar sein. Die Definition von mehreren Telegrammen muss möglich sein.

Es gibt Sonderfälle, in denen die Telegrammlänge fest auf eine Byte-Anzahl begrenzt oder aufgerundet werden muss. In diesem Fall ist eine Möglichkeit zu schaffen, die die Telegrammlänge, trotz eventuell kürzerem Telegramminhalt, auf eine frei definierbare Länge setzt. Dem Benutzer soll

bei der Eingabe des Telegramms dessen aktuelle Länge angezeigt werden.

- **AF10.4 Dialog zur Definition von E-Mail Adressen**

Die Eingabe der E-Mail-Adressen zur Kommunikation mit externen Entwicklern oder Inbetriebnehmer sollten in einem Dialog erfolgen. Mehrere E-Mail-Adressen müssen eingegeben werden können.

- **AF10.5 Dialog zum versenden von E-Mails**

Zum Versenden von Kommunikationsverläufen muss die Möglichkeit bestehen, über einen "E-Mail-Dialog" Daten an die externen Inbetriebnehmer oder Entwickler zu senden.

4.4.2. Kommunikation zum Kommunikationspartner

- **AF20.1 Socket Verbindung über TCP/IP**

Die Software muss in der Lage sein einen Verbindungsaufbau zu einem geeigneten Verbindungspartner über eine Socket TCP/IP Schnittstelle herzustellen.

- **AF20.2 Siemens DU05 Prozedur mit dem Kommunikationspartner**

Die Kommunikationsprozedur, die auf TCP/IP aufsetzt, muss die Siemens DU05 Prozedur sein. Diese ist bei TKSE Stan-

dard und ist Voraussetzung zum Aufbau einer Verbindung mit einem Kommunikationspartner. Prozedurverstöße sind abzufangen und zu behandeln und dürfen nicht zu einem nicht definierten Verhalten der Kommunikationssoftware führen.

- **AF20.3 freie Wahl des Kommunikationspartner**

Der Kommunikationspartner, definiert durch IP-Adresse und TCP-Port-Nummer, müssen zur Verbindung mit dem Partner frei wählbar sein, sie dürfen nicht im Quellcode fest verankert sein.

- **AF20.4 Festlegung der Verbindungs-Art Aktiv**

Die Kommunikationssoftware muss in der Lage sein einen Verbindungsaufbau zum Partner-System *aktiv* durchzuführen, d.h. die zuvor parametrisierten Informationen sollen aktiv genutzt werden um den Partner im Kommunikationsnetz zu kontaktieren.

- **AF20.5 Festlegung der Verbindungs-Art Passiv**

Die Kommunikationssoftware muss in der Lage sein, bei Wahl einer *passiven* Verbindung auf das Anfragen von Kommunikationspartnern zu „lauschen“. Dies bedeutet dass nur die zuvor parametrisierten IP-Adresse und der vorher definierte Port als Verbindungspartner Akzeptiert wird.

4.4.3. Telegrammdefinition

- **AF30.1 Länge des Telegramms**

Das Telegramm welches zu Inbetriebnahme Zwecke definiert werden soll, muss in seiner Länge frei wählbar sein.

- **AF30.2 LIVE Telegramme**

Das LIVE-Telegramm, welches zu Überwachung der Kommunikationsverbindung dient, ist im Programm fest verdrahtet. Das ausbleiben eines LIVE-Telegramms muss einen Verbindungsabbruch erzwingen, da davon ausgegangen wird dass ein Kommunikationsfehler vorliegt. Nach einem solchen Ereignis muss die Software einen erneuten Verbindungsaufbau durchführen.

- **AF30.3 Quittungstelegramme**

Der Aufbau eines positiven und eines negativen Quittungstelegramms sind in der Kommunikationssoftware fest verdrahtet. Diese werden benötigt, um empfangene oder gesendete Telegramme bei Fehler erneut anzufordern und den fehlerfreien Erhalt eines Telegramms zu bestätigen.

4.4.4. Dauerhafte Speicherung von Projekt-Parametern

- **AF40.1 Speichern von Projekt-Parameter**

Die Software muss in der Lage sein, projektspezifische Parameter, wie IP-Adresse, Port und Telegrammdefinitionen, projektbezogen auf ein Medium wie Festplatte, USB-Stick oder Diskette zu speichern.

- **AF40.2 Laden von Projekt-Parameter**

Die Software muss in der Lage sein, projektspezifische Parameter, wie IP-Adresse, Port und Telegrammdefinitionen, projektbezogen von einem Medium wie Festplatte, USB-Stick oder Diskette in die Programmumgebung zu laden.

4.4.5. Kommunikation

- **AF50.1 Bereitstellung von Kommunikationsverlaufsinformationen**

Die Kommunikationsschnittstelle muss die Möglichkeit bieten ausgeführte Aktionen als Klartext (ASCII-basiert) über eine Schnittstelle einem Visualisierungsprogramm zur Verfügung zu stellen um die Informationen dem Benutzer anzeigen zu können.

- **AF50.2 Senden von Telegrammen**

Die Software muss über eine Möglichkeit verfügen die zuvor definierten Telegramme an das Partner-System zu senden. Alle definierten Telegramme sollen in einem Zyklus von 2 Sekunden an den Partner versendet werden. Des Weiteren muss die Software in der Lage sein, Quittungstelegramme der zuvor gesendeten Telegramme auszuwerten und darauf entsprechend zu reagieren sowie Quittungstelegramme selbständig zu generieren und zu senden.

- **AF50.3 Empfangen von Telegrammen**

Die Software muss über eine Möglichkeit verfügen, die Antworttelegramme auf die zuvor gesendeten Telegramme vom Partner-System zu empfangen. Des Weiteren muss sie in der Lage sein empfangene Telegramme statusabhängig zu quittieren und ein Quittungstelegramm selbständig zu senden.

4.4.6. Informationsweitergabe

- **AF60.1 protokollieren des Kommunikationsverlaufes**

In einem geeigneten (Teil-)Fenster soll der Kommunikationsverlauf angezeigt und protokolliert werden um Unregelmäßigkeiten in der Kommunikation anzuzeigen.

- **AF60.2 drucken des Kommunikationsverlaufes**

Der Kommunikationsverlauf soll auf einem Ducker ausgegeben werden können. Telegramme die zu lang sind, müssen mit einem Zeilenumbruch versehen werden. Des Weiteren muss die Druckerfunktion in der Lage sein, lange Kommunikationsverläufe auf mehrere Seiten zu drucken. Die Seiten müssen fortlaufend nummeriert sein.

- **AF60.3 speichern des Kommunikationsverlaufes**

Der Kommunikationsverlauf sollte dauerhaft auf ein Speichermedium wie USB, HDD oder FDD gespeichert werden können.

- **AF60.4 E-Mail Versand des Kommunikationsverlaufes**

Der Kommunikationsverlauf muss per E-Mail verschickt werden können.

4.4.7. Programm-Handling

- **AF70.1 Drucken während Kommunikationsverlauf nicht erlaubt**

Während einer aktiven Verbindung, darf aus dem Programm nicht gedruckt werden, da dies zu Performance Einbrüchen führen kann.

- **AF70.3 Speichern allgemein während Kommunikations-**

verlauf nicht erlaubt

Während einer aktiven Verbindung, darf aus dem Programm nicht gespeichert werden, da dies zu Performance Einbrüchen führen kann.

- **AF70.4 E-Mail Versand während Kommunikationsverlauf nicht erlaubt**

Während einer aktiven Verbindung, darf aus dem Programm nicht gemailt werden, da dies zu Performance Einbrüchen führen kann.

- **AF70.5 IP-Adresse definieren während Kommunikationsverlauf nicht erlaubt**

Während einer aktiven Verbindung, darf die IP-Adresse und der Port nicht geändert werden, da eine aktive Verbindung zum Partner besteht.

4.5. Kann-Anforderung

Die nachfolgenden Anforderungen sind für einen erste Version nicht relevant.

4.5.1. AF70 Validierung des Telegramms

Die Telegramme sollen auf einen korrekten internen Aufbau hin überprüft werden. Bei nicht Einhaltung von einer vorgegebenen Telegrammkopf Gestaltung sollen dem Anwender Informationen über den Fehler und die fehlerhaften Zeichen mitgeteilt werden.

4.5.2. AF80 Validierung IP Adresse

Nach Eingabe einer IP-Adresse soll das Programm die IP Adresse validieren und bei einem Verstoß gegen die Konvention eine Fehlermeldung anzeigen.

4.5.3. AF90 Validierung des Ports

Nach Eingabe eines Ports soll die Software den Port auf Gültigkeit und Verfügbarkeit hin prüfen. Des Weiteren sollte kontrolliert werden ob der verwendete Port durch einen Well-Known

Port kollidieren könnten (0 bis 1023).

4.5.4. AF100 Senden und empfangen mit verschiedenen Partnern über das gleiche Programm

Das Programm sollte innerhalb einer Instanz mehrere Verbindungen zu verschiedenen Kommunikationspartnern parallel aufbauen könnte. Die Protokollierung der Unterschiedlichen Verbindungen müssten dann über Numerische Kennzeichen erkennbar sein.

4.5.5. AF110 Telegramm Analyse (Nettodaten verifizieren)

Die eingegebenen Telegramme sollten mit einem Telegramm-muster verglichen werden können und die Stelle des Fehler in der Definition anzeigen. Auch empfangenen Telegramme sollte auf ihren Inhalt analysiert werden können, um schneller Fehler in den Telegrammen erkennen können.

4.5.6. Nähere Beschreibung

Es kann erforderlich sein, dass die Kommunikation zu unterschiedlichen Partner-Systemen gleichzeitig getestet werden muss. Dazu muss das Programm bereits in der ersten produktiv ein-

setzbaren Version einsatzfähig sein, also mehrfach parallel mit unterschiedlichen Parametern gestartet werden können ohne dass sich die Instanzen gegenseitig beeinflussen. Es wäre von Vorteil wenn die Kommunikationsbeziehungen alle in einem Programm verwaltet werden könnten. Diese Anforderung muss aber in der ersten Version noch nicht zur Verfügung stehen.

Das damit auftretende Problem der Zuordnung der definierten Telegramme zu ihren Kommunikationspartnern kann in einer weiteren Version in Eigenarbeit der TKSE realisiert und erweitert werden.

Es wäre wünschenswert, gefilterte Fehler- und Ereignismeldungen in einem Status-(Teil-)Fenster anzuzeigen. Hierzu müsste ein weiterer Dialog geschaffen werden, der die Filter-Auswahl der Meldungen an die Kommunikationsschnittstelle weitergibt und diese im Fenster anzeigt. Es ist dem Entwicklungsteam zu überlassen, ob diese Anforderung in den zeitlichen Rahmen passt.

5. Nicht-Funktionale Anforderung

In der ersten Version muss das Programm nur eine Verbindung zu, *einem* Partner und zu einem Zeitpunkt aufbauen kön-

nen. Dies ist für die Skalierbarkeit und Performance ein Vorteil, da der Kommunikationsfluß und die Belastung des Rechners am Anfang nicht vorhersehbar ist. Da gewährleistet sein muss, dass die Protokollierung funktioniert auch wenn übermässig viele Telegramme kommen, ist die Wahl des Einzel-Starts für die erste Version effektiver.

Die Bedienbarkeit sollte so gehalten werden, dass auch ein Nicht-Entwickler das Programm bedienen kann. Die Anlagenbauer, die eine Reaktion in ihrer Anlage herbeiführen möchten, sind in der Regel keine Software-Entwickler und nicht zwangsläufig erfahren in der Bedienung von Computern und Computerprogrammen, müssen jedoch trotzdem Telegramme senden können. Die Hauptmotivation hierfür ist, dass auch im Schichtbetrieb Anlagen in Betrieb genommen werden müssen, aber des Nachts nicht beliebig geschultes Personal vor Ort vorhanden ist. Eine Inbetriebnahme sollte mangels in der Bedienung von Computerprogrammen unerfahrenem Personal nicht verzögert werden.

An die Optik der Programmoberfläche werden über die genannten Punkte hinaus keine Anforderungen gestellt.

5.1. NF10 Einfachheit des Programms

Das Simulationsprogramm soll einfach zu bedienen und die Bedienung selbst sollte selbsterklärend sein. Geeignete und allgemein bekannte Fensterkontrollelemente (Windows-Control) nach dem Microsoft-Windows-Standard, sollen hier zum Einsatz kommen, um die Benutzung zu erleichtern.

5.2. NF20 Programmierung in CSharp

Aus Gründen des Unternehmensstandards und der Unternehmens-Richtlinien, ist das Programm mit dem Entwicklungstool Microsoft Visual Studio 2008 unter Verwendung der Programmiersprache C#¹ zu entwickeln. **Gehört das eventuelle zu den Technischen Anforderungen?**

5.3. NF30 Qualitätsanforderung

5.3.1. Softwarequalität

Zur Definition von zu erreichenden Qualitätszielen muss der Begriff der Softwarequalität präzisiert werden. Dazu werden Qualitätsmerkmale definiert, deren Gesamtheit die Qualität der

¹gesprochen ci-ssharp

Software ausmacht. Eine solche Aufschlüsselung von Softwarequalitätsmerkmalen liefert unter anderem die ISO Norm 9126. Diese ISO Norm definiert die Qualitätsmerkmale die im Folgenden vorgestellt werden:

- Übertragbarkeit -> Anpassbarkeit und Installierbarkeit
- Anwendbarkeit -> Erlernbarkeit, Beherrschbarkeit und Verständlichkeit
- Effizienz -> raum-, zeit- und ressourcenbezogen
- Funktionalität -> Angemessenheit, Interoperabilität und Genauigkeit
- Zuverlässigkeit -> Fehlertoleranzen, geringe Fehlerhäufigkeit und Fehlerverfolgbarkeit
- Wartbarkeit -> Analysierbarkeit, Änderbarkeit, Stabilität und Testbarkeit

Nicht jeder dieser Punkte wird von der Software tangiert. Im Folgenden wird auf diejenigen Punkte eingegangen die von der Software zu realisieren sind.

5.3.2. Übertragbarkeit

Der Simulator sollte ohne administrative Rechte auf jedem Rechner installierbar sein, da Inbetriebnehmer oft auf Ihren Rechnern keine Rechte zur Installation von Programmen haben. Der Simulator sollte in der Lage sein jede Art von Inbetriebnahmen im Bereich Siemens DU05 Prozedur basierend auf TCP/IP Kommunikation und Telegrammtests durchzuführen.

5.3.3. Anwendbarkeit

Der Simulator muss ohne große Vorkenntnisse bedienbar sein. Dialoge sollten zur leichteren Bedienbarkeit gleich aufgebaut sein, also die gleichen Fensterkontrollelemente nutzen. Auf eine Menüleiste ist der Übersichtlichkeit halber zu verzichten und stattdessen zur einfacheren Bedienung eine Symbolleiste mit leicht erkenn- und zu den Funktionen zuordnungsbare Symbole einzusetzen. Eventuell können Tastenkürzel die Bedienung vereinfachen.

5.3.4. Effizienz

Das Reagieren auf empfangene Telegrammen, oder auch auf Antwortzeiten von Quittungstelegrammen unterliegen einem ge-

nau definierten Zeitverhalten. Daraus resultieren eventuell Verbindungsabbrüche (Prozedurverstöße) oder zu versendende Lebenstelegramme (sogenannte Live-Telegramme²). Das Zeitverhalten von zu senden Telegrammen ist fest auf 2 Sekunden eingestellt, d.h. das jedes einzelne definierte Telegramm alle zwei Sekunden verschickt wird bis die Liste der definierten Telegramme abgearbeitet ist. Während der Simulator verbunden ist und Telegramme mit dem Partner austauscht, muss die Bedienung des Programms jeder Zeit gewährleistet sein, d.h. auch bei hohem Kommunikationsaufkommen muss die Priorität auf dem Zeitverhalten der DU05 Prozedur liegen genauso wie auf die Bedienbarkeit des Simulators.

5.3.5. Wartbarkeit

Das Programm sollte modular aufgebaut werden um eine leichte Wartbarkeit und Änderbarkeit zu gewährleisten.

5.3.6. Zuverlässigkeit

Das Programm darf bei Kommunikationsproblemen wie Prozedurverstößen, Netzwerkproblemen, dem Ausbleiben von LIVE-

²Live-Telegramme kontrollieren eventuell nicht gemerkte Verbindungsabbrüche durch zwischengeschaltete Router oder Switches im Netzwerkbereich. Sie werden in einem bestimmten Zeitzyklus gesendet und müssen positiv quittiert werden.

Telegrammen, Steuerzeichen innerhalb eines empfangenen Telegramms und Fehlbedienungen des Programms nicht abbrechen. Auch fehlende Bedingungen für den korrekten Funktionsablauf dürfen nicht zum Abbruch des Programms führen, stattdessen sollen die Fehler abgefangen und an den Benutzer gemeldet werden.

5.3.7. Sicherheit

Der Punkt Sicherheit hat in diesem Kontext eine besondere Bedeutung. Im Produktionsbetrieb kann es durchaus einer Gefahr für Leib und Leben kommen, da Telegramme komplette Anlagen direkt und unmittelbar steuern. Telegramme die mit dieser Software definiert wurden, dürfen *nur* nach dem Siemens DU05 Prozedur Standard versendet werden. Im Fehlerfall, beispielsweise bei Prozedurverstöße, Telegrammlängenverstößen und Inkonsistenzen in den Plausibilitätsprüfungen, müssen Telegramme verworfen und die Verbindung abgebaut und wieder neu aufgebaut werden.

5.4. Technische Anforderung und Einschränkungen

Die Software sollte so entwickelt werden, dass sie auf Microsoft Windows Systemen ab Windows XP und vorhandenem Micro-

soft Framework 3.5 lauffähig ist. Des Weiteren sollte sie einfach zu installieren oder im Idealfall komplett ohne Installation lauffähig sein, so dass auch Anlagenbauer diese Software auf Ihren mitgebrachten Geräten installieren können. Falls eine Installation nötig ist sollte sich diese darauf beschränken, dass die Softwarekomponenten einfach in ein Verzeichnis kopiert werden, ohne in der Registry oder in Config-Dateien Einträge oder Änderungen gemacht werden müssen. Dies erlaubt den Einsatz auf betrieblich genutzten PC Systemen, die von deren Benutzern nicht mit Software erweitert werden dürfen.

5.4.1. Lizenzen

Da diese Software nicht kommerziell eingesetzt wird, sind Lizenzierungsverfahren und Produktregistrierung hier nicht zu berücksichtigen und nicht zu implementieren.

6. Architektur des Gesamtsystems

Die zu realisierende Software ist unter Zuhilfenahme des kostenlosen Microsoft® Visual Studio Express 2008 Entwicklungsumgebung zu erstellen. Der Hauptgrund für die Vorgabe einer Entwicklungsumgebung ist der Umstand, dass die Software auch nach Abschluss des Studienprojekts weiterhin von Mitarbeitern der TKSE gewartet, eventuell angepasst und erweitert werden muss.

Ein modularer Aufbau einzelnen Teilfunktionalitäten ist wünschenswert um eine einfache Wartbarkeit und Erweiterbarkeit auch bei wechselnder Betreuung durch unterschiedliche Programmierung zu ermöglichen.

Vereinfachte Darstellung des Gesamtsystems.



Abbildung 6: Abbildung Gesamtübersicht

7. Skizzen der Benutzeroberfläche

Die Skizzen der Benutzeroberfläche:

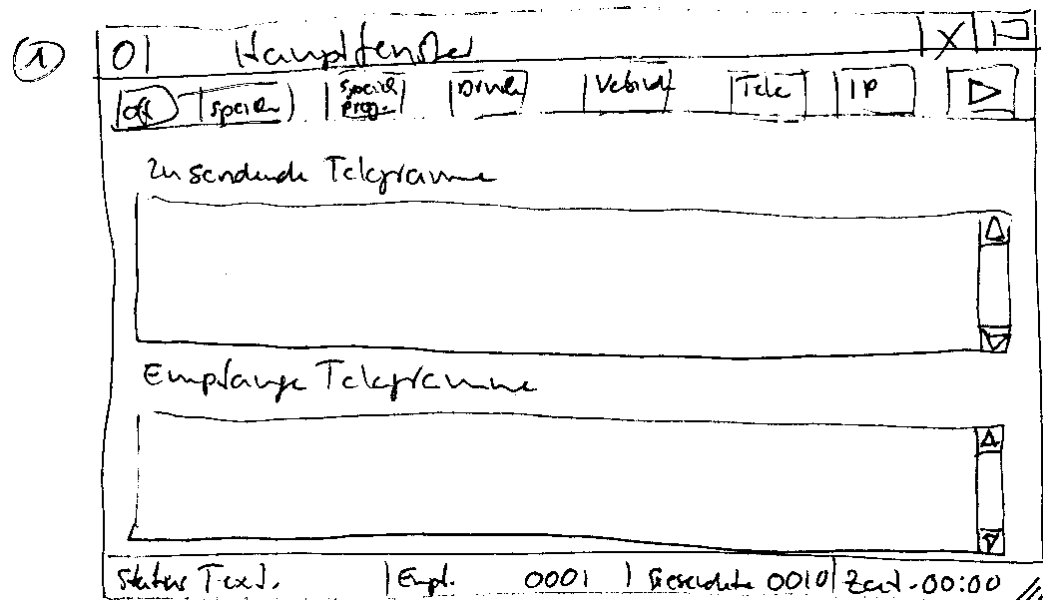


Abbildung 7: Skizze des Hauptfensters

Die Skizzen des IP Parameter Dialoges:

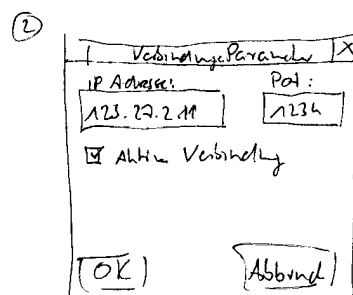


Abbildung 8: Skizze IP Kommunikations Dialog

Die Skizzen des Telegrammdefinitions-Dialoges:

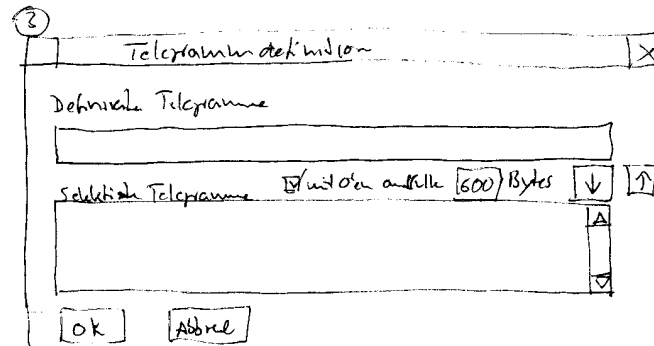


Abbildung 9: Skizze Telegrammdefinitions-Dialog

Die Skizzen des E-Mail Adressen-Dialoges:

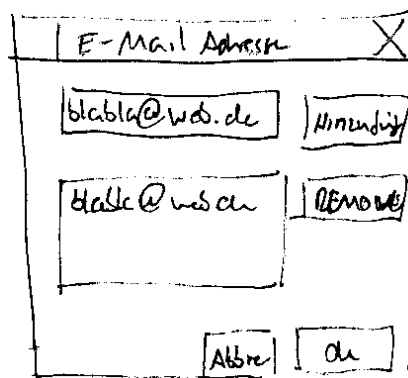


Abbildung 10: Skizze E-Mail Adressen Dialog

Die Skizzen des E-Mail Senden-Dialoges:

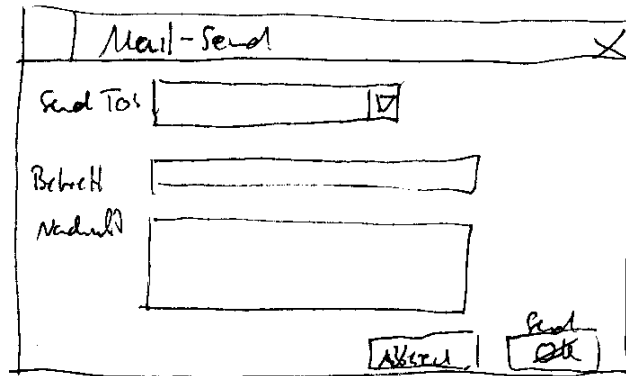


Abbildung 11: Skizze E-Mail senden-Dialog

8. Abnahmekriterien

Folgende Kriterien sind Voraussetzung für eine Abnahme des Produkts:

- Umsetzung aller Muss-Anforderungen wie in Kapitel 4.2 beschrieben
- Umsetzung aller nichtfunktionalen Anforderungen wie in Kapitel 5 beschrieben
- optionale Umsetzung der Kann-Anforderungen wie in Kapitel 4.5 beschreiben
- angestrebte Qualität und Bedienbarkeit des Produktes
- Vollständige Dokumentation des Produkts
- Einhaltung des vereinbarten Zeitrahmens
- Übergabe des Produkts inklusive dokumentiertem Source-code und Dokumentation
- Vorstellung und Demonstration der Software innerhalb der TKSE

9. Erfahrungsbericht (Fazit)

Nachdem dass Projekt feststand, und die Gruppenmitglieder in das Thema eingearbeitet waren, wurde begonnen das Lastenheft zu erstellen. Hier war es immer wieder erforderlich, das Produktionsumfeld der TKSE zu erklären und den Teammitgliedern näher zu bringen um Verständnis für gewisse Anforderungen zu entwickeln. Denn die Arbeitsbereiche der einzelnen Mitglieder, liegen außerhalb von Produktionsumgebungen.

Nach unzähligen Gesprächen, wurden die Anforderungen ermittelt. Dies erwies sich als äußerst schwierig, da sich die Anforderungen immer wieder durch den Betrieb änderten oder neue hinzu kamen. Dazu kam noch, dass die Kommunikation zum Kunden (dem Betrieb) nur über eine Person laufen konnte, da dieser im Betrieb ansässig ist. Die Schwierigkeiten bei der Erstellung des Lastenheftes lagen in erster Linie in der Einarbeitung und den Umgang mit dieser, für uns neuen, Vorgehensweise zur Entwicklung von Software. Des Weiteren war es nicht immer einfach die Inhalte zu den, noch zu erzeugenden, Artefakte klar zu trennen.

Es ist festzustellen, dass bei der Einarbeitung in das Projekt und in die Erstellung des Lastenheftes die Trennung zwischen dem Lasten- und dem Pflichtenheft sich oft als sehr schwierig

erwies. Erst mit der ständigen Reflexion mit dem Thema und der Besuch der Präsenzveranstaltungen wurde es immer klarer.

Teil III.

Pflichtenheft

Versionshistorie:

Version	Datum	Was?
0.1	20.03.2010	Einstellung des Dokumentes
0.2	27.03.2010	Produktfunktionen eingearbeitet
0.3	30.03.2010	Use Cases eingearbeitet
0.4	13.05.2010	Bereinigung des Dokumentes
0.5	18.05.2010	Korrekturen nach PeerReview
0.6	XX.XX.2010	Angelegt

Tabelle 11: Versionsübersicht Pflichtenheft

1. Allgemeines

1.1. Sinn und Zweck des Dokumentes

Dieses Dokument beruht auf dem Lastenheft und gibt Auskunft darüber, welche Anforderungen an das Simulationsprogramm sowie an dessen Elemente gestellt werden. Hierzu werden die spezifischen Anforderungen definiert und in einer strukturierten Weise in diesem Dokument zusammengefasst.

1.2. Zielbestimmung

Der DU05 Simulator ist ein auf Windows basiertes Programm zum Simulieren von TCP/IP Telegrammen im betrieblichen Umfeld der ThyssenKrupp Steel Europe AG.

In diesem Zusammenhang werden die Funktionalitäten entworfen die im Rahmen von Erweiterungen und/oder der Inbetriebnahme von Neu-Anlagen benötigt werden. Dazu zählen Definition von Telegrammen, Parametrierung von Verbindungseinstellungen, Benutzerverwaltung für E-Mail / SMTP-Server, Kommunikationsaufbau und Protokollierung des Kommunikationsverlaufes. Der Kernbereich ist aber das Senden und empfangen von Telegrammen. Hierzu müssen Telegramme definiert

und Verbindungs-Parameter eingegeben werden. Nach erfolgreicher Definition und Parametrierung kann der Kommunikationsverlauf getestet werden. Während der Verbindungslaufzeit dürfen Telegramme geändert, doch die Verbindungs-Parameter dürfen nicht geändert werden, da dies einen Verbindungsabbruch hervorrufen würde.

An das Gesamtsystem werden zusätzlich Qualitätsanforderungen und technische Anforderungen für die Entwicklungsumgebung definiert. Anforderungen die keinem technischen Element zugeordnet werden können, werden als allgemeine Anforderungen festgehalten.

1.3. Zielgruppen

Die Personen der Zielgruppen sind Mitarbeiter der Thyssen-Krupp Steel Europe AG Standort Duisburg, Abteilung Betriebsleitsysteme (Softwareentwicklung), alle Inbetriebnehmer die Anlagen im Bereich des TBM-Konverters³ installieren und Zulieferunternehmen die Kommunikationssoftware für Neu-Anlagen bei der TKSE erstellen. Des Weiteren zählen zu der Zielgruppe die Mitarbeiter in den Anlagen (Leitstand-Personal) die diese

³TBM Konverter ist ein Gefäß mit einem Fassungsvermögen von mehreren Hundert Tonnen, welches Roheisen zu Rohstahl wandelt, indem Sauerstoff eingeblasen wird und stahlerzeugende Materialien eingefüllt werden.

Software zum Teil bedienen soll. Durch die E-Mail Anbindung zum SMTP Server zählt auch diese Abteilung dazu.

2. Produktdaten

2.1. Telegrammdefinition

LD10 Telegrammdefinition

- Anzahl: n
- max. Länge: ohne Begrenzung
- Typ: String
- erlaubte Zeichen: ASCII-Zeichen von A-Z, 0-9 und ' ' und '-' Zeichen für Vorzeichen behaftete numerische Werte. z.B. negative Werte = -100 positive Werte: 100

2.2. Verbindungsparameter

LD20 IP-Adresse

- Anzahl: 1
- max. Länge: 15 Zeichen

- Typ: String
- erlaubte Zeichen: ASCII-Zeichen von 0-9 und ".'

LD30 Portnummer

- Anzahl: 1
- max. Länge: 5 Zeichen
- Typ: Integer
- erlaubte Zeichen: ASCII-Zeichen von 0-9
- Plausibilität: Wert größer 0

LD40 Verbindungsart (aktiv/passiv)

- Anzahl: 1
- Typ: Boolean

2.3. Email

LD50 Email-Adresse

- Anzahl: n
- max. Länge: 254 Zeichen
- Typ: String

- erlaubte Zeichen: Local Part: ASCII-Zeichen von A-Z, 0-9, (RFC 2822)

2.4. Ein- / Ausgabedateien

LD60 Projekt-Parameter-Datei Datei zu Speicherung von projektspezifischen Parametern. Pro Datei werden folgende Parameter abgespeichert:

- 1 x IP-Adresse
- 1 x Portnummer
- 1 x Verbindungsart (aktiv/passiv)
- n x Telegrammdefinitionen
- n x Email-Adressen. Die Parameter werden serialisiert in einer Projekt-Datei (*.prj) abgelegt

LD70 Kommunikationsverlauf-Datei

Datei zur Speicherung des Kommunikationsverlaufes. Der Inhalt soll möglichst exakt wie im Programmausgabefenster lesbar sein. (Datum, Uhrzeit, Infomeldung). Die Inhalte werden als ASCII basierte Textdatei (*.txt) abgespeichert, da diese Log-Dateien auf jedem Betriebssystem mit einem einfachen Editor lesbar sein sollen. Dazu soll nicht der DU05 Simulator auf die-

sem System installiert werden, nur die Datei soll zu auswerte Zwecken bearbeitbar sein.

3. Funktionale Anforderung

3.1. Anwendungsfälle

3.1.1. Grafische Benutzeroberfläche

- **AF10 Grafische Oberfläche zur Benutzung des Programms**

Das Programm sollte eine grafische Oberfläche bereitstellen und alle Möglichkeiten der Fenster basierten Programmierung nutzen. Das Fenster muss in der Höhe und Breite verstellt werden können. Der Kommunikationsverlauf sollte im einem Window-Basierten Fenster sichtbar sein. Hierzu müssen eine Symbolleiste und Statusleiste definiert werden. Über die Symbolleiste wird das Programm bedient. Jedes Symbol auf der Symbolleiste repräsentiert eine Funktion die über ein entsprechendes Icon angezeigt wird. Dadurch dass die Software von unterschiedlichem Benutzern (auch TKSE Fremde Benutzer wie Lieferanten) bedient wird, und das Programm eine sehr einfache Bedienung hat, wird hier aus Übersichtlichkeitsgründen auf eine Menüleiste verzichtet. Die Telegramme die im Telegrammdefinitions-Dialog

definiert wurden, müssen nach positivem verlassen des Dialoges in einem scrollbaren Windows-Control angezeigt werden. Der Kommunikationsverlauf soll ebenfalls in einem scrollbaren Windows-Control angezeigt werden. Diese beiden scrollbaren Windows-Control müssen sich in dem Hauptfenster befinden, damit sie jederzeit kontrolliert werden können. Alle Windows-Controls müssen bei Größenänderung des Hauptfensters mit verändern. Die Dialoge zur Definition von Telegrammen, Parametrierung der Verbindung, Verwaltung der E-mail Adressen und der Dialog zum senden von E-Mails müssen über die Symbolleiste aufgerufen werden. Während einer aufgebauten Verbindung dürfen die Telegrammdefinitionen jederzeit geändert werden, jedoch die Verbindungsparameter, wie IP Adresse oder Port, nicht.

- **AF10.1 Bedienung auch für fachfremdes Personal**

Die Software sollte intuitiv und einfach per Maus bedienbar sein, damit auch fachfremdes Personal ein Nutzen davon hat. Die Symbole auf der Symbolleiste sollten eindeutig sein und die dahinter liegenden Funktionen erkennbar machen. Anlagenbediener sollten durch die einfache Bedienung in die Lage versetzt sein, Anweisungen eines Inbetriebnehmers auszuführen in dem Sie per einfachen Mausklick

auf einige Symbole klicken. Des Weiteren sollte das Verweilen mit dem Mauszeiger auf einem Windows-Control dazu führen, das ein so genanntes Tool-Tip-Fenster erscheint, in dem eine kleine Erläuterung der Bedienung mitgeteilt wird.

- **AF10.2 Dialog zur Eingabe der Parameter zum Verbindungsaufbau**

Die Eingabe der IP Adresse und der Ports erfolgt über einen Window basierten Dialog und sollte über eine Textbox erfolgen. Für die IP-Adresse sind folgende Zeichen erlaubt: 0-9 und das '.'-Zeichen. Für den Port sind nur numerische Zeichen erlaubt: 0-9. Mit einer Checkbox (Windows-Control) muss im Dialog die Möglichkeit bestehen die Verbindungsart auf Aktiv zu setzen. Sollte die Checkbox nicht markiert werden, ist die Verbindung auf Passiv gesetzt. Diese Einstellungen müssen dem Benutzer nach Verlassen des Dialoges und bei Aufbau der Verbindung über eine geeignete Anzeige mitgeteilt werden.

Für die IP-Adresse und den Port gilt eine Einschränkung; Die Software darf zu einem bestimmten Rechner **keine** Verbindung aufbauen und es sollten keine Well-Known Ports (0 - 1023) benutzt werden. Mit dem OK Button wird der Dialog geschlossen und die Daten übernommen. Mit dem

Button Abbrechen werden alle eingegebenen oder geänderten Daten wieder verworfen.

- **AF10.3 Dialog zur Definition von Telegrammen**

Die Eingabe von Telegrammen erfolgt über einen Dialog durch Eingabe in einer Textbox. Mit dem Übernehmen des eingetippten Telegramms, über ein Button Control, soll das Telegramm dann in einem scrollbaren Fenster angezeigt werden und die Textbox zu Eingabe wieder leer und bereit sein. Die Länge sowie der Inhalt des Telegramms muss frei wählbar sein. Mit Eingabe eines Telegramms sollte dem Bediener die aktuelle Länge des eingegebenen Telegramms angezeigt werden. Dies ist vorteilhaft um den Telegrammaufbau einfacher kontrolliert zu können. Die Definition von mehreren Telegrammen muss möglich sein. Die Reihenfolge der Telegrammeingabe, legt automatisch auch die Reihenfolge des Telegrammversands fest. Jedes Telegramm kann nach Eingabe nachträglich noch geändert werden, um gegebenenfalls Nachträge in den Telegrammen zu tätigen.

Es gibt Sonderfälle, in denen die Telegrammlänge fest auf eine Byte-Anzahl begrenzt ist. In diesem Fall ist eine Möglichkeit zu schaffen, die die Telegrammlänge, trotz eventuell kurzem Telegramminhalt, auf eine frei definierbare Län-

ge setzt. Die nicht definierten Bytes in einem festen Längen-Telegramms werden dann mit 0'en aufgefüllt. Mit dem OK Button wird der Dialog geschlossen und die Daten übernommen. Mit dem Button Abbrechen werden alle eingegebenen oder geänderten Daten wieder verworfen.

- **AF10.4 Dialog zur Definition von E-Mail Adressen**

Die Eingabe der E-Mail-Adressen zur Kommunikation mit externen Entwicklern oder Inbetriebnehmern sollten in einem Dialog durch Eingabe in eine Textbox erfolgen. Mit einem Windows-Control Button zur Übernahme der E-mail Adresse soll diese dann in einem scrollbaren Fenster erscheinen. Mehrere E-Mail-Adressen können so eingegeben und gespeichert werden. Mit dem Übernehmen einer E-Mail-Adresse in das scrollbare Fenster muss die Textbox wieder leer und bereit zu nächsten Eingabe zur Verfügung stehen. Mit dem OK Button wird der Dialog geschlossen und die Daten übernommen. Mit dem Button Abbrechen werden alle eingegebenen oder geänderten Daten wieder verworfen.

- **AF10.5 Dialog zum Versenden von E-Mails**

Zum versenden von Kommunikationsverläufen muss ein Dialog zur Verfügung stehen in dem alle E-Mail relevanten Angaben über Textboxen eingegeben werden können. Hier

sollte über ein geeignetes Windows-Control die Möglichkeit bestehen mehrere schon definierte E-Mail-Adressen zu wählen. Mit dem Versuch zu senden (Button senden), wird eine Benutzer-Authentifizierung angestoßen bei der die Benutzerdaten von einem SMTP-Server in einem Dialog abgefragt werden. In diesem Dialog werden Benutzername und dazugehöriges Passwort eingetragen. Erst mit gültiger Authentifizierung wird die E-Mail dann versandt.

3.1.2. Kommunikation zum Kommunikationspartner

- **AF20.1 Socket Verbindung über TCP/IP**

Die Software muss in der Lage sein einen Verbindungsaufbau zu einem geeigneten Verbindungspartner über eine Socket TCP/IP Schnittstelle herzustellen. Die benötigten Parameter, werden über einen Dialog eingegeben und stehen dann dem Programm zur Verfügung.

- **AF20.2 Siemens DU05 Prozedur mit dem Kommunikationspartner**

Die Kommunikationsprozedur, die auf TCP/IP aufsetzt, muss die Siemens DU05 Prozedur sein. Diese ist bei der TKSE Standard und ist Voraussetzung zum Aufbau einer Verbindung mit einem Kommunikationspartner. Prozedurverstöße

Be müssen abgefangen werden und dürfen nicht zu einem unvorhersagbarem Verhalten der Kommunikationssoftware führen. Verbindungsabbrüche, bewusste und / oder unbewusste, müssen von der Software selbstständig durchgeführt werden. Nach einem unerwarteten Verbindungsabbruch, muss die Verbindung selbstständig wieder aufgebaut werden. In einer geeignete Weise, sollte der Benutzer über die Verbindungsabbrüche informiert werden. Die Siemens DU05 Prozedur ist als Anhang mit angefügt.

- **AF20.3 freie Wahl des Kommunikationspartner**

Der Kommunikationspartner, vielmehr seine IP Adresse und Port Nummer, sollten zur Verbindung mit dem Partner frei wählbar sein. Sie darf nicht im Quellcode fest verankert sein. Ausnahme ist der Verbindungsaufbau zu einer IP-Adresse des Produktions-Servers.

- **AF20.4 Festlegung der Verbindungs-Art Aktiv**

Die Kommunikationssoftware muss in der Lage sein einen Verbindungsaufbau zum Partner-System *aktiv* durchzuführen. D.h. dass die zuvor parametrierten Verbindungs-Informationen aktiv genutzt werden um den Partner im Kommunikationsnetz zu kontaktieren und sich mit ihm zu verbinden.

- **AF20.5 Festlegung der Verbindungs-Art Passiv**

Die Kommunikationssoftware muss in der Lage sein, bei Wahl einer *passiven* Verbindung auf das Anfragen von Kommunikationspartnern zu „lauschen“. Dies bedeutet dass die zuvor parametrisierten Informationen auf eine bestimmte IP-Adresse und einen bestimmten Port zu warten das eine Verbindung aufgebaut wird.

3.1.3. Telegrammdefinition

- **AF30.1 Länge des Telegramms**

Die Kommunikation mit der DU05 Prozedur sieht eine freie Telegrammlänge vor. Der Partner erkennt die gesendete Länge aus den zwei Byte-Binärdaten des Telegrammkopfes. Nachdem die Länge des Telegramms aus dem Telegrammkopf ausgelesen wurde, wird diese bekannte Länge von der Socketschnittstelle gelesen. Quittungstelegramme haben erwartungsgemäß eine sehr kurze Länge, denn hier wird nur der positive oder negative Telegrammempfang quittiert.

- **AF30.2 LIVE Telegramme**

Das LIVE-Telegramm welches zu Überwachung der Kommunikationsverbindung dient, ist im Programm fest codiert. Das Ausbleiben eines LIVE-Telegramms würde einen Ver-

bindungsabbruch erzwingen, da davon ausgegangen wird dass ein Kommunikationsfehler vorliegt. Danach würde die Software einen erneuten Verbindungsaufbau durchführen. Auch diese Daten werden alle von der Socketschnittstelle *abgeholt* und dann verworfen. Auf dieses empfangene LIVE-Telegramm wird ein Quittungstelegramm gesendet. Die Kennung, dass es sich um ein LIVE-Telegramm handelt, steht direkt nach der Telegrammlänge. Dieser sieht folgendermaßen aus: II999-LIVE-TELEGRAMM (II=zwei Byte Binärdaten für die Länge, 999=Kennung LIVE-Telegramm)

- **AF30.3 Quittungstelegramme**

Der Aufbau eines positiven oder auch negativen Quittungstelegramms ist in der Kommunikationssoftware fest codiert. Dieses wird benötigt um empfangene oder gesendete Telegramme bei Fehler erneut anzufordern oder den Fehlerfreien Erhalt zu bestätigen. Der Aufbau eines positiven Quittungstelegramms sieht folgendermaßen aus: IIFFFF (II=zwei Byte Binärdaten für die Länge) FF=255 in HEX Der Telegrammaufbau für ein Negatives-Quittungstelegramm sieht folgendermaßen aus: IIFFFE (II=zwei Byte Binärdaten für die Länge)

3.1.4. Dauerhafte Speicherung von Projekt-Parametern

- **AF40.1 Speichern von Projekt-Parametern**

Die Software muss projektspezifische Parameter, wie IP-Adresse, Port, E-Mail-Adressen und Telegrammdefinitionen, projektbezogen auf ein Medium wie HDD, USB-Stick oder FDD speichern.

- **AF40.2 Laden von Projekt-Parameter**

Die Software muss projektspezifische Parameter, wie IP-Adresse, Port, E-Mail-Adressen und Telegrammdefinitionen, projektbezogen von einem Medium wie HDD, USB-Stick oder FDD in die Programmumgebung zu laden, damit diese dann für Inbetriebnahmen oder Test-Zwecke zur Verfügung stehen. Der Dateiname muss zu Informationszwecken in der Hauptfenster-Titelleiste stehen.

3.1.5. Kommunikation

- **AF50.1 Bereitstellung von Kommunikationsverlaufsinformationen**

Die Kommunikationsschnittstelle muss die Möglichkeit bieten ausgeführte Aktionen als Klartext (ASCII-Basiert) über einen Programm-Schnittstelle der Visualisierungsebene zur Verfügung zu stellen um die Informationen dem Benutzer

anzeigen zu können. Dies können sein: Informationen über...

- Verbindungsaufbau erfolgt
- LIVE-Telegramm gesendet
- Telegramm gesendet (mit Netto Daten)
- Telegramm empfangen (mit Nettodaten)
- Verbindungsabbruch
- Fehler
- usw...

Diese sollen in einem scrollbaren Windows-Control angezeigt werden damit diese kontrolliert werden können.

• **AF50.2 Senden von Telegrammen**

Die Software muss über eine Möglichkeit verfügen die zuvor definierten Telegramme an das Partner-System zu senden. Alle definierten Telegramme sollen in einem Zyklus von 2 Sekunden an den Partner versendet werden. Des Weiteren muss die Software in der Lage sein, Quittungstelegramme, der zuvor gesendeten Telegramme, auszuwerten und darauf entsprechend zu reagieren, indem sie ein Quittungstelegramm selbständig generiert und sendet. Das definierte Telegramm darf als Telegramminhalt nur ASCII-

Zeichen von A-Z, 0-9 und ',','' und '-'-Zeichen für Vorzeichen behaftete numerische Werte beinhalten.

- **AF50.3 Empfangen von Telegrammen**

Die Software muss über eine Möglichkeit verfügen, die Antworttelegramme auf die zuvor gesendeten Telegramme vom Partner-System, zu empfangen. Des Weiteren muss sie in der Lage sein empfangene Telegramme statusabhängig zu quittieren und ein Quittungstelegramm selbständig zu senden.

3.1.6. Informationsweitergabe

- **AF60.1 protokollieren des Kommunikationsverlaufes**

Der Kommunikationsverlauf soll in einem scrollbaren Windows-Control protokolliert werden. Sollten Unregelmäßigkeiten auftreten können diese visuell sofort erkannt und eingegriffen werden.

- **AF60.2 drucken des Kommunikationsverlaufes**

Der Kommunikationsverlauf soll als Bericht auf einem Drucker ausgegeben werden. Telegramme die zu lang sind, müssen mit einem Word-Wrap (Zeilenumbruch) versehen werden, damit deren Inhalt auf dem Bericht kontrolliert werden kann. Des Weiteren muss die Druckerfunktion in der

Lage sein lange Kommunikationsverläufe auf mehrere Seiten zu drucken. Die Seiten sollten für eine bessere Übersicht nummeriert sein.

- **AF60.3 speichern des Kommunikationsverlaufes**

Der Kommunikationsverlauf sollte dauerhaft auf einem Speichermedium wie USB, HDD oder FDD gespeichert werden können. Es sollte eine ASCII basierte Textdatei erstellt werden die mit einem Standard Texteditor betrachtet werden kann.

- **AF60.4 E-Mail Versand des Kommunikationsverlaufes**

Der Kommunikationsverlauf sollte per E-Mail verschickt werden können. Hier soll der Kommunikationsverlauf der im scrollbaren Windows-Control angezeigt wird in die zu übermittelnde E-Mail als Anhang mit gesendet werden. Mit der Aktion Senden der E-Mail muss sich das Programm an einem SMTP-Server anmelden und den Benutzer authentifizieren. Ist dieser dem SMTP-Server bekannt

3.1.7. Programm-Einschränkungen

- **AF70.1 Drucken während Kommunikationsverlauf nicht erlaubt**

Während einer aktiven Verbindung, darf aus dem Programm

nicht gedruckt werden, da dies zu Performance Einbrüchen führen kann.

- **AF70.3 Speichern allgemein während Kommunikationsverlauf nicht erlaubt**

Während einer aktiven Verbindung, darf aus dem Programm nicht gespeichert werden, da dies zu Performance Einbrüchen führen kann.

- **AF70.4 E-Mail Versand während Kommunikationsverlauf nicht erlaubt**

Während einer aktiven Verbindung, darf aus dem Programm nicht gemailt werden, da dies zu Performance Einbrüchen führen kann.

- **AF70.5 IP-Adresse definieren während Kommunikationsverlauf nicht erlaubt**

Während einer aktiven Verbindung, darf die IP-Adresse und der Port nicht geändert werden, da eine aktive Verbindung zum Partner besteht.

3.1.8. Use Case: Definition Telegramme

Beschreibung (description)

Über den Dialog Telegrammdefinition sollen Telegramme zu Inbetriebnahmezwecke eingegeben werden.

Beteiligte Akteure (actors)

Primärakteure sind interne Inbetriebnehmer die in erster Linie Telegramme zu Testzwecken definieren. Sekundärakteure sind hier die externen Inbetriebnehmer die Anweisungen geben, welcher Teil der Anlage über Telegramme getestet werden soll.

Status

Bereit zum Review.

Verwendete Anwendungsfälle (includes)

keine

Auslöser (rationale oder trigger)

Der interne Inbetriebnehmer.

Vorbedingungen (preconditions)

Das Hauptprogramm (DU05 Simulator) ist gestartet.

Invarianten

Alle Bedingungen, die innerhalb und durch den Anwendungsfall

nicht verändert werden dürfen, also auch in einem Misserfolgs- oder Fehlerszenario immer noch gewährleistet werden müssen.

Nachbedingung / Ergebnis (postconditions)

Die definierten Telegramme erscheinen, zum Versand zum Kommunikationspartner, in einem scrollbaren Windows-Control, im Hauptfenster.

Standardablauf (normal flow)

Die Symbolleiste wird betätigt um Telegramme zu bearbeiten oder neu zu definieren. Im erscheinenden Dialog werden dann schon definierte Telegramme angezeigt oder er erscheint mit leeren Windows-Controls falls noch keine Telegramme definiert wurden.

In einem Eingabefeld wird ein, aus numerischen und alphanumerischen Zeichen, Telegramm-Text eingegeben. Während der Eingabe kann der Bediener erkennen, wie Lang sein Telegramm schon ist indem eine Text-Anzeige die Länge des aktuellen Telegramms anzeigt. Die Bedeutung der einzelnen Zeichen und deren Position innerhalb des Telegramm-Textes werden in Absprache mit dem externen Inbetriebnehmer definiert.

Sollten Telegramme definiert werden, mit einer konstanten Länge, kann der Bediener eine Markierung innerhalb des Dialoges

wählen und in einem Eingabefeld die Länge aller jetzt einzugebenden Telegramme fest vorgeben. Sollte ein Telegramm kürzer sein wird der Rest mit Nullen aufgefüllt.

Nach Eingabe eines Telegramm-Textes wird ein Windows-Control 'Übernahme' betätigt welches das eben definierte Telegramm in einem scrollbares Windows-Control innerhalb des Dialoges anzeigt und das Eingabefeld für Telegramm-Text löscht. Dieser Vorgang kann mehrfach wiederholt werden.

Wählt man ein Telegramm aus dem scrollbaren Windows-Control, erscheint dieses zur Bearbeitung in dem Eingabefeld. Nach der Bearbeitung wird dieses Telegramm wieder über das 'Übernahme' Windows-Control zurück in das scrollbare Windows-Control geschrieben.

Nachdem alle Telegramm definiert oder bearbeitet wurden, wird der Dialog über ein Windows-Control positiv quittiert. Der Dialog wird geschlossen und die definierten Telegramme im scrollbaren Windows-Control des Hauptfensters angezeigt.

Nun stehen dem Programm und dem Primärakteur die definierten Telegramme für die Inbetriebnahme zur Verfügung.

Alternative Ablaufschritte (alternative flow)

Die Symbolleiste wird betätigt um Telegramme zu bearbeiten oder neu zu definieren. Im erscheinenden Dialog werden dann

schon definierte Telegramme angezeigt oder er erscheint mit leeren Windows-Controls falls noch keine Telegramme definiert wurden.

In einem Eingabefeld wird ein, aus numerischen und alphanumerischen Zeichen, Telegramm-Text eingegeben. Die Bedeutung der einzelnen Zeichen und deren Position innerhalb des Telegramm-Textes werden in Absprache mit dem externen Inbetriebnehmer definiert. Nach Eingabe eines Telegramms wird ein Windows-Control 'Übernahme' betätigt, welches das definierte Telegramm in ein scrollbares Windows-Control innerhalb des Dialoges anzeigt. Das Eingabefeld wird dann gelöscht. Dieser Vorgang kann mehrfach wiederholt werden.

Sollten die Eingaben oder Änderungen (Telegramme) doch nicht benötigt werden, kann der Dialog über ein Windows-Control 'Abbrechen' negativ quittiert werden und der Dialog schließt ohne die neu definierten oder geänderten Telegramme zu übernehmen.

Ende

3.2. Use Case: Definition Verbindungsparameter

Beschreibung (description)

Über den Dialog Verbindungsparameter sollen die IP Adresse, ein Port und die Art der Verbindung (Aktiv / Passiv) eingetragen werden.

Beteiligte Akteure (actors)

Primärakteure sind interne Inbetriebnehmer, die die IP-Adresse, den Port und die Art der Verbindung in das Programm einpflegen. Sekundärakteure sind hier die externen Inbetriebnehmer die die IP-Adresse, den Port und die Art der Verbindung kennen müssen und in Ihre Anlage einpflegen.

Status

Bereit zum Review.

Verwendete Anwendungsfälle (includes)

keine

Auslöser (rationale oder trigger)

Der interne Inbetriebnehmer.

Vorbedingungen (preconditions)

Das Hauptprogramm (DU05 Simulator) ist gestartet.

Invarianten

Bedingung zur gültigen Eingabe der Verbindungsparameter ist, dass das DU05 Simulationsprogramm gestartet ist.

Alle Bedingungen, die innerhalb und durch den Anwendungsfall nicht verändert werden dürfen, also auch in einem Misserfolgs- oder Fehlerszenario immer noch gewährleistet werden müssen.

Nachbedingung / Ergebnis (postconditions)

Die Verbindungsparameter werden nach positiver Beendigung des Dialoges an das Hauptprogramm, für einen späteren Verbindungsaufbau zum Partner, zurückgegeben.

Standardablauf (normal flow)

Die Symbolleiste, mit dem Icon zur Eingabe der Verbindungsparameter, wird betätigt um die IP-Adresse, Port und die Art der Verbindung einzugeben.

Sollten die Verbindungsparameter schon einmal vorher korrekt eingegeben worden sein, erscheinen diese schon im Dialog. Die IP-Adresse sowie der Port werden in einem Eingabefeld angezeigt. Die Art der Verbindung kann über einen Haken gesetzt und damit auf eine aktive Verbindung gesetzt werden. Wird der Haken nicht gesetzt, handelt es sich um eine passive Verbindung.

In einem Eingabefeld wird eine, aus numerischen-Zeichen und

dem Punkt, die IP-Adresse eingegeben. Im zweiten Eingabefeld für den Port, können nur numerische Zeichen eingegeben werden. Es sollten keine *Well-Known-Ports* verwendet werden.

Nachdem alle Verbindungsparameter eingegeben wurden kann der Dialog mit einem Windows-Control 'OK' positiv quittiert und geschlossen werden. Vor der Schließung des Dialoges werden die eingegebenen Daten Validiert. Sind die Daten in einem Programm gültigen Zustand wird der Dialog geschlossen und die Daten stehen dem Programm zur Verfügung.

Fällt die Validierung negativ aus, d.h. dass die Daten nicht der Konformität entsprechen, wird eine Warnmeldung dem Bediener angezeigt und der Dialog bleibt geöffnet.

Nun können die Daten korrigiert werden und der Dialog erneut mit dem Windows-Control 'OK' positiv quittiert werden. Sind keine Fehler mehr vorhanden werden die Daten dem Hauptprogramm übergeben und der Dialog wird geschlossen.

Alternative Ablaufschritte (alternative flow)

Die Symbolleiste wird betätigt um Telegramme zu bearbeiten oder neu zu definieren. Im erscheinenden Dialog werden dann schon definierte Telegramme angezeigt oder er erscheint mit leeren Windows-Controls falls noch keine Telegramme definiert wurden.

In einem Eingabefeld wird ein, aus numerischen und alphanumerischen Zeichen, Telegramm-Text eingegeben. Die Bedeutung der einzelnen Zeichen und deren Position innerhalb des Telegramms werden in Absprache mit dem externen Inbetriebnehmer definiert. Nach Eingabe eines Telegramms wird ein Windows-Control 'Übernahme' betätigt welches das definierte Telegramm in ein scrollbares Windows-Control innerhalb des Dialoges anzeigt. Daraufhin wird das Eingabefeld gelöscht. Dieser Vorgang kann mehrfach wiederholt werden.

Sollten die Eingaben oder Änderungen (Telegramme) doch nicht benötigt werden, kann der Dialog über ein Windows-Control 'Abbrechen' negativ quittiert werden und der Dialog schließt ohne die neu definierten oder geänderten Telegramme zu übernehmen.

Ende

3.2.1. Use Case: Definition E-Mail-Adressen

Beschreibung (description)

Über den Dialog zur definition der projektbezogenen E-Mail Adressen.

Beteiligte Akteure (actors)

Primärakteure sind interne Inbetriebnehmer, die die Aufgabe haben das Ergebnis des Kommunikationsverlaufes in besonderen Fällen an die externen Entwickler oder auch an die internen Entwickler, aus der Anlagenumgebung heraus, zu mailen. Die Mail-Adressen sind in der Regel projektbezogen und werden in diesem Dialog eingegeben. Sekundärakteure sind hier die externen Inbetriebnehmer, die daran interessiert sind, den Kommunikationsverlauf von Ihren (externen) Entwicklern in besonderen Fällen überprüfen zu lassen und selbst aus der Anlagenumgebung heraus keine Möglichkeit haben über einen SMTP Server Mails zu senden.

Status

Bereit zum Review.

Verwendete Anwendungsfälle (includes)

keine

Auslöser (rationale oder trigger)

Der interne Inbetriebnehmer. (bei Projektvorbereitungen)

Vorbedingungen (preconditions)

Das Hauptprogramm (DU05 Simulator) ist gestartet.

Invarianten

E-Mail Adressen müssen bekannt sein.

Alle Bedingungen, die innerhalb und durch den Anwendungsfall nicht verändert werden dürfen, also auch in einem Misserfolgs- oder Fehlerszenario immer noch gewährleistet werden müssen.

Nachbedingung / Ergebnis (postconditions)

Die E-Mail Adressen werden nach positiver Beendigung des Dialoges in den Speicher des Hauptprogramms geladen, und dienen als Daten für den Dialog E-Mail senden.

Standardablauf (normal flow)

Die Symbolleiste, mit dem Icon zur Eingabe der E-Mail Adressen, wird betätigt um die E-Mail Adressen einzugeben.

Sollten die E-Mail Adressen schon vorhanden sein, erscheinen diese schon im Dialog. Nachdem in das Eingabefeld eine gültige E-Mail Adresse eingegeben hat, wird diese durch das Windows-Control 'Übernehmen' in ein scrollbares Windows-Control übernommen. Bei Eingabe einer ungültigen E-Mail Adresse, erscheint ein Nachricht für den Benutzer, dass die E-Mail Adresse nicht gültig ist. Sind für das Projekt alle E-Mail Adressen eingegeben worden, kann der Dialog über ein Windows-Control 'OK' positiv quittiert werden. Die Daten werden in den Speicher des Hauptprogramms geladen und können von dort aus persistent auf ein Speichermedium angelegt werden.

Alternative Ablaufschritte (alternative flow)

Die Symbolleiste, mit dem Icon zur Eingabe der E-Mail Adressen, wird betätigt um die E-Mail Adressen einzugeben.

Sollten die E-Mail Adressen schon vorhanden sein, erscheinen diese schon im Dialog. Sollte sich im Projektverlauf die Zuständigkeiten ändern und sich E-Mail Adressen ändern, können diese über den Dialog gelöscht oder neue hinzugefügt werden. Dies geschieht über das Windows-Control 'Löschen'. Dazu wird die zu entfernenden E-Mail Adresse im scrollbaren Windows-Control mit einem Zeigegerät markiert und das Windows-Control 'Entfernen' betätigt. Nun ist die E-Mail Adresse gelöscht. Wird der Dialog über das Windows-Control 'OK' positiv quittiert stehen die E-Mail Adressen dem Programm zur Verfügung.

Wir der Dialog negativ über das Windows-Control 'Abbrechen' quittiert bleiben die eventuell entfernten E-Mail Adressen erhalten. **Ende**

3.2.2. Use Case: Definition Speichern und Laden der Projektdatei

Beschreibung (description)

In der Symbolleiste gibt es ein Icon, das mit einem Disketten Symbol gekennzeichnet ist. Wurde das Programm für ein

Inbetriebnahme-Projekt mit Daten (Telegramme / Verbindungsparameter / E-Mail Adressen) versorgt, können diese Daten persistent auf ein Speichermedium angelegt werden. Dazu wird das Diskettensymbol mit der Mausclick betätigt und es erscheint ein Datei-Speichern Dialog. Hier wird der Dateiname eingegeben und die Daten gespeichert.

Wird das Programm neu gestartet und es wurden noch keine Daten in das Programm eingegeben, können diese aus der gespeicherten Projektdatei (*.prj) in das Programm geladen werden. Die Daten stehen dann dem, Program zur Verfügung.

Beteiligte Akteure (actors)

Primärakteure sind interne Inbetriebnehmer, die die projektbezogenen Informationen persistent speichern möchten.

Status

Bereit zum Review.

Verwendete Anwendungsfälle (includes)

keine

Auslöser (rationale oder trigger)

Der interne Inbetriebnehmer. (bei Projektvorbereitungen)

Vorbedingungen (preconditions)

Das Hauptprogramm (DU05 Simulator) ist gestartet und pro-

jektbezogene Daten wurden bereits eingegeben und gespeichert oder sollen gespeichert werden.

Invarianten

Projektbezogenen Daten müssen im System vorhanden sein. Dabei kann es sich Telegrammdefinitionen, Mail Adressen oder Verbindungsparameter handeln. Des Weiteren können Daten schon gespeichert worden sein, und sollen zu Inbetriebnahmezwecken in das Programm geladen werden.

Nachbedingung / Ergebnis (postconditions)

Zwei Möglichkeiten die zur Auswahl stehen wie mit Daten umgegangen werden kann. Die erste Möglichkeit die Daten aus dem System sollen gespeichert werden. Die zweite Möglichkeit ist das Laden einer vorhandenen Projekt-Datei um Daten in dem Programm zur Verfügung zu stellen.

Standardablauf (normal flow)

Daten sind in das Programm eingegeben worden. Dies können alle programmbezogenen Daten sein, wie Telegrammdefinitionen, Mail-Adressen oder Verbindungsparameter. Das Icon in der Symbolleiste zum speichern muss mit einem Zeigegerät gewählt werden. Darauf erscheint ein Dialog zum Speichern von Dateien. In dem Eingabefeld 'Dateiname' wird der Projektbezogene Name der Datei eingegeben. Nach Eingabe wird das

Windows-Control 'Speichern' betätigt, die Daten werden persistent gespeichert und der Dialog geschlossen.

Alternative Ablaufschritte (alternative flow)

Daten sind durch das Programm persistent gespeichert worden und sollen in das Programm geladen werden. Dies können alle Programm bezogenen Dateien sein, wie Telegrammdefinitionen, Mail Adressen oder Verbindungsparameter Das Icon in der Symbolleiste zum Öffnen muss mit einem Zeigegerät gewählt werden. Es erscheint ein Dialog zum Öffnen von Dateien. In dem Eingabefeld 'Dateiname' wird der projektbezogene Name der Datei, die geladen werden soll, eingegeben. Nach Eingabe wird das Windows-Control 'Öffnen' betätigt und die Daten werden in das Programm geladen und der Dialog wird geschlossen. **Ende**

3.2.3. Szenarien

Hier sollten einige Szenarien stehen.

3.3. Akteure

Als Akteure werden 5 Gruppen definiert:

- interne Inbetriebnehmer
- interne Softwareentwickler
- externe Inbetriebnehmer
- externe Softwareentwickler
- Leitstand-Personal

In der Regel bleiben die Akteure in Ihren Gruppen und es gibt nur selten Querschnittsbereiche, doch es kann vorkommen, dass interne Softwareentwickler zur Problemlösung an den Anlagen- bzw. Software-Lieferanten *ausgeliehen* werden, um z.B. bei Lösungen zu Kommunikationsproblemen zu helfen und ihr Know How einfließen zu lassen.

3.3.1. internen Inbetriebnehmer

Die Gruppe der **internen Inbetriebnehmer** hat die Funktion die im Auftrag an den Lieferanten definierte Telegrammverläufe und Verbindungsparameter im Programm über die Telegrammdefinition und Kommunikationsverbindung so vorzubereiten, dass die Inbetriebnahmetests, mit Neu-Anlagen oder Anlagen-Erweiterungen, reibungslos durchgeführt werden können. Sie kontrollieren den Kommunikationsverlauf auf Telegrammebene und schreiten bei Problemen sofort ein. Des Weiteren sind sie in der Lage, die

Ergebnisse des Kommunikationsverlaufes an die entsprechenden Zuständigkeitsbereiche, über die geeigneten Programm-Funktionen, weiter zu leiten.

3.3.2. internen Softwareentwickler

Die Gruppe der **internen Softwareentwickler** nutzt diesen Simulator um die vorhandene Anlagen-Software in der Entwicklungsumgebung auf die Neu- oder Erweiterte-Anlagen-Software anzupassen. Hier werden die Telegramme im Simulator definiert die aus der Neu-Anlage kommen und mit der vorhanden Software kommunizieren soll. Die Kommunikationsparameter werden auf die Kommunikations-Gegenstelle gesetzt. Auch hier werden Protokolle des Kommunikationsverlaufes an die entsprechenden Zuständigkeitsbereiche weitergeleitet.

3.3.3. externen Inbetriebnehmer

Die Gruppe der **externen Inbetriebnehmer** kommuniziert mit der Gruppe der **internen Inbetriebnehmer**. Sie definieren den Startzeitpunkt der Kommunikation, das Ändern oder das Definieren von Telegrammen. Aufgetretene Fehler oder Ungereimtheiten während des Kommunikationsverlaufs werden über den im Simulator protokollierten Kommunikationsverlauf analysiert.

Anhand des Ergebnisses der Kommunikationsverlaufs-Analyse entscheidet der externe Inbetriebnehmer ob das Kommunikationsprotokoll an die *externen Softwareentwickler* gesendet werden muss. Er gibt der Gruppe des **Leitstand-Personals** Anweisungen die Telegramme, aus dem verbundenen Simulator, an den Partner zu senden. Dieses Szenario ist nicht die Regel und ist gedacht für die eventuell nächtliche Inbetriebnahme, in denen kein interner Inbetriebnehmer oder interner Softwareentwickler zur Verfügung steht.

3.3.4. externe Softwareentwickler

Die Gruppe der **externe Softwareentwickler** nutzt diesen Simulator um die Neu zu liefernde Anlagen-Software in deren Entwicklungsumgebung zu erstellen und die Kommunikation zu testen, und gegebenenfalls schon im eigenen Unternehmen ihre Verbindungssoftware anzupassen. Hier werden die Telegramme im Simulator definiert, die schon in der aktuell laufenden Software in der Anlage vorhanden sind und zu Testzwecken an die neu Anlage gesendet werden sollen. Die Kommunikationsparameter werden auf die Kommunikations-Gegenstelle gesetzt. Auch hier werden Protokolle des Kommunikationsverlaufes an die entsprechenden Zuständigkeitsbereiche weitergeleitet.

3.3.5. Leitstand-Personals

Die Gruppe des **Leitstand-Personals** nutzt diese Software auf Anweisung vom externen Inbetriebnehmer s.o.. Dieser hat dafür zu sorgen, dass der Simulator in einem für das Leitstand-Personal geeigneten Zustand ist d.h. die Verbindungen zum Partner-Systeme besteht und über **eine** Bedienung das Senden der Telegramme angestoßen wird. Des Weiteren kennt in der Regel nur das Leitstand-Personal den Anlagenablauf (Produktionsablauf) und kann erkennen ob die Anlage sich während des Kommunikationsverlaufs so verhält, wie er es erwartet oder kennt. Auch angezeigte Testdaten auf Leitstand-Monitoren, können vom Leitstand-Personal auf Korrektheit hin geprüft werden.

3.4. Anwendungsfalldiagramm

Sequenz-Diagramme zur Eingabe der Verbindungsparameter:

Sequenz-Diagramme zur Eingabe der Telegramme:

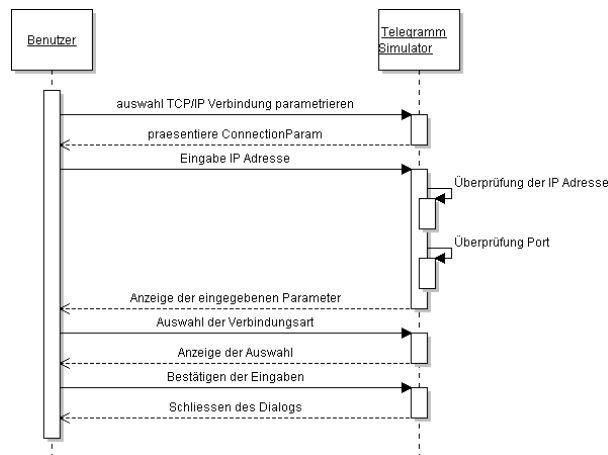


Abbildung 12: Sequenz-Diagramm Gesamtübersicht

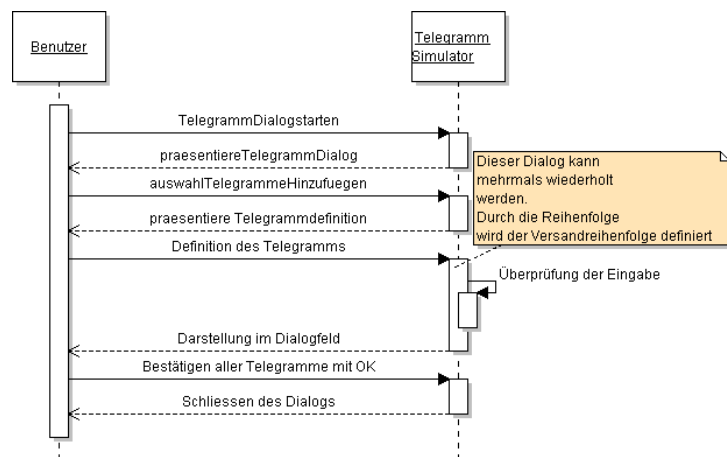


Abbildung 13: Sequenz-Diagramm Gesamtübersicht

Sequenz-Diagramme zur Eingabe der E-Mail Adressen:

Sequenz-Diagramme zur Eingabe der E-Mail senden:

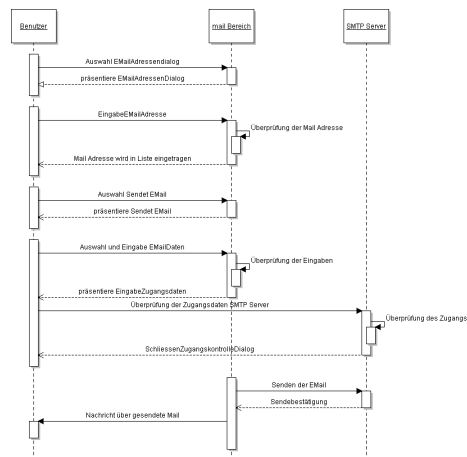


Abbildung 14: Sequenz-Diagramm Gesamtübersicht

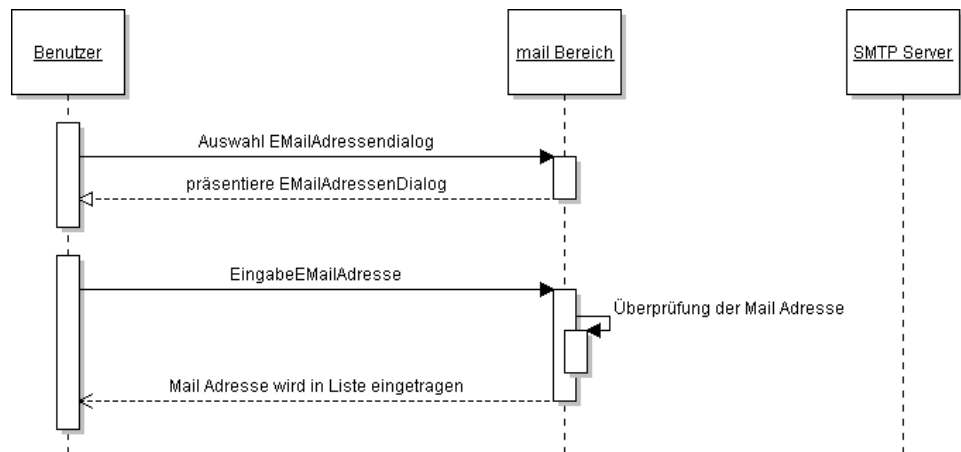


Abbildung 15: Sequenz-Diagramm Gesamtübersicht

3.5. Klassen

3.6. Verhaltensdiagramme Anforderungsanalyse

3.7. Benutzungs- und Systemschnittstellen

4. Nicht-Funktionale Anforderung

4.1. NF10 Einfachheit des Programms

Eine Anforderung an das Simulationsprogramm ist die Einfachheit der Bedienung. Die Bedienung sollte selbsterklärend sein. Geeignete Windows-Controls sollten hier zum Einsatz kommen, um die Benutzung zu erleichtern.

4.2. NF20 Programmierung in CSharp

Aus Gründen der Unternehmensstandards und der Unternehmens-Richtlinien, ist das Programm mit dem Entwicklungstool Microsoft® Visual Studio 2008 unter Verwendung der Programmiersprache C#⁴ zu entwickeln.

⁴gesprochen ci-sharp

4.3. NF30 Qualitätsanforderung

4.3.1. Softwarequalität

Zur Definition von zu erreichenden Qualitätszielen muss der Begriff der Softwarequalität präzisiert werden. Dazu werden Qualitätsmerkmale, deren Gesamtheit die Qualität der Software ausmacht. Eine solche Aufschlüsselung von Softwarequalitätsmerkmalen liefert die unter anderem die ISO Norm 9126. Diese ISO Norm definiert die Qualitätsmerkmale die im Folgenden vorgestellt werden:

- Übertragbarkeit -> Anpassbarkeit und Installierbarkeit
- Anwendbarkeit -> Erlernbarkeit, Beherrschbarkeit und Verständlichkeit
- Effizienz -> und zwar raum-, zeit- und ressourcenbezogen
- Funktionalität -> Angemessenheit, Interoperabilität und Genauigkeit
- Zuverlässigkeit -> Fehlertoleranzen, geringe Fehlerhäufigkeit und Fehlerverfolgbarkeit
- Wartbarkeit -> Analysierbarkeit, Änderbarkeit, Stabilität und Testbarkeit

Nicht jeder dieser Punkte wird von der Software tangiert. Im Folgenden wird auf diejenigen Punkte eingegangen die von der Software zu realisieren sind.

4.3.2. Übertragbarkeit

Der Simulator sollte ohne administrative Rechte auf jedem Rechner installierbar sein. Denn Inbetriebnehmer haben oft auf Ihren Rechnern keine Rechte zur Installation von Programmen. Der Simulator sollte des Weiteren in der Lage sein jede Art von Inbetriebnahme im Bereich Siemens DU05 Prozedur basierend auf TCP/IP Kommunikation und Telegrammtests durchzuführen. Die verwendete Programmiersprache für den DU05 Simulator ist Microsoft C#.

4.3.3. Anwendbarkeit

Der Simulator muss ohne große Vorkenntnisse Bedienbar sein. Dialog sollten zur leichteren Bedienbarkeit von den Windows-Controls her, gleich aufgebaut sein. Auf eine Menüleiste ist der Übersichtlichkeit zu verzichten und stattdessen zur einfacheren Bedienung eine Symbolleiste einzusetzen.

4.3.4. Effizienz

Das Reagieren auf empfangene Telegrammen, oder auch auf Antwortzeiten von Quittungstelegrammen unterliegen einem genau definierten Zeitverhalten. Daraus resultieren eventuell Verbindungsabbrüche (Prozedurverstöße) oder zu versendende Lebenstelegramme (sogenannte Live-Telegramme⁵). Das Zeitverhalten von zu senden Telegrammen ist fest auf 2 Sekunden eingestellt, d.h. das jedes einzelne definierte Telegramm alle zwei Sekunden verschickt wird bis die Liste der definierten Telegramme abgearbeitet ist. Während der Simulator verbunden ist und Telegramme mit dem Partner austauscht, muss die Bedienung des Programms jeder Zeit gewährleistet sein, d.h. auch bei hohem Kommunikationsaufkommen muss die Priorität auf dem Zeitverhalten der DU05 Prozedur liegen genauso wie auf die Bedienbarkeit des Simulators.

4.3.5. Wartbarkeit

Das Programm sollte Modular aufgebaut werden um eine leichte Wartbarkeit und Änderbarkeit zu gewährleisten.

⁵Live-Telegramme kontrollieren eventuell nicht gemerkte Verbindungsabbrüche durch zwischengeschaltete Router oder Switches im Netzwerkbereich. Sie werden in einem bestimmten Zeitzyklus gesendet und müssen positiv quittiert werden.

4.3.6. Stabilität

Das Programm darf bei Kommunikationsproblemen wie, Prozedurverstöße, Netzwerkproblemen, ausbleiben von LIVE-Telegrammen, Steuerzeichen (Schmierzeichen) innerhalb eines empfangenen Telegramms und Fehlbedienungen des Programms nicht abbrechen. Auch fehlende Bedingungen für den korrekten Funktionsablauf dürfen nicht zum Abbruch des Programms führen.

4.3.7. Sicherheit

Der Punkt Sicherheit hat an dieser Stelle eine besondere Bedeutung. Hier kann es zu einer Gefahr für Leib und Leben kommen. Denn die im DU05 Simulator definierten Telegramme steuern, nach Versand, direkt und unmittelbar ganze Anlagen. Telegramme die mit dieser Software definiert wurden, dürfen *nur* nach dem Siemens DU05 Prozedur Standard versendet werden. Im Fehlerfall, wie Prozedurverstöße, Telegramm-Längen Probleme und Plausibilitätsprüfungen, müssen Telegramme verworfen und/oder die Verbindung abgebaut werden und wieder neu aufgebaut werden.

4.4. Technische Anforderung und Einschränkungen

Nachfolgend werden die Anforderungen an die Installation, Rechnerhardware, Betriebssystem und Netzinfrastruktur beschrieben.

4.4.1. Installation

Die Installation sollte sich darauf beschränken, dass die Softwarekomponenten einfach in ein Verzeichnis kopiert werden können, ohne in der Registry oder in Config-Dateien Einträge oder Änderungen gemacht werden müssen. Denn es ist davon auszugehen dass viele mitgebrachte PC-Systeme nicht von jedem mit Software erweitert werden darf.

4.4.2. Betriebssystem

Für das Programm DU05 Simulator wird ein Betriebssystem der Firma Microsoft® vorausgesetzt (oder höher) ab der Version 5.01 namentlich mit Windows XP® bezeichnet. Auf diesem Betriebssystem muss das Microsoft Framework 3.5 installiert sein um alle Funktionalitäten des Simulators nutzen zu können. Die Treiber für die Grafikkarte und für die Netzwerkkarte müssen installiert sein um die Telegramme später mit dem

Partner über Netz austauschen zu können. Der Grafikkarten-Treiber gewährleistet ein flimmerfreies Verhalten der Software bei schnellem Telegrammaustausch und den Informationen die im Fenster über den Kommunikationsverlauf angezeigt werden. Mit nicht installiertem Treiber wird die Performance des Programms deutlich verschlechtert.

4.4.3. Hardware

Die Mindest-Voraussetzung für einen Rechner auf dem der DU05 Simulator laufen soll, sollte folgende Mindest-Ausstattung besitzen:

- Pentium 300-MHZ-Prozessor oder schneller
- Mindestens 512 MB RAM (1024 MB werden empfohlen)
- Mindestens 1,5 GB Speicherplatz auf der Festplatte
- Ethernet-Netzwerkkarte
- Tastatur und Microsoft Mouse oder ein kompatibles Zeigergerät
- Videoadapter und Monitor mit Super VGA (800 x 600) oder höherer Auflösung

4.4.4. Netzinfrastruktur

Die Netzinfrastruktur in den Anlagen und den Entwicklungsumgebungen in denen der DU05 Simulator eingesetzt wird muss über ein Ethernet-Netzwerk mit dem Standard IEEE-Norm 802.3 verfügen auf dem ein TCP/IP-Protokoll mindestens in der Version 4 zum Einsatz kommt.

4.4.5. Lizenzen

Da diese Software nicht kommerziell eingesetzt wird, sind Lizenzierungsverfahren und Produktregistrierung hier nicht zu berücksichtigen und nicht zu implementieren.

5. Benutzeroberflächen

Die Gestaltung der Benutzeroberfläche ist hier in verfeinerter Form dargestellt:

The 'Hauptfenster' interface features a title bar with the text 'Hauptfenster' and standard window controls (minimize, maximize, close). Below the title bar is a toolbar with buttons for 'öffnen', 'speichern', 'Proj. speichern', 'drucken', 'verbinden', 'IP', and a right-pointing arrow. The main area contains two large text input fields: the top one is labeled 'zu sendende Telegramme' and the bottom one is labeled 'empfangene Telegramme'. At the bottom, there is a status bar with four sections: 'Status-Text', 'Empfangen 000', 'Gesendet 000', and 'Zeit 00:00'.

Abbildung 16: Spezifizierte Benutzeroberfläche Hauptfenster

The 'E-Mail Adressen' dialog box has a title bar with the text 'E-Mail Adressen' and standard window controls. The main area is titled 'E-Mail-Adresse:' and contains two text input fields, one with 'emp2@mail.de' and another with 'emp1@mail.de'. To the right of these fields are two buttons: 'Hinzufügen' and 'Löschen'. At the bottom of the dialog are two buttons: 'OK' and 'Abbruch'.

Abbildung 17: Dialog Zeichnung Mail Adressen definition

Telegrammdefinition		—	■	✕
Definierte Telegr.: <div></div>				
Selektierte Telegr.: <input checked="" type="checkbox"/> Mit 0'en auffüllen <div>600</div> Bytes				
<div></div>				
OK		Abbruch		

Abbildung 18: Dialog Telegrammdefinition

Verbindungsparameter		—	■	✕
IP-Adresse:	Port:			
<div>123.27.211</div>	<div>1234</div>			
<input checked="" type="checkbox"/> aktive Verbindung				
OK		Abbruch		

Abbildung 19: Dialog Verbindungsparameter

Abbildung 20: Dialog Mail senden

5.1. Architektur

Die hier in diesem Projekt zu wählende Architektur beruht auf dem MVC Konzept.

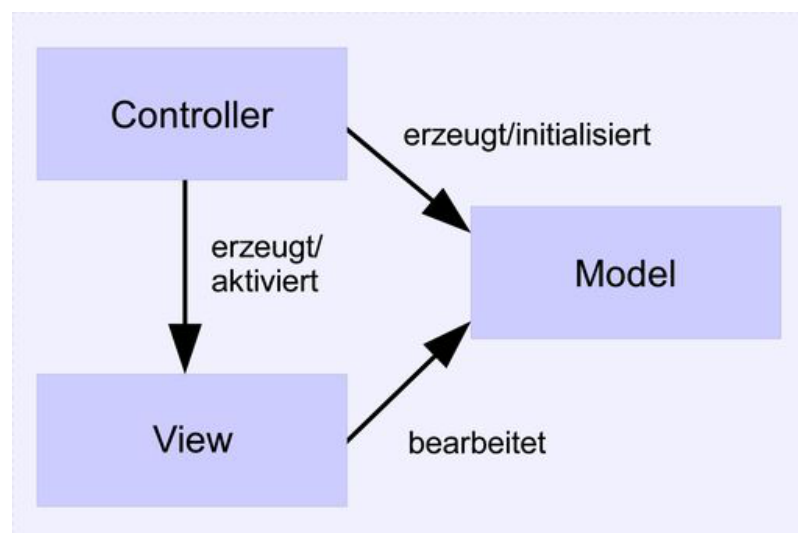


Abbildung 21: Programm-Architektur MVC

5.2. Technische Anforderungen und Einschränkungen

6. Fachliche Architektur

Die fachliche Architektur beinhaltet Klassen-Diagramme, Aktivitäten-Diagramme und Use-Case-Diagramme, welche im Hinblick auf die formale Sprache der Entwickler angepasst wurden. Genau so wie bei den Aktivitäten-Diagrammen zur Anforderungsermittlung werden auch hier nur die Diagramme berücksichtigt, die für den Anwendungsfall „Simulation DU05 Telegramme“ von Bedeutung sind, d.h. dass Klassen des Programmiersprachen-Frameworks nicht dargestellt werden.

6.1. Klassendiagramme Softwarespezifikation

Klassen-Diagramm Programm DU05 Simulator:

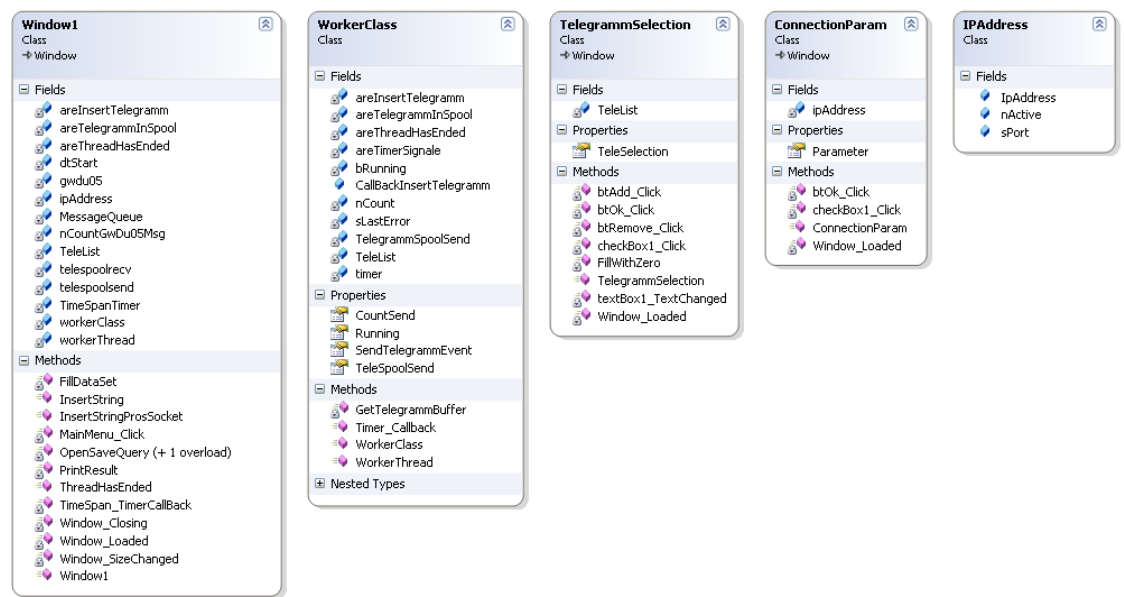


Abbildung 22: Klassen-Diagramm DU05 Simulator

Klassen-Diagramm Kommunikationsschnittstelle:

Klassen-Diagramme in der Gesamtübersicht:

7. Lieferumfang

Im Lieferumfang enthalten ist der Siemens DU05 Simulator, Microsoft® Dot. Framework 3.5 (wenn nicht schon vorhanden) und die SocketClass.dll (in der die Kommunikationsprozedur enthalten ist). Der Du05 Simulator liegt als nicht zu Installierende EXE Datei vor. Sollte das Microsoft® nicht auf dem Rechner vorhanden sein muss die Installations-Version des Frameworks installiert werden.

8. Abnahmekriterien

Die Abnahme Kriterien, die durch das Unternehmen TKSE vorgelegt wurden, bestehen aus Installations-, Kommunikations-, Integrations- und Stabilitäts-Tests.

8.1. Vorgaben des Lastenheftes

8.2. Abnahmetest

Folgende Tätigkeiten werden bei den Tests für die Abnahme des DU05 Simulators durchgeführt:

- Installations-Test

- die DU05 Simulator-EXE Datei und die Assembly SocketClass.dll werden in ein Verzeichnis auf einem Microsoft Windows XP Rechner (ohne besondere Rechte des Benutzers) kopiert
- die SocketClass.dll sollte im gleichen Verzeichnis liegen wie die EXE Datei, wenn nicht muss diese in ein Verzeichnis kopiert werden die über die Environment Variable „Path“ gefunden wird
- durch klicken auf die Siemens DU05 Simulator EXE Datei muss das Programm nun Einwand- und Fehlerfrei starten
- nachdem das Programm einwandfrei gestartet wurde, werden alle Anforderungen an des Programm kontrolliert und getestet
- Kommunikations-Test
 - das Programm wird anhand der Anforderungen parametrisiert (IP-Adresse, Port, Verbindungsart)
 - das Partner Programm wird ebenfalls auf die Gegebenheiten parametrisiert ((IP-Adresse, Port, Verbindungsart)
 - die Kommunikations wird aufgebaut

- nun werden unterschiedliche Fehler simuliert (Verbindungsabbrüche / Fehlerhafte Telegramme / usw...)
- Integrations-Test
 - der Test-Rechner wird mit einer, für die Anlage benötigte IP Adresse vergeben und in der Anlage aufgestellt und in die Netzwerkinfrastruktur eingebunden
 - nun werden wiederum Kommunikationstests durchgeführt
- Stabilitäts-Test
 - der Rechner (das Simulations-Programm) wird in der Testumgebung aufgebaut
 - nun werden Telegramme in einem hohen Zyklus an den Test-Partner gesendet und empfangen
 - in dieser Situation dürfen keine Telegramme verloren gehen
 - es dürfen keine Verbindungsabbrüche auftreten
 - das Programm darf zu keinem Zeitpunkt abbrechen

9. Erfahrungsbericht (Fazit)

Auch hier steht wieder an erster Stelle die Schwierigkeit die Granularität in den Beschreibungen einzuhalten. Oft wird festgestellt dass die Feinheit hier schon überschritten wurde oder dass Anmerkungen / Beschreibungen zu detailliert sind. Somit wieder die Problematik der Trennung zwischen Lasten und Pflichtenheft.

Der Arbeitsaufwand wurde immer dann enorm hoch, wenn Änderungen in dem Pflichtenheft, durch den Kunden, vorgenommen wurden, und diese dann Rückwirkend in das Lastenheft integriert werden mussten. Das Gleiche gilt auch für das Lastenheft, wenn dort Änderungen gemacht wurden, mussten diese wiederum Rückwirkend in alle noch folgenden Artefakte integriert werden.

Des Weiteren war es schwierig die unterschiedlichen Diagramme den einzelnen Artefakten zuzuordnen. Nicht immer war klar wohin mit dem Diagramm. Dann kam es schon mal vor dass die gewählten Tools nicht alle Anforderungen erfüllten, die im neuen Artefakt erforderlich waren. Dies kostete wiederum sehr viel Zeit die in einem engen Zeitrahmen (Verbundstudium) nicht immer gegeben war.

Wiederum entstanden kleine Probleme, nämlich, dass in den Meetings manche Themen zu langwierig wurden und diese auf ein anderes Mal verschoben werden mussten um vielleicht noch weitere, auch wichtige, Anforderungen zu besprechen.

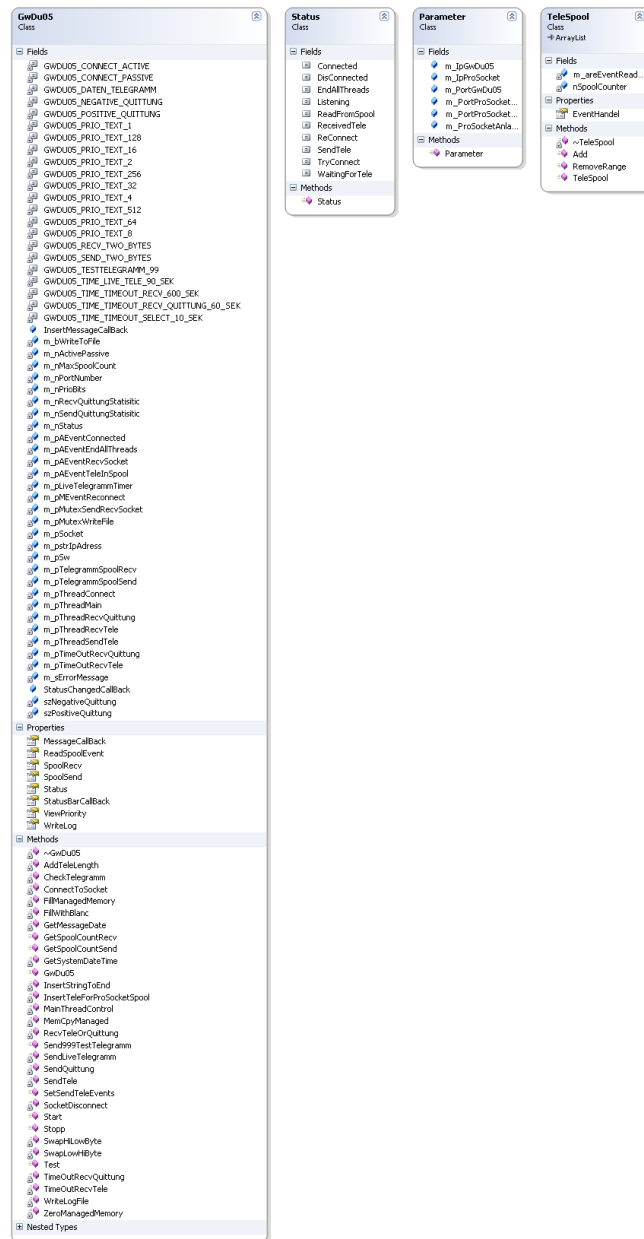


Abbildung 23: Klassen-Diagramm Kommunikation



Teil IV.

Grobentwurf

Versionshistorie:

Version	Datum	Was?
0.1	20.03.2010	Einstellung des Dokumentes
0.2	27.03.2010	Produktfunktionen eingearbeitet
0.3	01.05.2010	Bereinigung des Dokumentes
0.4	18.05.2010	Korrekturen nach PeerReview
0.5	15.07.2010	kleine Korrekturen

Tabelle 12: Versionsübersicht Grobentwurf

1. Einleitung Grobentwurf

Der Grobentwurf basiert auf dem Architekturkonzept und der Softwarespezifikation, welche beide im Rahmen des Pflichtenheftes vorgestellt wurden. Daraus ergibt sich eine Klassenkonstellation die sehr stark am methodischobjektorientierten Softwareentwicklungsansatz angelehnt ist. Es existiert für jede im Rahmen des Pflichtenheftes erläuterte Rolle eine AS Klasse (Anwendungsfall Schnittstellenklasse), welche die Aufrufe entgegennimmt. Anschließend werden diese Aufrufe an die AAS Klassen (Akteur Anwendungsfallsschnittstellenklasse) weitergeleitet. Fast für jeden Anwendungsfall existiert eine solche Klasse. In einzelnen Fällen war es jedoch sinnvoll mehrere Anwendungsfälle in einer AAS Klasse zusammenzufassen, um redundanten Code und eine unnötige komplizierte Bedienung zu vermeiden. Die AAS Klassen interagieren anschließend mit den K Klassen (Kontrollklassen), welche die eigentliche Anwendungslogik beinhalten und direkt mit den Entitätsklassen zusammenarbeiten.

Der Grobentwurf strukturiert das System auf der Ebene von Komponenten. Im Grobentwurf wird die oberste Ebene von Komponenten definiert. Diese Ebene bestimmt die Gesamtstruktur des Systems, seine Architektur.

Nachfolgend werden daher für die Komponenten "Telegrammdefinition", "Verbindungsparameter", "Speichern des Projekts", "Emailadressen definieren" und "Email senden" jeweils der Dialogentwurf, ein Aktivitätsdiagramm sowie ein Sequenzdiagramm vorgestellt.

2. Telegrammdefinition

2.1. Dialog

Telegrammdefinition

Definierte Telegr.:

Selektierte Telegr.: ☒ Mit 0'en auffüllen 600 Bytes

OK Abbruch

Abbildung 25: Dialog Telegrammdefinition

2.2. Aktivitätsdiagramm

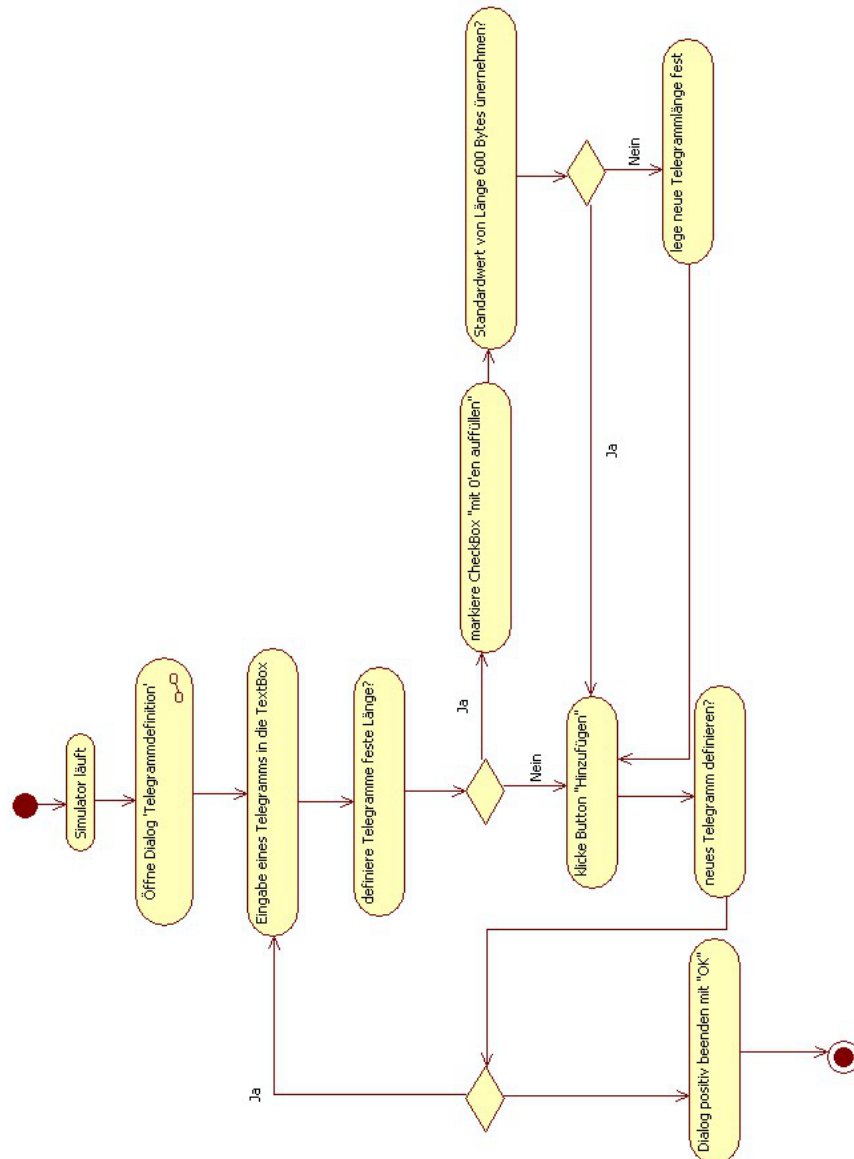


Abbildung 26: Aktivitätsdiagramm Telegrammdefinition

2.3. Sequenzdiagramm

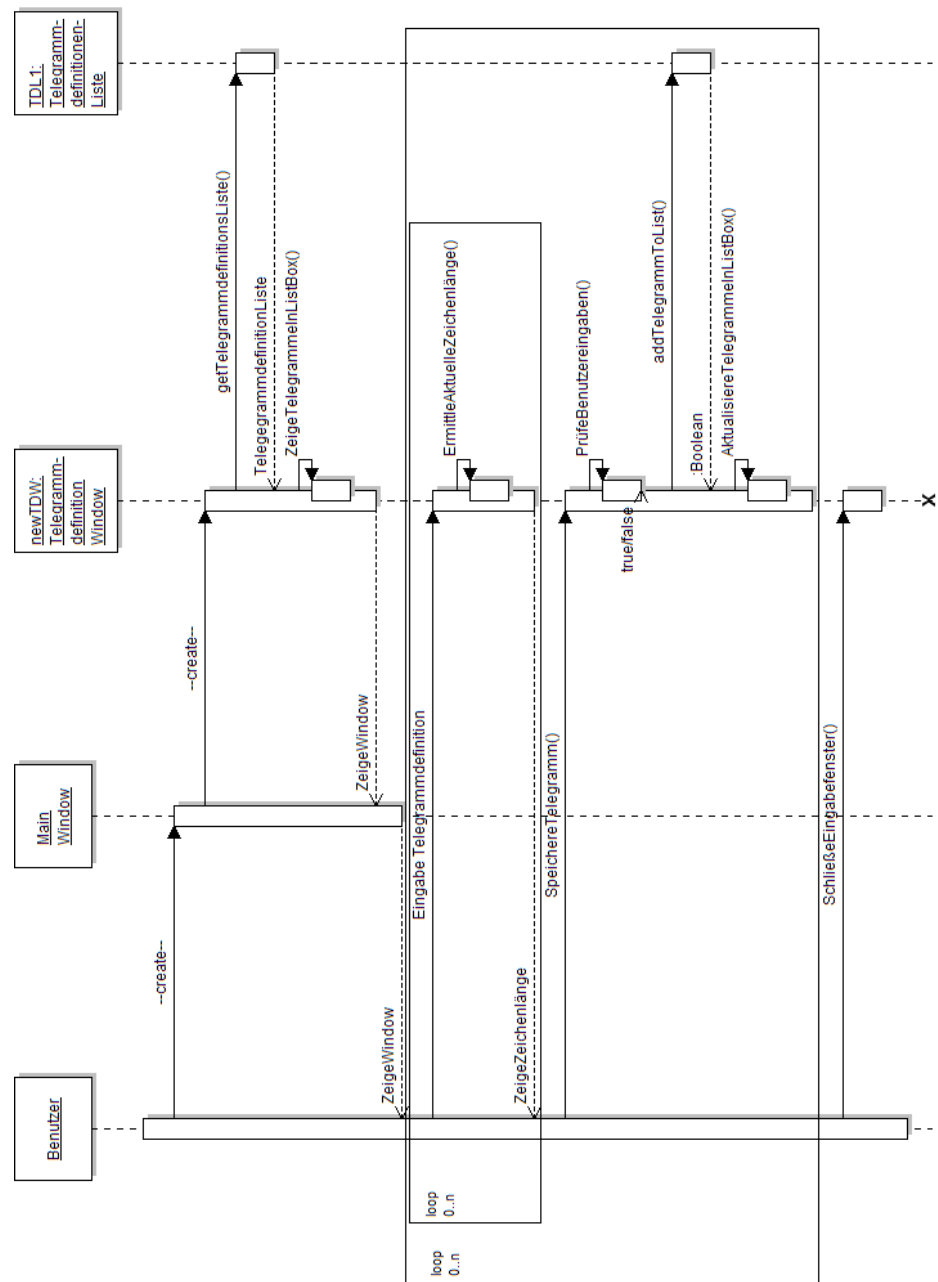


Abbildung 27: Sequenzdiagramm Telegrammdefinition

3. Verbindungsparameter

3.1. Dialog

Verbindungsparameter		—	■	✕
IP-Adresse:	Port:			
<input type="text" value="123.27.211"/>	<input type="text" value="1234"/>			
<input checked="" type="checkbox"/>	aktive Verbindung			
<input type="button" value="OK"/>		<input type="button" value="Abbruch"/>		

Abbildung 28: Dialog Verbindungsparameter

3.2. Aktivitätsdiagramm

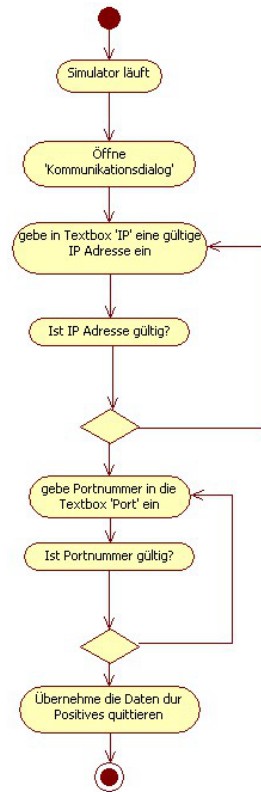


Abbildung 29: Aktivitätsdiagramm Verbindungsparameter

3.3. Sequenzdiagramm

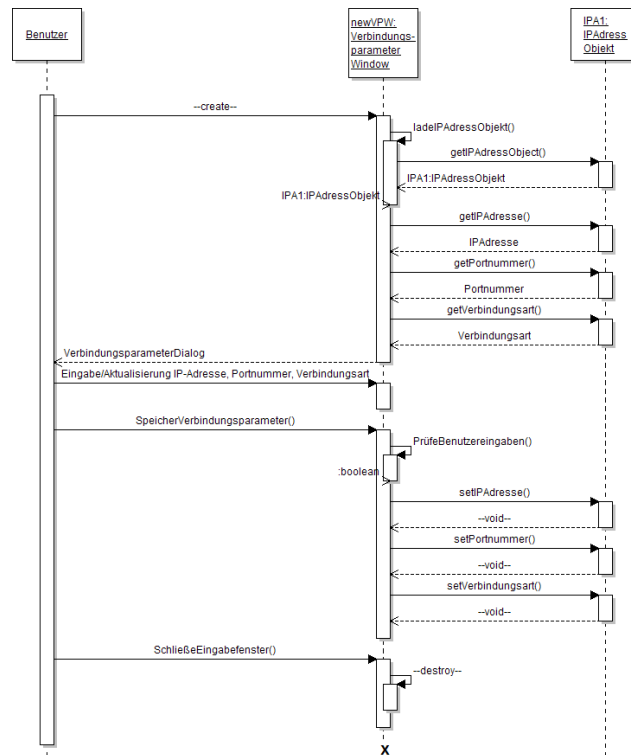


Abbildung 30: Sequenzdiagramm Verbindungsparameter

4. Speichern des Projektes

4.1. Dialog

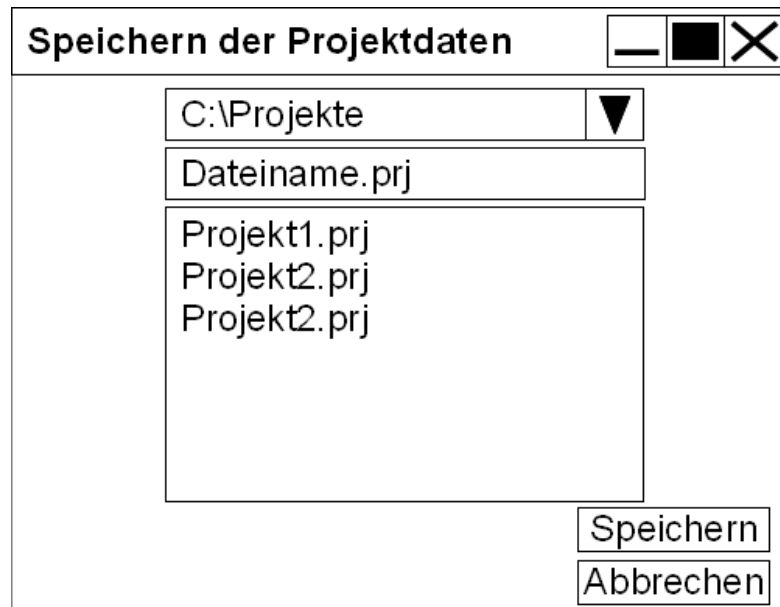


Abbildung 31: Dialog Speichern des Projektes

4.2. Aktivitätsdiagramm

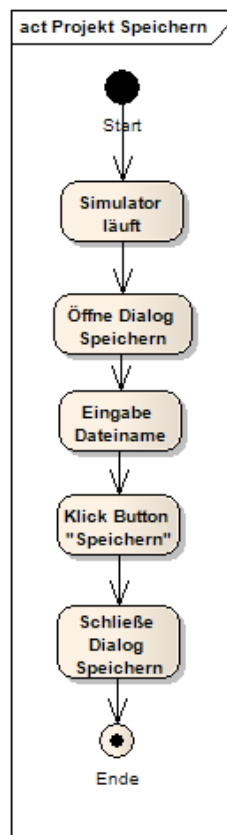


Abbildung 32: Aktivitätsdiagramm Speichern des Projektes

4.3. Sequenzdiagramm

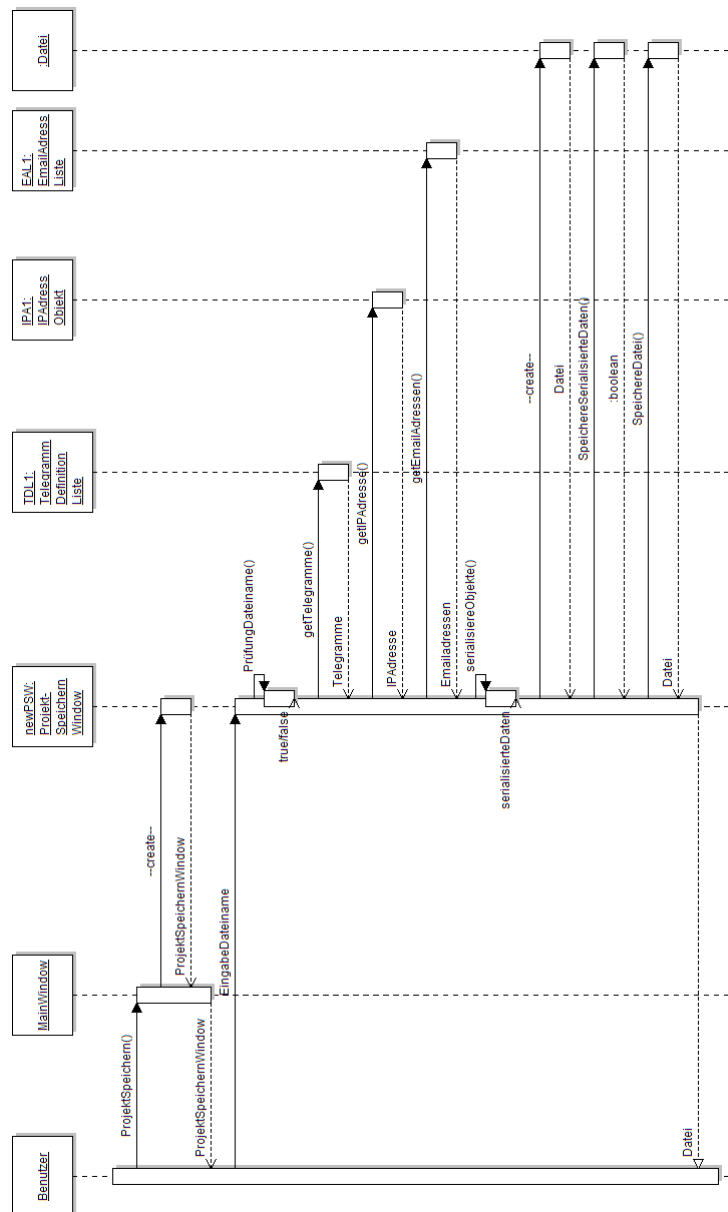


Abbildung 33: Sequenzdiagramm Speichern des Projektes

5. Email Adressen definieren

5.1. Dialog

E-Mail Adressen

E-Mail-Adresse:

emp2@mail.de

Hinzufügen

emp1@mail.de

Löschen

OK

Abbruch

Abbildung 34: Dialog Email-Adressen definieren

5.2. Aktivitätsdiagramm

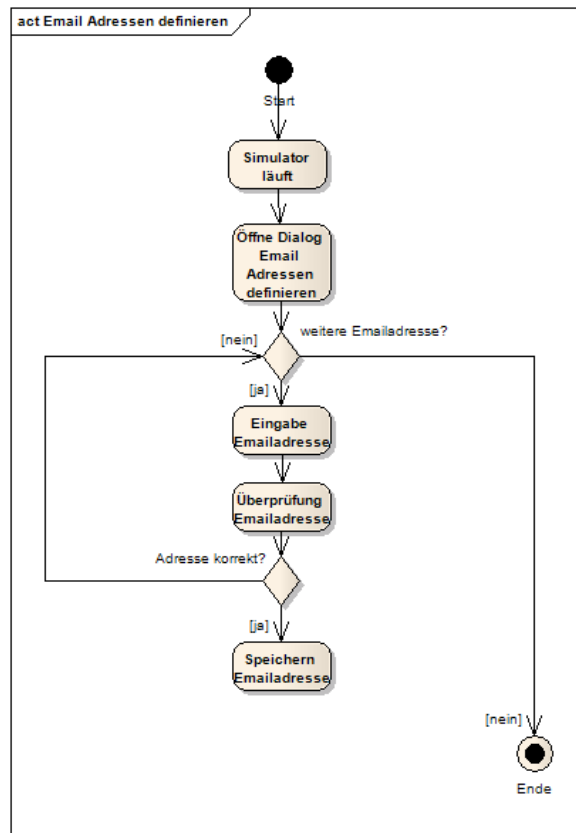


Abbildung 35: Aktivitätsdiagramm Email-Adressen definieren

5.3. Sequenzdiagramm

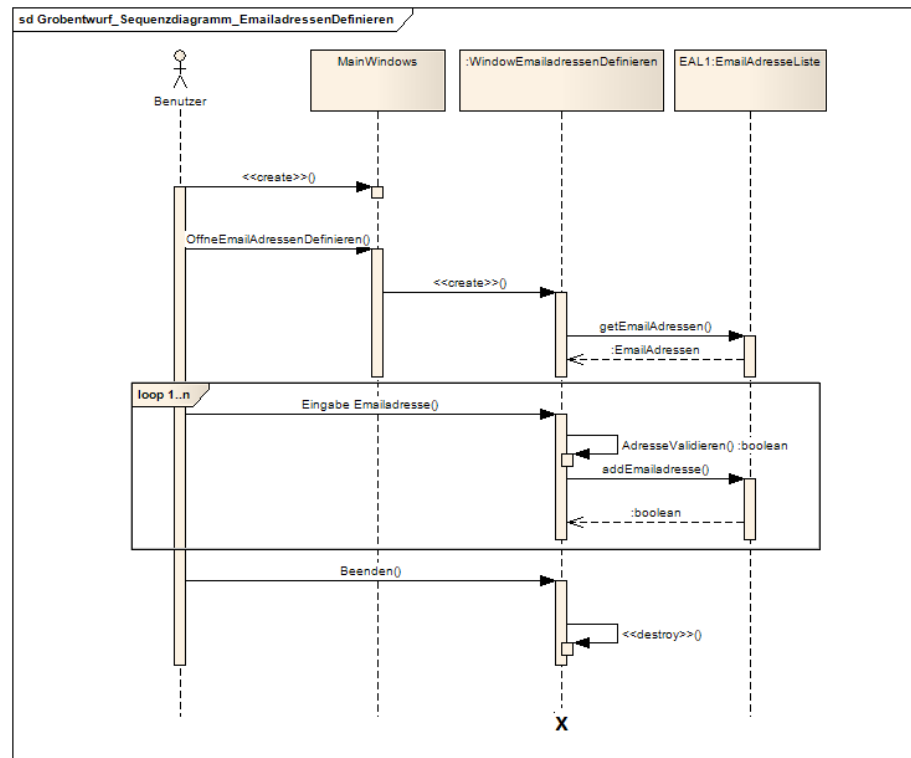
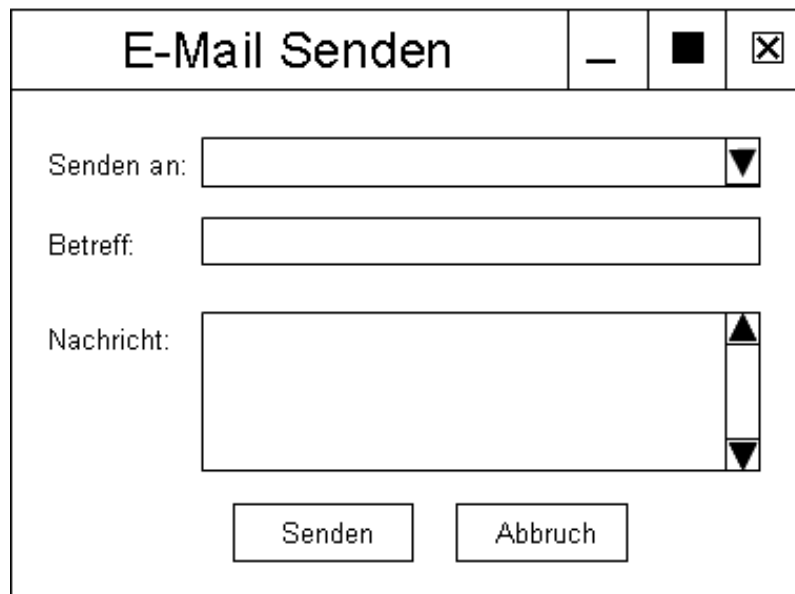


Abbildung 36: Sequenzdiagramm Email-Adressen definieren

6. Email Senden

6.1. Dialog



The image shows a standard Windows-style dialog box titled "E-Mail Senden". The title bar includes a minus sign, a maximize button (represented by a solid black square), and a close button (represented by an 'X' in a square). The main area of the dialog contains three input fields: "Senden an:" with a text box and a dropdown arrow, "Betreff:" with a text box, and "Nachricht:" with a larger text area and a vertical scrollbar. At the bottom of the dialog are two buttons: "Senden" and "Abbruch".

Abbildung 37: Dialog Email senden

6.2. Aktivitätsdiagramm

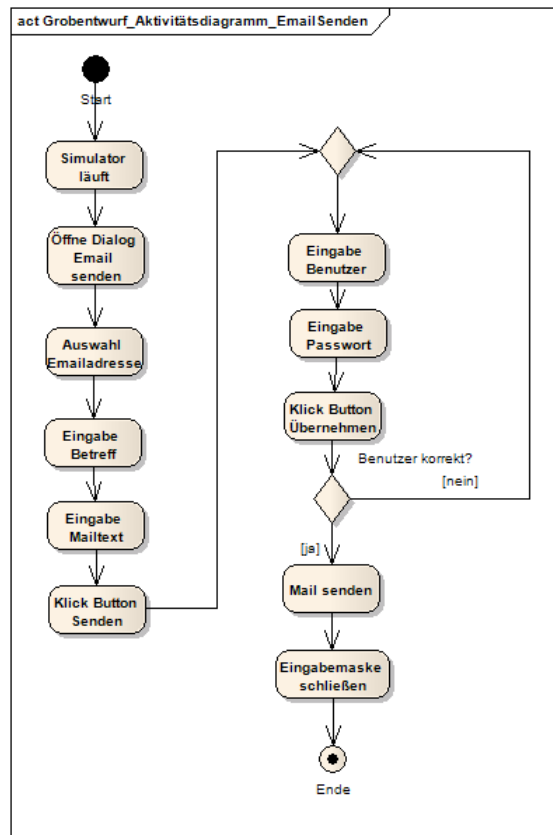


Abbildung 38: Aktivitätsdiagramm Email senden

6.3. Sequenzdiagramm

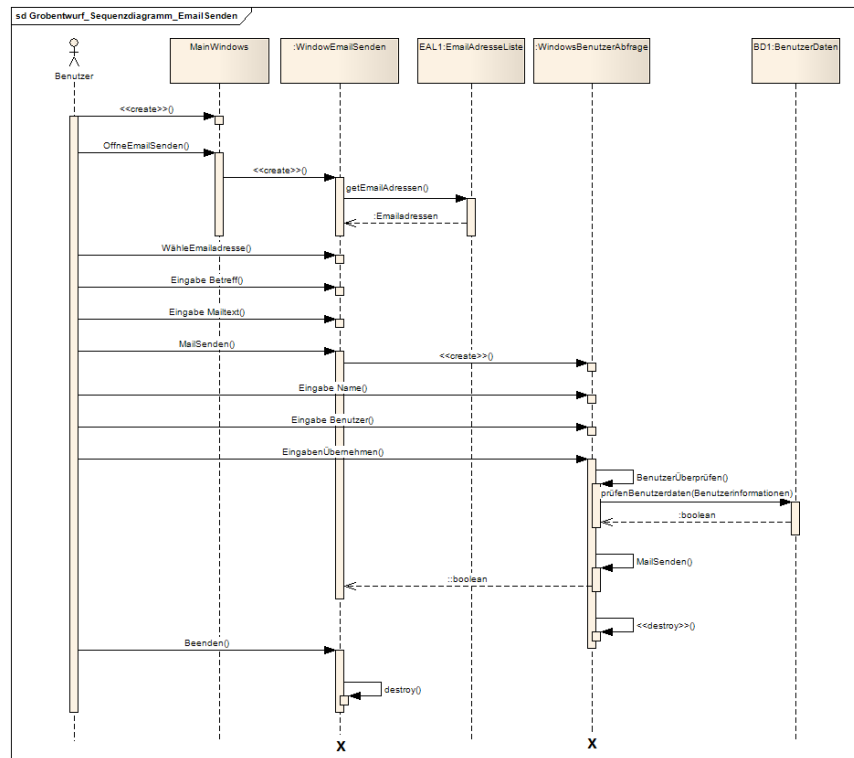


Abbildung 39: Sequenzdiagramm Email Senden

7. Erfahrungsbericht

Durch die schwierige Trennung der Erfahrungsberichte zwischen Grob- und Feinentwurf wurden beide Erfahrungsberichte im Feinentwurf zusammengefasst.

Teil V.

Feinentwurf

Versionshistorie:

Version	Datum	Was?
0.1	20.03.2010	Einstellung des Dokumentes
0.2	27.03.2010	Produktfunktionen eingearbeitet
0.3	01.05.2010	Bereinigung des Dokumentes
0.4	18.05.2010	Korrekturen nach PeerReview
0.5	17.07.2010	kleine Korrekturen

Tabelle 13: Versionsübersicht Feinentwurf

1. Einleitung Feinentwurf

Im Gegensatz zum Grobentwurf wurde im Feinentwurf nur der Anwendungsfall „Telegramme Definieren“ als Verhaltensdiagramm skizziert und wird im folgenden Kapitel abgebildet. Der Feinentwurf als systematische Verfeinerung des Grobentwurfs, berücksichtigt die angestrebte Laufzeitumgebung und die anzuwendenden Programmiersprache.

2. Programmübersicht

2.1. Hauptfenster

Unten dargestellt das Hauptfenster des Simulators (Prototyp).

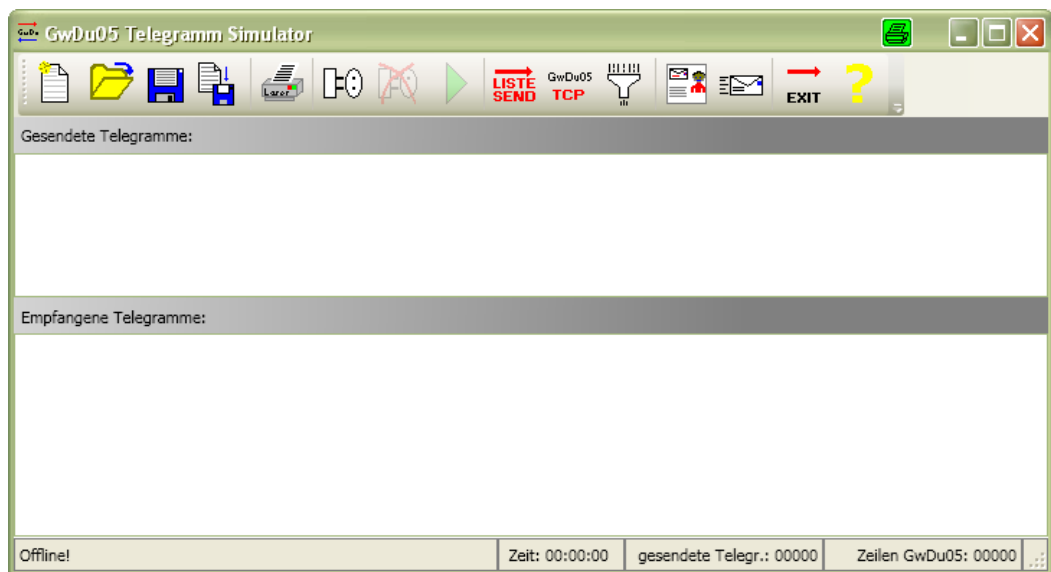
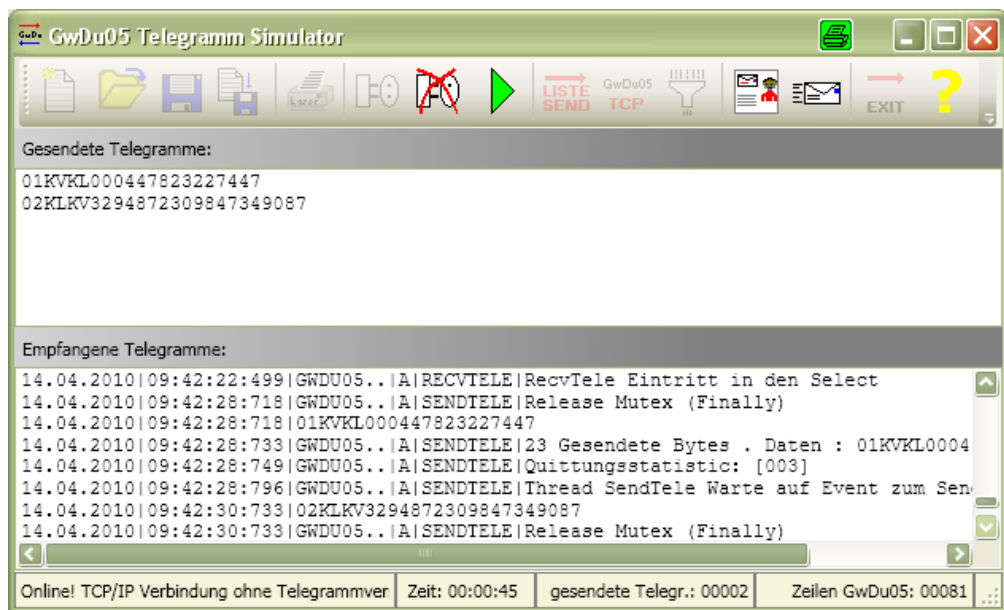


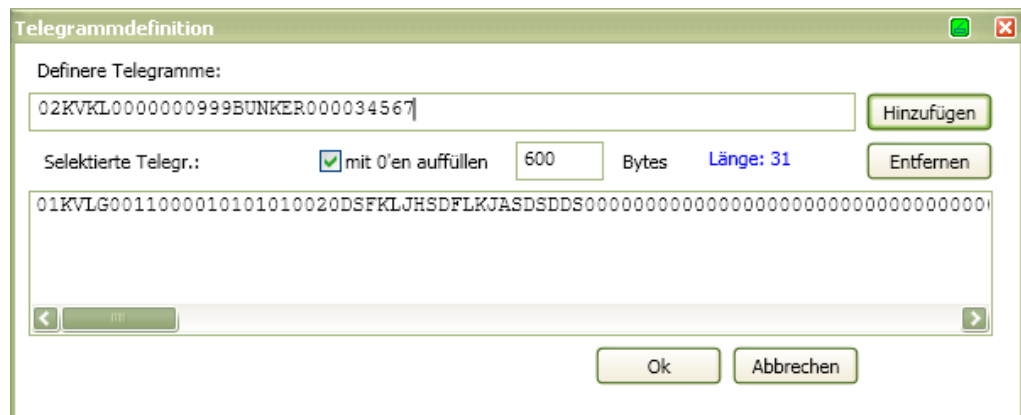
Abbildung 40: Darstellung Simulator Hauptfenster

In der zweiten Darstellung ist das Programm (Prototyp) in einer Testumgebung mit einem anderen Programm verbunden.



3. Telegrammdefinition

3.1. Dialog



3.2. Aktivitätsdiagramm

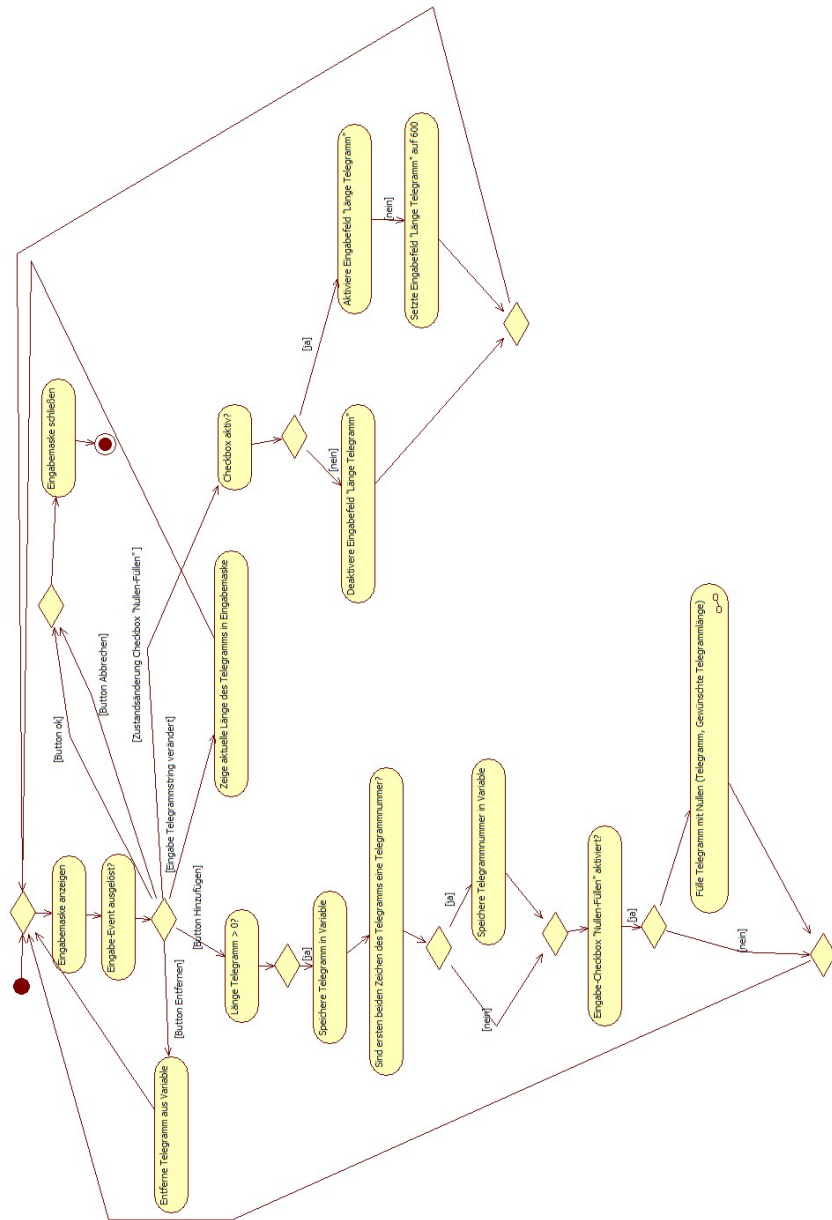


Abbildung 43: Aktivitätsdiagramm Telegrammdefinition

3.3. Sequenzdiagramm

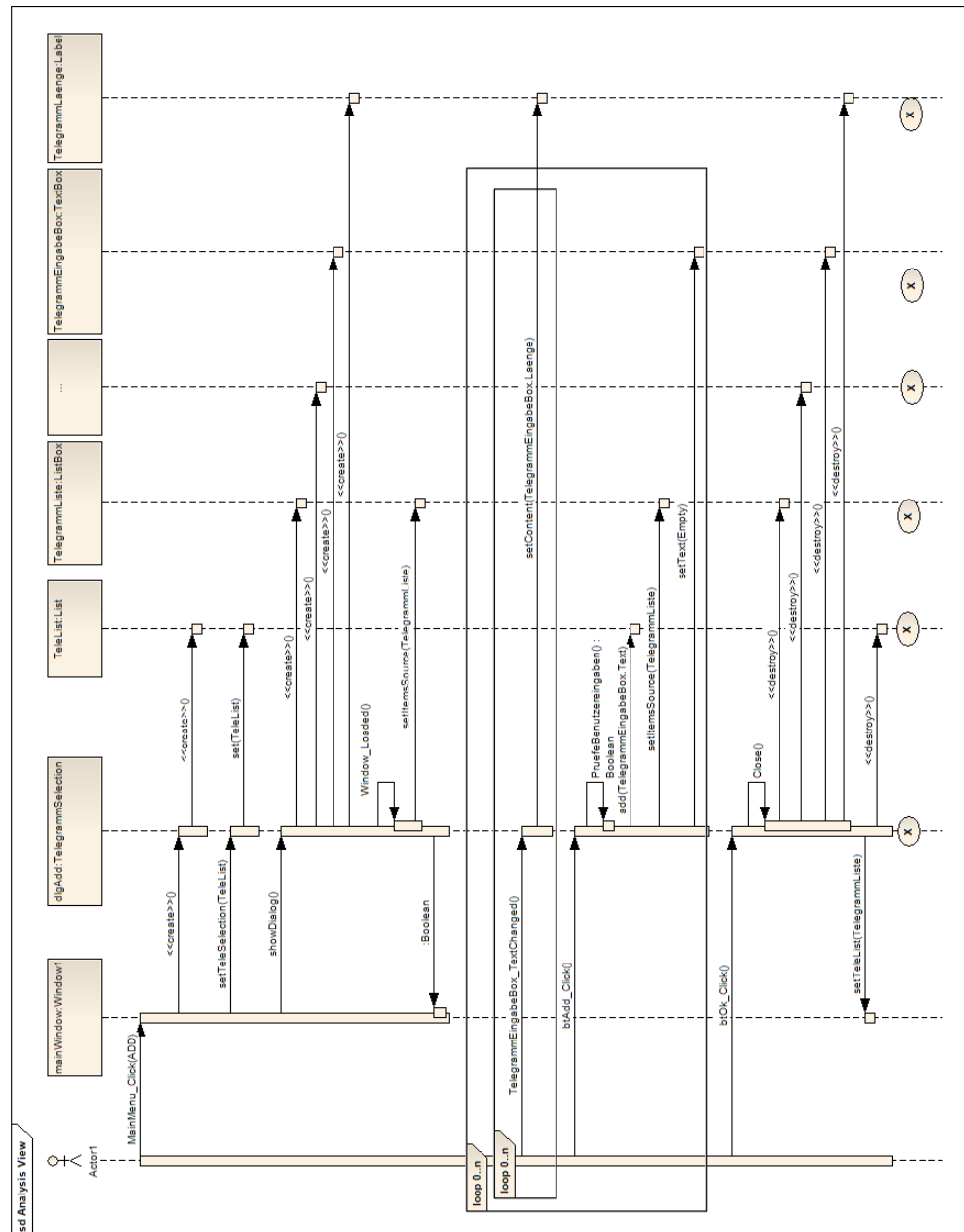


Abbildung 44: Sequenzdiagramm Telegrammdefinition

4. Verbindungsparameter

4.1. Dialog

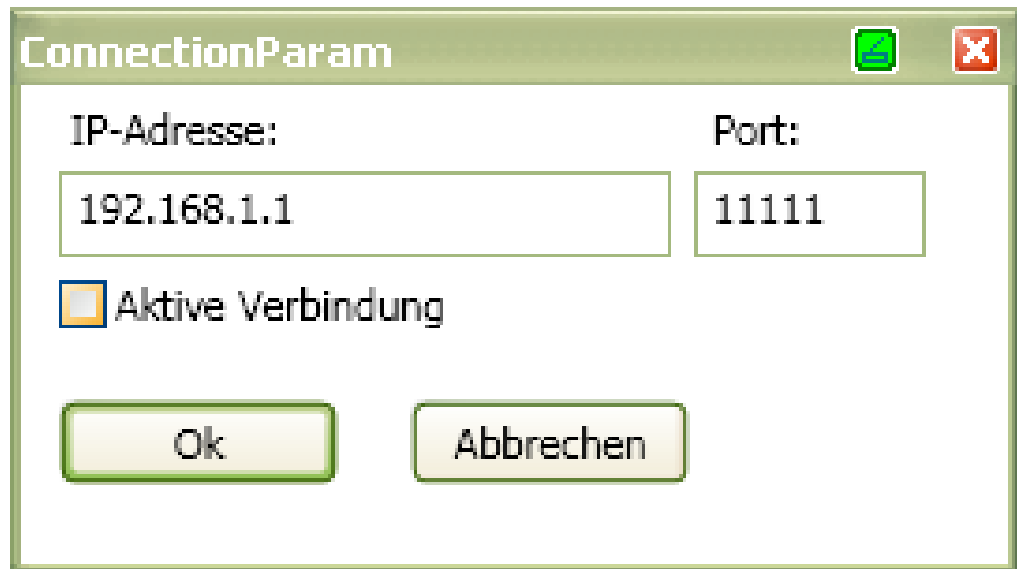


Abbildung 45: Verbindungsparameter-Dialog

5. Speichern des Projektes

5.1. Dialog

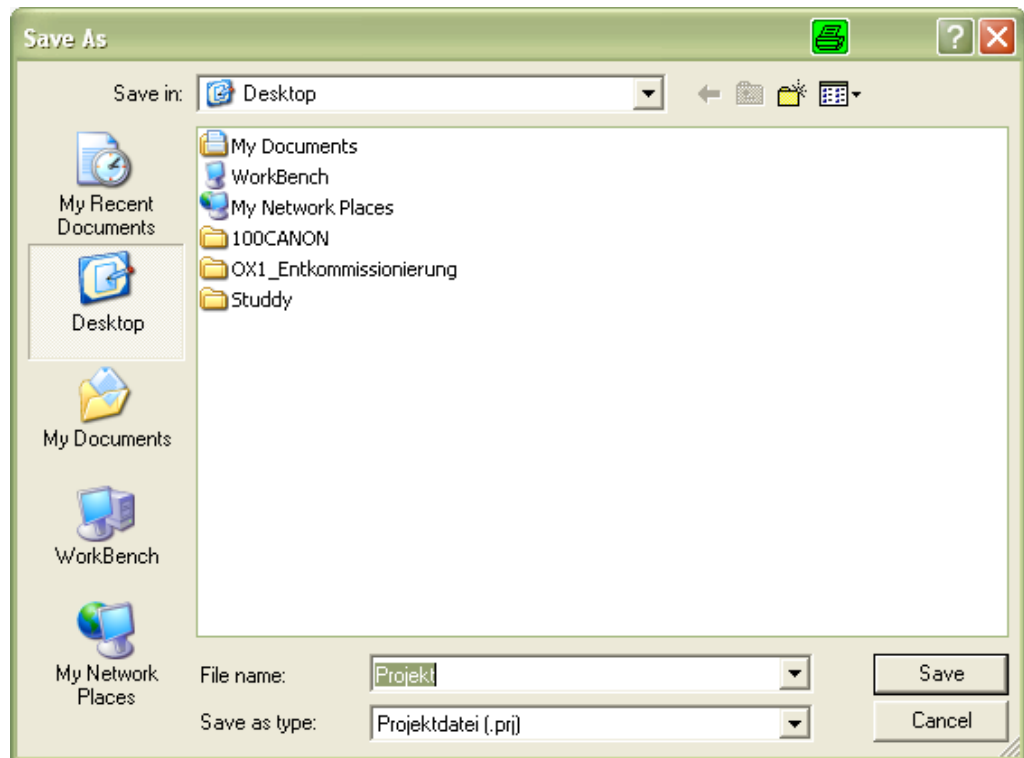


Abbildung 46: Speichern des Projektes

6. E-Mail Adressen definieren

6.1. Dialog

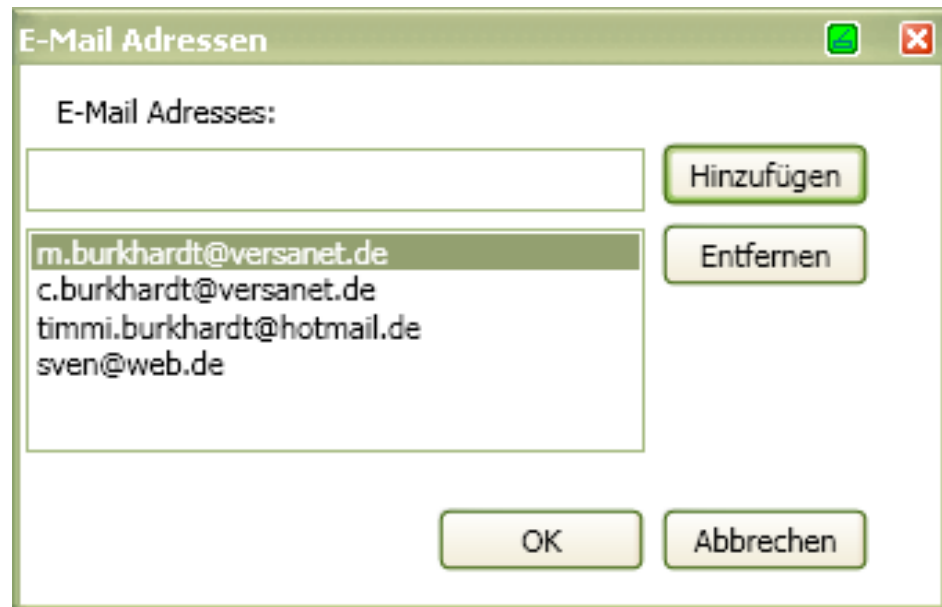


Abbildung 47: E-Mail Adressen Dialog

7. E-Mail senden

7.1. Dialog



The image shows a dialog box titled "MailSend". It has a standard Windows-style title bar with a green icon, a red close button, and a green maximize button. The dialog contains three input fields: "An:" with the value "sven@web.de" and a dropdown arrow, "Betreff:" with a single vertical bar, and "Nachricht:" with a large empty text area. To the right of the "An:" field is a "Send" button, and below it is an "Abbrechen" button.

Abbildung 48: E-Mail Adressen senden Dialog

8. Erfahrungsbericht

Nach der erfolgreichen Aufnahme der Anforderungen vom Kunden, deren Analyse und anschließender schriftlicher Fixierung im Lastenheft wurden die Kundenanforderungen weiter konkretisiert. Dazu wurde das System mit seinen Funktionen, relevanten Entitäten und dem Verhalten der Schnittstellen spezifiziert. Die Ergebnisse wurden im Pflichtenheft festgehalten. Im Grobentwurf wurde das Verhalten des Systems, für die im Pflichtenheft beschriebenen Anwendungsfälle, mit Hilfe von Aktivitätsdiagrammen modelliert. Mit Hilfe der Diagramme wird die Vernetzung der elementaren Aktionen und deren Verbindung mit Kontroll- und Datenflüssen grafisch dargestellt. Zusätzlich wurden die Anwendungsfälle in Sequenzdiagrammen modelliert. Sie stellen die Interaktion der Objekte und deren Nachrichtenaustausch sowie die zeitliche Ordnung der Ereignisauftritte dar. Es wurde jeweils der positive Ablauf des Anwendungsfalles durch das Sequenzdiagramm beschrieben. Der Grobentwurf erfolgte unabhängig von Laufzeitumgebung und der Programmiersprache. Neben der Modellierung des Systemverhaltens durch UML-Diagramme wurden die Eingabemasken der Benutzeroberfläche in einem ersten Entwurf grafisch dargestellt.

Im Feinentwurf wurden die erstellten UML-Diagramme weiter

konkretisiert und an die Gegebenheiten der Laufzeitumgebung sowie der Programmiersprache angepasst. Durch die sehr detaillierte Beschreibung der einzelnen Anwendungsfälle durch die UML-Diagramme kann anschließend die Codierung erfolgen. Der Feinentwurf zeigt zusätzlich die endgültigen Versionen der Eingabemasken des Programms. Die Phase des Grob- und Feinentwurfs war eine große Herausforderung für das Projektteam. Zunächst mussten wir klären welche Diagrammtypen erstellt werden sollten und welche zum Grob- beziehungsweise Feinentwurf zählen. Die Analyse ergab, dass der Grob- wie auch Feinentwurf im Wesentlichen auf UML-Diagrammen basiert. Im Projektteam war zwar Wissen diesbezüglich vorhanden, doch lag dieses teilweise schon länger zurück, so dass ein erneuter Einarbeitungsaufwand inkl. Vertiefung der UML Kenntnisse notwendig war. Bedingt durch beruflichen Stress und der Doppelbelastung durch Arbeit und Studium verzögerte sich die Einarbeitung in UML und die eigentliche Durchführung der Phase immer mehr. Zudem mussten wir feststellen, dass die Erstellung der Diagramme deutlich mehr Zeit in Anspruch nahm, als wir zuvor erwartet und geplant hatten. Durch bereits geplante Urlaubszeiten und des sehr straffen Zeitplans des Moduls „Fortgeschrittene Softwaretechnologie“ waren im Projektplan kaum Änderungen möglich. Die noch zur Verfügung stehenden Zeitpuffer wurden in dieser Phase maximal ausgereizt,

so dass die eingeplante Lernphase vor den Klausuren stark gefährdet war und auch einige Teammitglieder weniger Zeit in die Klausurvorbereitung investieren konnten als sie geplant hatten.

Die Phase bewegte sich, wie eben beschrieben, auf einem kritischen Projektpfad. Die Phase war sehr zeitaufwendig und die Zeit war knapp. Zusätzlich wollte das Team, nach den langwierigen Phasen der Anforderungsaufnahme und erster Systemspezifikation im Lasten- und Pflichtenheft, am liebsten mit der Programmierung starten, anstatt das System noch feiner zu spezifizieren und zu dokumentieren. Nach Abschluss der Phase wurde dem Team allerdings auch deutlich, wie wichtig die genaue Spezifikation der Anwendungsfälle für die anschließende Codierung war. Durch die sehr detaillierte Verhaltensbeschreibung und die klare Spezifikation der Schnittstellen konnten große Teile einfach in Code umgesetzt werden.

Teil VI.

Implementierung

Versionshistorie:

Version	Datum	Was?
0.1	20.03.2010	Einstellung des Dokumentes
0.2	27.03.2010	Produktfunktionen eingearbeitet
0.3	01.05.2010	Bereinigung des Dokumentes
0.4	18.05.2010	Korrekturen nach PeerReview
0.5	17.07.2010	kleine Korrekturen

Tabelle 14: Versionsübersicht Implementierung

1. Einleitung Implementierung

Die nachfolgend dargestellte Implementierung beschreibt die Vorgaben, Möglichkeiten, Installation und Aufbau von Testumgebungen zur Nutzung der eingesetzten Softwarekomponente. Basierend auf dem gewählten Vorgehensmodell SCRUM wurden Implementierung anhand der Sprints umgesetzt. Wie auch schon in vorherigen Kapiteln erläutert, handelt sich bei diesem Projekt nicht nur um ein Studienprojekt sondern einer bei der TKSE angeforderten Software. Aus diesem Grund war die Projektgruppe nicht immer frei in Ihren Entscheidungen. Viele Vorgaben bzgl. der Anforderungen, Gestaltung und Design, Programmiersprache und Entwicklungsumgebung wurde durch TKSE gestellt. Der Prototyp wurde von allen Projekt-Mitgliedern gleichermaßen erstellt. Die Implementierung in die Release Version geschieht weiterhin mit dieser Projektgruppe, allerdings mit einem Code-Review durch TKSE Interne Entwickler. Hier kann es sein, dass Anforderungen oder Code-Snippets geändert werden müssen!

1.1. Produkt-Backlog

Hier der Produkt-Backlog mit Vergabe der einzelnen Sprints.

Planung Sprints zur Erstellung des Simulators														
Anforderungen		Sprint												
		1	2	3	4	5	6	7	8	9	10			
Anforderungs Nr.	Planungspunkte bis zum Release	10	10	10	20	10	10	10	10	10	10			
AF10.0	Grafische Oberfläche zur Benutzung des Programms	6	4											
AF10.1	Bedienung auch für fachfremdes Personal	3	3	4										
AF10.2	Dialog zur Eingabe der Parameter zum Verbindungsaufbau													
AF10.3	Dialog zur Definition von Telegrammen	2	2	2	2	2								
AF10.4	Dialog zur Definition von E-Mail Adressen													
AF10.5	Dialog zum versenden von E-Mails													
AF20.1	Socket Verbindung über TCP/IP													
AF20.2	Siemens DU05 Prozedur mit dem Kommunikationspartner													
AF20.3	freie Wahl des Kommunikationspartner													
AF20.4	Festlegung der Verbindungs-Art Aktiv													
AF20.5	Festlegung der Verbindungs-Art Passiv													
AF30.1	Länge des Telegramms													
AF30.3	Quittungstelegramme													
AF40.1	Speichern von Projekt-Parameter													
AF40.2	Laden von Projekt-Parameter													
AF50.1	Bereitstellung von Kommunikationsverlaufsinformationen													
AF50.2	Senden von Telegrammen													
AF50.3	Empfangen von Telegrammen													
AF60.1	protokollieren des Kommunikationsverlaufes													
AF60.2	drucken des Kommunikationsverlaufes													
AF60.3	speichern des Kommunikationsverlaufes													
AF60.4	E-Mail Versand des Kommunikationsverlaufes													
AF70.1	Drucken während Kommunikationsverlauf nicht erlaubt													
AF70.3	Speichern während Kommunikationsverlauf nicht erlaubt													
AF70.4	E-Mail Versand während Kommunikationsverlauf nicht erlaubt													
AF70.5	Telegramme definieren während Kommunikationsverlauf erlaubt													
AF70.5	IP-Adresse definieren während Kommunikationsverlauf nicht erlaubt													

Abbildung 49: Übersicht Product-Backlog

1.2. Systemumgebung

Das Programm wird durch die im Umfeld der TKSE bestehenden Infrastruktur, in einer Microsoft Windows XP basierten Umgebung eingesetzt. Auch externe Entwickler oder Inbetriebnehmer sind gezwungen sich an diese Infrastruktur zu halten! Das Microsoft Framework 3.0 oder höher muss zwingend auf dem Rechner installiert sein!

1.3. Netzwerkumgebung

Die Netz-Infrastruktur bei der TKSE wird durch die Externe Firma Hewlett Packard (HP) administriert. Bestehend aus einem Ethernet mit Switchen und Hubs.

1.4. Kommunikation

Die Kommunikation mit den Partner-Systemen, geschieht über TCP/IP und der DU05 Prozedur.

1.5. Verbindungsaufbau Testumgebung

Der Aufbau einer Testumgebung sollte folgendermaßen erfolgen:

- Klärung ob der Simulator mit Aktiver oder Passiver Verbindung eingesetzt werden soll
- der gewählte Port sollte ein Well-Known Port sein und nicht durch andere Kommunikations-Software belegt sein
- Telegramme sollten über den Telegramm-Definitions-Dialog eingegeben sein oder aus einer bereits auf der Festplatte oder einem anderen Speichermedium befindlichen Datei gelesen werden
- wenn der Partner ebenfalls parametrisiert ist kann über die entsprechende Schaltfläche die Verbindung aufnehmen oder in den Verbindungs-Wartezustand gehen

1.6. Partnersysteme

Die DU05 basierenden Partnersysteme bestehen aus Produktions-Anlagen, Siemens SPS-Anlagen oder auf andere Softwarekomponenten die den Siemens DU05 Standard implementiert haben.

1.7. Installation

Wie schon an anderer Stelle beschrieben, wird die Programmdatei und die dazugehörigen Assemblies einfach in Verzeichnis auf dem Rechner kopiert, und durch einen Doppelklick, auf die Programmdatei, gestartet.

Teil VII.

Prototyp

1. Beschreibung Prototyp

Der Prototyp des DU05 Simulators beinhaltet in erster Linie die Benutzeroberfläche. Jedoch kann schon eine Kommunikation aufgebaut werden. Der Aufbau funktioniert folgendermaßen:

1. das Programm wird zweimal gestartet
2. im *ersten* Programm wird unter Verbindung die IP Adresse des Rechners eingetragen. Sollte keine IP Adresse vergeben sein ist dort die IP des Localhost 127.0.0.1 einzutragen.
3. im selben Dialog wird ein freier nicht Well-Known-Port vergeben z.B. 20000
4. die Art der Verbindung wird auf 'Aktiv' gesetzt.
5. danach wird der Dialog mit OK geschlossen.
6. nun werden im Telegrammdefinitionsdialog Telegramme definiert und der Dialog mit OK quittiert
7. im *zweiten* Programm wird im Dialog Verbindung auch die IP-Adresse und der Port eingetragen die schon im *ersten* Programm vergeben wurde.
8. hier wird die Verbindung Art allerdings auf 'Passiv' belas-

sen.

9. der Dialog wird mit OK quittiert
10. nun werden beide Programm über den Button 'Verbindung herstellen' zur Verbindung zum Partner aufgefordert.
11. nachdem die Programm die Verbindung aufgebaut haben, kann aus dem *ersten* Programm heraus die Telegramme gesendet werden, indem der Button mit dem grünen Pfeil betätigt wird.

Des Weiteren können einige Dialog aufgerufen werden, jedoch steckt noch keine Funktionalität hinter diesen.

2. Installationsvoraussetzung

Auf dem Rechner muss das Microsoft® Framework 3.5 installiert sein. Des weiteren muss die SocketClass.dll Assembly im gleichen Verzeichnis liegen wie die eigentliche EXE Datei. Das Programm setzt keine Installation voraus, da auch keine Registry-Einträge notwendig sind. Das benötigte Framework 3.5 SP1 kann unter folgendem Link aus dem Netz herunter geladen werden: [Microsoft Framework 3.5 SP1](#)

Nach Installation des Framework ist kein Neustart des Rech-

ners nötig, und das Programm DU05 Simulator kann gestartet werden.

3. Start des Programms

Nach erfolgreicher installation des Microsoft Frameworks 3.5 SP1 kann nun mit dem Start des Programms begonnen werden. Die Dateien *SocketClass.dll* und *WpfDu05Simulator.exe* in ein beliebiges Verzeichnis kopieren.

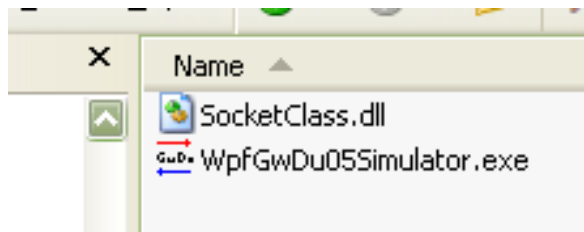


Abbildung 50: Prototyp: Information Startordner

Die Datei *WpfDu05Simulator.exe* mit einem Doppelklick starten.

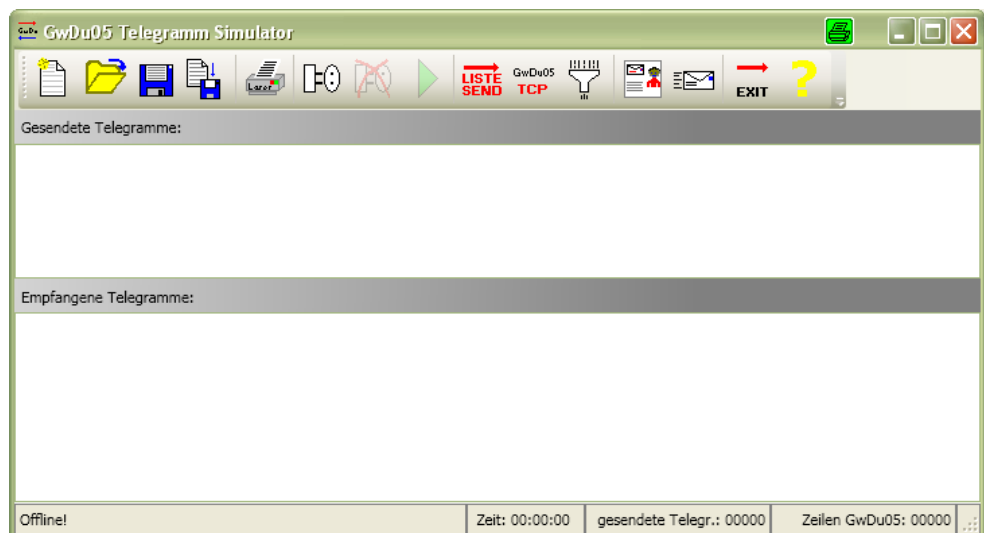


Abbildung 51: Prototyp: Programm DU05 Simulator

Nun kann über die Symbolleiste jeder Dialog aufgerufen wer-

den. Jedoch ist zu berücksichtigen das nicht hinter jedem Button eine Funktionalität in diesem Prototypen steckt.

Folgende Funktionen sind hinter den Icons schon realisiert.



Abbildung 52: Prototyp: Aussehen der Symbolleiste

Erklärung der Icon's:

- Icon 1: Neues Projekt anlegen
- Icon 2: Öffnen eines vorhanden Projekts
- Icon 3: Speichern eines neuen oder geänderten Projekts
- Icon 4: Speichern des Kommunikationsverlaufs in einen Datei
- Icon 5: Drucken des Kommunikationsverlaufs
- Icon 6: Verbinden zum angegebenen Partner
- Icon 7: Verbindung vom Partner beenden
- Icon 8: Definierte Telegramme im Zyklus senden
- Icon 9: Dialog Telegramme definieren
- Icon 10: Dialog Verbindungsparameter eingeben
- Icon 11: Dialog Einstellung für den Informationsfilter
- Icon 12: Dialog E-Mail-Adressen eingeben

- Icon 13: Dialog senden einer E-Mail
- Icon 14: Programm verlassen (automatisches beenden der Verbindung incl.)
- Icon 15: Dialog Programminformation

3.1. Dialog E-Mail Adressen

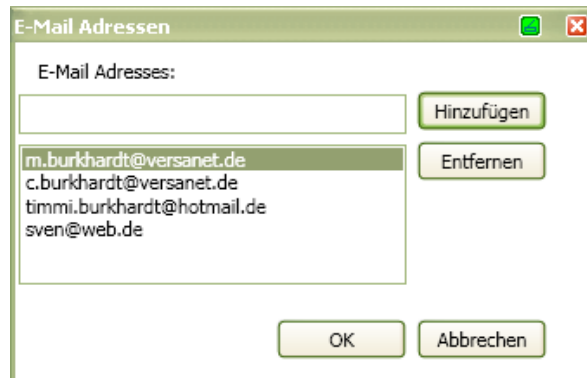


Abbildung 53: Prototyp: Dialog Eingabe E-Mail Adressen

3.2. Dialog Verbindungsparameter

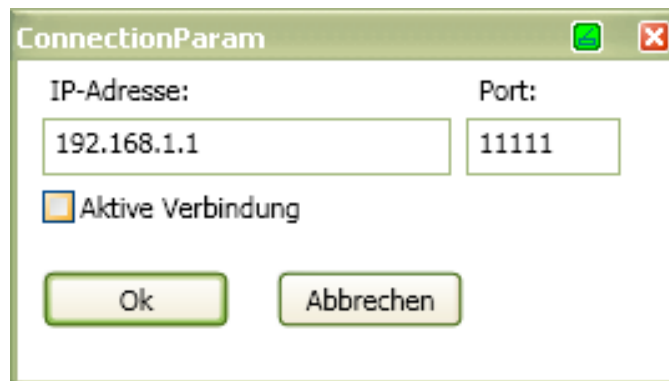


Abbildung 54: Prototyp: Dialog Verbindungsparameter

3.3. Dialog sende Mail



MailSend

An: sven@web.de

Betreff: |

Nachricht:

Send

Abbrechen

Abbildung 55: Prototyp: Dialog E-Mail Versand

3.4. Dialog Telegramm definition

[illegible]

Abbildung 56: Prototyp: Dialog Telegramm definition

4. Deinstallation

Um das Programm (Prototyp) wieder von seinem Rechner zu entfernen, muss lediglich das Programm beendet und die Dateien inklusive Programm Ordner von der Festplatte gelöscht werden.

Teil VIII.

Abschluss

1. Fazit

An dieser Stelle wollten wir uns mal allgemein zum gesamten Modul äußern. Wir haben festgestellt, dass die Vorgehensweise für Softwareentwicklung nicht darin besteht eine gute Idee zu haben oder zwischen Tür und Angel, mitgeteilte Anforderungen umzusetzen, sondern vergleichbar wie mit einem Hausbau, eine saubere Planung zu erstellen. Niemand stellt sich in der Regel mit einem Spaten bewaffnet auf eine grüne Wiese und fängt an sein Haus zu bauen, ohne einen Plan, Architekten oder Fachpersonal zu haben. Wir kannten zwar Vorgehensmodelle, haben aber, wie vielleicht in vielen Unternehmen üblich, in der Regel uns Gedanken zum Software-Produkt gemacht während wir vielleicht schon die ersten Code-Zeilen geschrieben haben. Leider erst viel zu spät bemerkten wir, dass wir viele Anforderungen noch nicht kennen und diese vielleicht mit anderen Anforderungen interagieren die wir zu keinem Zeitpunkt berücksichtigt haben. Dies führt in der Regel zu erhöhtem Zeitaufwand und lässt die Qualität und Stabilität deutlich sinken.

In diesem Modul *Erweiterte Softwaretechnologien* wurde uns erst so richtig bewusst, wie arbeitsintensiv und feingranular eine Software-Entwicklung sein kann ohne nur eine Code-Zeile geschrieben zu haben! Bevor festgelegt wird in welcher Pro-

grammiersprache die Software entstehen soll, werden unzählige Kommunikationswege erstellt, Meetings und Diskussionen abgehalten, Projektplanungen erstellt, Stakeholder mehrfach befragt. Jedoch muss an dieser Stelle erwähnt werden dass wir schnelle erkannt haben das es sich in jedem Fall lohnt sich die Mühen zu machend. Die Vorteile wie die relativ leichte Erweiterbarkeit, Änderungsmanagement, Stabilität, Wartbarkeit, Pflege und letztlich das Coding spricht eine deutliche auf der Hand, und dies sind nur einige Punkte die hier genannt werden. Durch die vorhandenen Artefakte befindet sich der Entwickler letztlich in einer ausgezeichneten Position, in der er, wie Anhand eines Planes, die Software *runter programmieren* kann. Nicht zuletzt die rechtlichen und vertraglichen Vorteile die Ihm im Falle von Mängeln (die eigentlich keine sind) oder aus Sicht des Kunden fehlende Anforderungen durch die erstellten Artefakte jederzeit belegbar sind.

Als Fazit können wir über das Modul *Erweiterte Softwaretechnologien* sagen, das es nicht nur sehr Interessant, sondern auch sehr Praxis bezogen und Informativ ist. Es reiht sich Nahtlos in den Diplom- oder Bachelor-Studiengang ein. Des Weiteren muss erwähnt werden, dass die von Ihnen bereitgestellten Lerneinheiten den absolut aktuellsten Stand haben und diese in unseren Unternehmen als Nachschlagewerk dienen. Dies ist

in anderen Modulen leider nicht immer so.

Nun möchten wir die Gelegenheit nutzen und kreative Kritik äussern. Es wäre für die Zukunft wünschenswert, ausgesprochene oder durch schriftliche Mitteilungen bekannt gegebene Vorgaben nicht zu kurzfristig zu ändern. Es war von Anfang an bekannt, dass wir 6 Gruppen sind und 20 min. für die Präsentation geplant sind. Darauf haben alle Gruppen hingearbeitet. Mit der letzten Mail Ihrerseits (3 - 4 Tage vor der Präsentation) wurde das Zeitfenster auf 60 min. Vor Beginn der Präsentation wurde das Zeitfenster wieder auf 30 - 45 min. gekürzt. Uns sehr wohl bewusst, dass es im wahren Berufsleben eben oft genau so vorkommt und man sich oft mit solchen Änderungen konfrontiert sieht. Jedoch geht es hier um Noten mit denen sich die Studenten eventuell bewerben müssen. Des Weiteren ist mir aufgefallen, dass viele Projekte **nie** zur Realisierung vorgesehen waren! Das erschwert viele Bereiche der Hausarbeit, da die Kommilitonen nicht erkennen können (zumindest eine Vielzahl) dass Sie sich mit dem Projekt eventuell übernommen haben.

2. Ausblick

Das Projekt ist noch nicht komplett abgeschlossen, wird aber für ein nun anstehendes großes Neu-Projekt im Oxygenstahlwerk II dringend benötigt. Dort werden neue Daten-Verteiler eingebaut die Telegramme im Siemens DU05 Prozedurstandard senden und empfangen. Hier ist der Simulator geradezu prädestiniert, denn die neuen Daten-Verteiler werden in der gesamten Produktionsumgebung implementiert. Wenn es diesen Simulator nicht gäbe, würden nun alle vorhandenen Systeme im Telegrammverkehr testweise erweitert um die Daten-Verteiler zu testen. Hier wären Störungen vorprogrammiert.

Meine Kommilitonen haben zugesagt das Projekt weiter in Ihren Möglichkeiten zu begleiten und tatkräftig daran mit zu entwickeln. Die Artefakte werden weiterhin erweitert und gepflegt.

Hiermit bedankt sich die Gruppe 2 für die gute Zusammenarbeit!

Teil IX.

Anhang

1. Anhang

1.1. Meeting-Protokolle

Besprechungsprotokoll

Besprechung:	Fortgeschrittene Softwaretechnologie SS 2010 Gruppe 2: DU05 Simulator bei der ThyssenKrupp Steel Europe AG		
Zielsetzung:			
Frequenz:	Alle 14 Tage	Datum:	08.04.2010
Initiator:	Michael Burkhardt	Uhrzeit (von .. bis):	18:00 – 21:00
Protokollant:	Stefan Albert	Ort:	Dortmund
Anwesende:	Michael Burkhardt ; Sven Münzner ; Michael Rongen (per Telefon); Thomas Baier Stefan Albert		
Verteiler:	Anwesende		
Vertraulichkeit:	Erstellt / Geändert: 07.04.2010		

Agenda

TOP	Thema
1	Reflexion Vorlesung etc.
2	Aktueller Stand Lastenheft
3	Aktueller Stand Pflichtenheft
4	Aktueller Stand Grobentwurf
5	Aktueller Stand Feinentwurf
6	Terminplanung

Ergebnisse

Lfd. Nr.	Zu TOP	Typ <small>Aktion Beschluss Information</small>	Beschreibung	Verantwortlich <small>(Name oder Initialen)</small>	Fällig <small>(Datum)</small>
1	1	I	Keine besonderen Fragen	alle	
2	2	I	Erste Überarbeitung des gesamt Textes ist heute abgeschlossen. Die einzelnen Muss-Anforderungen wurden noch mal diskutiert.	alle	
3	2	A	Neue Version wird morgen per Mail verteilt	MB	07.04.10
4	3	I	Die Use Cases wurden noch mal im Detail besprochen	alle	
5	3	B	Z.Zt. keine weitere Anpassung des Pflichtenheftes	alle	
6	4	I	Produktfunktionen wurden eingearbeitet	alle	
7	5	I	Produktfunktionen wurden eingearbeitet	alle	
8	6	I	Neuer Termin am 29.04.2010	alle	

Abbildung 57: Meeting Protokoll 1

Besprechungsprotokoll

Besprechung:	Fortgeschrittene Softwaretechnologie SS 2010		
	Gruppe 2: DU05 Simulator bei der ThyssenKrupp Steel Europe AG		
Zielsetzung:			
Frequenz:	Alle 14 Tage	Datum:	29.04.2010
Initiator:	Michael Burkhardt	Uhrzeit (von .. bis):	18:00 – 21:00
Protokollant:	Stefan Albert	Ort:	Dortmund
Anwesende:	Michael Burkhardt ; Sven Münzner ; Michael Rongen (per Telefon); Thomas Baier Stefan Albert		
Verteiler:	Anwesende		
Vertraulichkeit:	Erstellt / Geändert: 30.04.2010		

Agenda

TOP	Thema
1	Feedback Kurzpräsentationen
2	Vorbereitung Peer Review
3	Grundkonzept Präsentation
4	Terminplanung

Ergebnisse

Lfd. Nr.	Zu TOP	Typ <small>Aktion Beschluss Information</small>	Beschreibung	Verantwortlich <small>(Name oder Initialen)</small>	Fällig <small>(Datum)</small>
1	1	I	Offene Diskussion über die gehaltene Kurzpräsentationen vom 24.04.10	alle	
2	1	B	Bei der Abschlusspräsentation sollen die geforderten Bewertungsrichtlinien wie „Metadaten“ etc. mehr beachtet werden. Zudem soll die Präsentation mehr Schaubilder enthalten, um das Projekt besser visualisieren zu können. Wie der Prototyp präsentiert werden soll ist noch offen.	alle	
2	2	I	Informationen zum Peer Review sind seit dem 20.04.10 online. Die Eckdaten sind: 02.05.10 Abgabe unser Artefakte und Erhalt der Artefakte der anderen Gruppe. Review - Ergebnis bis 09.05.10 versenden. Am 16.05 soll zudem der Entwurf der Abschlusspräsentation versendet werden.	alle	
3	2	A	Protokoll für Peer Review anschauen ggf. Abstimmung per Mail	alle	01.05.10
4	2	A	Artefakte nochmals sichten. Sind noch dringend Korrekturen notwendig. Abstimmung per Mail	alle	01.05.10
5	2	A	Überprüfung des Prototypen	MB	01.05.10
6	3	B	Grobe Gliederung wurde erstellt. Ausarbeitung nach Peer Review	alle	

Meeting_Protokoll_20100429.doc

26.07.2010

Seite 1 von 2

Abbildung 58: Meeting Protokoll 2

Besprechungsprotokoll

Besprechung:	Fortgeschrittene Softwaretechnologie SS 2010		
	Gruppe 2: DU05 Simulator bei der ThyssenKrupp Steel Europe AG		
Zielsetzung:			
Frequenz:	Alle 14 Tage	Datum:	04.05.2010
Initiator:	Michael Burkhardt	Uhrzeit (von .. bis):	18:00 – 21:00
Protokollant:	Stefan Albert	Ort:	Dortmund
Anwesende:	Michael Burkhardt ; Sven Münzner ; Michael Rongen (per Telefon); Thomas Baier Stefan Albert		
Verteiler:	Anwesende		
Vertraulichkeit:	Erstellt / Geändert: 05.05.2010		

Agenda

TOP	Thema
1	Durchführung Peer Review
2	Zwischenstand Präsentationsentwurf
3	Terminplanung

Ergebnisse

Lfd. Nr.	Zu TOP	Typ <small>Aktion Beschluss Information</small>	Beschreibung	Verantwortlich <small>(Name oder Initialen)</small>	Fällig <small>(Datum)</small>
1	1	I	Unsere Artefakte wurden termingerecht an die andere Gruppe versendet und auch wir haben die Artefakte erhalten.	alle	
2	1	A	Peer Review Lastenheft S. 1- 10	SA	08.05.10
3	1	A	Peer Review Lastenheft S. 11-17	MR	08.05.10
4	1	A	Peer Review Grobentwurf	SM	08.05.10
5	1	A	Peer Review Product Backlog und Sprint Backlog	TB	08.05.10
6	1	A	Peer Review Beschreibung Prototyp und Beurteilung Prototyp	MB	08.05.10
7	2	B	Keine neuen Ergebnisse. Ausarbeitung nach Peer Review	alle	
8	3	I	Neuer Termin am 12.05.2010	alle	

Abbildung 59: Meeting Protokoll 3

Besprechungsprotokoll

Besprechung:	Fortgeschrittene Softwaretechnologie SS 2010		
	Gruppe 2: DU05 Simulator bei der ThyssenKrupp Steel Europe AG		
Zielsetzung:			
Frequenz:	Alle 14 Tage	Datum:	12.05.2010
Initiator:	Michael Burkhardt	Uhrzeit (von .. bis):	18:00 – 21:00
Protokollant:	Stefan Albert	Ort:	Dortmund
Anwesende:	Michael Burkhardt ; Sven Münzner ; Michael Rongen (per Telefon); Thomas Baier Stefan Albert		
Verteiler:	Anwesende		
Vertraulichkeit:	Erstellt / Geändert: 13.05.2010		

Agenda

TOP	Thema
1	Peer Review Ergebnis
2	Zwischenstand Präsentationsentwurf
3	Terminplanung

Ergebnisse

Lfd. Nr.	Zu TOP	Typ <small>Aktion Beschluss Information</small>	Beschreibung	Verantwortlich <small>(Name oder Initialen)</small>	Fällig <small>(Datum)</small>
1	1	I	Das Ergebnis des Peer Review liegt vor. Fazit: Überwiegend Rechtschreibung und Interpunktion	alle	
2	1	A	Bereinigung der Dokumente gemäß Peer Review	alle	18.05.10
3	2	B	Vorschlag für Masterfoliensatz wurde angenommen und die Präsentation wurde aufgeteilt	alle	
4	2	A	Jeder soll für seinen Part das Feinkonzept erstellen	alle	15.05.10
5	2	A	Zusammenfassung der Ergebnisse und Weiterleitung der Zwischenpräsentation an M.Winter	SA	16.05.10
6	3	I	Neuer Termin am 27.05.2010	alle	

Abbildung 60: Meeting Protokoll 4

Besprechungsprotokoll

Besprechung:	Fortgeschrittene Softwaretechnologie SS 2010		
	Gruppe 2: DU05 Simulator bei der ThyssenKrupp Steel Europe AG		
Zielsetzung:			
Frequenz:	Alle 14 Tage	Datum:	27.05.2010
Initiator:	Michael Burkhardt	Uhrzeit (von .. bis):	18:00 – 21:00
Protokollant:	Stefan Albert	Ort:	Dortmund
Anwesende:	Michael Burkhardt ; Sven Münzner ; Michael Rongen (per Telefon); Thomas Baier Stefan Albert		
Verteiler:	Anwesende		
Vertraulichkeit:	Erstellt / Geändert: 28.05.2010		

Agenda

TOP	Thema
1	Feedback Präsentationsentwurf vom 26.05.10
2	Probepresentation
3	Terminplanung

Ergebnisse

Lfd. Nr.	Zu TOP	Typ <small>Aktion Beschluss Information</small>	Beschreibung	Verantwortlich <small>(Name oder Initialen)</small>	Fällig <small>(Datum)</small>
1	1	I	Das Feedback zum Präsentationsentwurf liegt seit gestern vor. Fazit: Was fehlt ist der „rote Faden“ (anhand von Artefakten) ggf. eine technische Reflektion. Neue Präsentationsdauer 40 Min.	alle	
2	2	I	Durchführung einer Probepresentation nach alten Vorgaben	alle	
3	2	A	Kritikpunkte in den jeweiligen Bereich beachten und Teilpresentation überarbeiten	alle	28.05.10
4	3	I	Neuer Termin am 10.06.2010	alle	

Abbildung 61: Meeting Protokoll 5

Besprechungsprotokoll

Besprechung:	Fortgeschrittene Softwaretechnologie SS 2010		
	Gruppe 2: DU05 Simulator bei der ThyssenKrupp Steel Europe AG		
Zielsetzung:			
Frequenz:	Alle 14 Tage	Datum:	10.06.2010
Initiator:	Michael Burkhardt	Uhrzeit (von .. bis):	18:00 – 21:00
Protokollant:	Stefan Albert	Ort:	Dortmund
Anwesende:	Michael Burkhardt ; Sven Münzner ; Michael Rongen (per Telefon); Thomas Baier Stefan Albert		
Verteiler:	Anwesende		
Vertraulichkeit:	Erstellt / Geändert: 11.06.2010		

Agenda

TOP	Thema
1	Reflexion Abschlusspräsentation
2	Aufbau Hausarbeit
3	Terminplanung

Ergebnisse

Lfd. Nr.	Zu TOP	Typ <small>Aktion Beschluss Information</small>	Beschreibung	Verantwortlich <small>(Name oder Initialen)</small>	Fällig <small>(Datum)</small>
1	1	I	Die Abschlusspräsentation wird von allen als gelungen empfunden. Das Feedback durch die Fragebögen (seit dem 09.06.10 online) unterstreicht dies. Negativ fällt eine Wertung mit 5,0 für den Punkt „Projekt anschaulich beschrieben“ auf. Diese ist für uns nicht nachvollziehbar.	alle	
2	2	B	Aufbau und Abgrenzung der Hausarbeit wurde festgelegt.	alle	
3	2	A	Überarbeitung des Feinentwurfs Teil 1	alle	17.06.10
4	3	I	Neuer Termin am 17.06.2010	alle	

Abbildung 62: Meeting Protokoll 6

Besprechungsprotokoll

Besprechung:	Fortgeschrittene Softwaretechnologie SS 2010		
	Gruppe 2: DU05 Simulator bei der ThyssenKrupp Steel Europe AG		
Zielsetzung:			
Frequenz:	Alle 14 Tage	Datum:	17.06.2010
Initiator:	Michael Burkhardt	Uhrzeit (von .. bis):	18:00 – 21:00
Protokollant:	Stefan Albert	Ort:	Dortmund
Anwesende:	Michael Burkhardt ; Sven Münzner ; Michael Rongen (per Telefon); Thomas Baier Stefan Albert		
Verteiler:	Anwesende		
Vertraulichkeit:	Erstellt / Geändert: 18.06.2010		

Agenda

TOP	Thema
1	Zwischenstand der Hausarbeit
2	Zwischenstand Feinentwurf
3	Zwischenstand Implementierung
4	Terminplanung

Ergebnisse

Lfd. Nr.	Zu TOP	Typ <small>Aktion Beschluss Information</small>	Beschreibung	Verantwortlich <small>(Name oder Initialen)</small>	Fällig <small>(Datum)</small>
1	1	I	Jeder hat mit seinen Teil der Hausarbeit angefangen. Die Teilergebnisse sollten per Mail allen zugänglich gemacht werden, um einen Qualitätscheck durchführen zu können.	alle	
2	2	A	Überarbeitung des Feinentwurfs Teil 2.	alle	29.06.10
3	2	A	Dokumentation der Implementierung abschließen	alle	29.06.10
4	3	I	Neuer Termin am 29.06.2010	alle	

Abbildung 63: Meeting Protokoll 7

Besprechungsprotokoll

Besprechung:	Fortgeschrittene Softwaretechnologie SS 2010		
	Gruppe 2: DU05 Simulator bei der ThyssenKrupp Steel Europe AG		
Zielsetzung:			
Frequenz:	Alle 14 Tage	Datum:	29.06.2010
Initiator:	Michael Burkhardt	Uhrzeit (von .. bis):	18:00 – 20:00
Protokollant:	Stefan Albert	Ort:	Dortmund
Anwesende:	Michael Burkhardt ; Sven Münzner ; Michael Rongen (per Telefon); Thomas Baier Stefan Albert		
Verteiler:	Anwesende		
Vertraulichkeit:	Erstellt / Geändert: 30.06.2010		

Agenda

TOP	Thema
1	Zwischenstand der Hausarbeit
2	Zwischenstand Implementierung
4	Terminplanung

Ergebnisse

Lfd. Nr.	Zu TOP	Typ <small>Aktion Beschluss Information</small>	Beschreibung	Verantwortlich <small>(Name oder Initialen)</small>	Fällig <small>(Datum)</small>
1	1	I/A	Jeder hat seinen aktuellen Zwischenstand vorgestellt. Kleinere Korrekturen sollen bis Ende Juli abgeschlossen sein.	alle	
2	2	I	Dokumentation der Implementierung ist abgeschlossen	alle	29.06.10
3	3	I	Kein neuer Termin vereinbart. Falls nötig AdHoc-Kommunikation per Mail und Telefon	alle	

Abbildung 64: Meeting Protokoll 8

Erklärung

Wir versichern an Eides statt, dass wir den Beitrag zur vorliegenden Gruppenarbeit selbständig angefertigt haben. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer entnommen sind, haben wir als entnommen kenntlich gemacht. Das gleiche gilt für die von den auf dem Titelblatt der Arbeit genannten Autoren gemeinsam verfassten Teile. Sämtliche Quellen und Hilfsmittel, die wir für die Arbeit benutzt haben, sind angegeben. Die Arbeit hat mit gleichem Inhalt bzw. in wesentlichen Teilen noch keiner anderen Prüfungsbehörde vorgelegen.

Dortmund, den 29. Juli 2010

.....

(Michael Burkhardt)

(Michael Rongen)

(Sven Münzner)

(Thomas Baier)

(Stefan Albert)

Teil X.

Abkürzungsverzeichnis

ASCII American Standard Code for Information
Interchange, 38, 68

FDD Floppy Disc Drive, 37, 38, 68

GUI Graphical User Interface, 5

HDD Hard Disc Drive, 37, 38, 68

MHZ Mega Herz, 93

MVC Modell View Controller, 97

SPS Speicher programmierbare Steuerung, 26

TBM Thyssen Blas-Metallurgie, 59

TKSE ThyssenKrupp Steel Europe, 7

USB Universal Serial Bus, 37, 38, 68

VGA Video Graphics Array, 93

