

Ad-Hoc Chatsystem für mobile Netze Barracuda

Softwareentwicklungspraktikum
Sommersemester 2007

Pflichtenheft



Auftraggeber

Technische Universität Braunschweig
Institut für Betriebssysteme und Rechnerverbund
Prof. Dr.-Ing. Lars Wolf
Mühlenpfordtstraße 23
38106 Braunschweig

Betreuer

Sven Lahde, Oliver Wellnitz

Auftragnehmer

Gruppe 2

Name	E-Mail
Stephan Friedrichs	stephan.friedrichs@tu-bs.de
Henning Günther	h.guenther@tu-bs.de
Oliver Mielentz	o.mielentz@tu-bs.de
Martin Wegner	mw@mroot.net

Braunschweig, 15. Juni 2007

Versionsübersicht

Phasenverantwortlicher: Oliver Mielentz

Version	Datum	Autor	Status	Kommentar
5	15.06.2007	Stephan Friedrichs		Produktfunktionen und -daten leicht korrigiert; Benutzeroberfläche und technische Produktumgebung angepasst; Protokollspezifikation Nr. 09 eingebunden
4	14.05.2007	Martin Wegner		Produktdaten verbessert, Hinweis auf NTP eingefügt
3	07.05.2007	Henning Günther		Geschäftsprozess hinzugefügt
2	23.04.2007	Stephan Friedrichs, Martin Wegner		Kleine Korrekturen
1	16.04.2007	Gruppe 2 (s. Auftragnehmer)		Erste Version

Inhaltsverzeichnis

1 Zielbestimmung	3
1.1 Musskriterien	3
1.2 Wunschkriterien	3
1.3 Abgrenzungskriterien	4
2 Produkteinsatz	4
2.1 Anwendungsbereiche	4
2.2 Zielgruppen	4
2.3 Betriebsbedingungen	4
3 Produktübersicht	4
4 Produktfunktionen	8
5 Produktdaten	10
5.1 Allgemeine Bemerkungen	11
6 Produktleistungen	11
7 Qualitätsanforderungen	11
7.1 Übersicht	11
7.2 Erläuterungen	11
7.3 Weitere Anforderungen	12
8 Benutzeroberfläche	13
8.1 Entwurf	13
9 Nichtfunktionale Anforderungen	14
10 Technische Produktumgebung	14
10.1 Software	14
10.2 Hardware	14
10.3 Orgware	14
10.4 Produktschnittstellen	14

11 Glossar	15
12 Referenzen	15

1 Zielbestimmung

Es soll ein Chatsystem für mobile und drahtlose Netze erstellt werden, das ohne Infrastruktur wie zum Beispiel einem zentralen Server auskommt. Dabei werden die Nachrichten ähnlich wie in einem Peer-to-Peer Netz direkt vom Absender an die Empfänger geschickt. Ist ein Empfänger nicht direkt erreichbar, so kann gegebenenfalls ein anderer Netzteilnehmer die Nachricht stellvertretend entgegennehmen und seinerseits an den Empfänger weiterleiten. Ein zentraler Punkt dabei ist, dass sich die Topologie des Ad-Hoc Netzes im laufenden Betrieb ständig ändern kann. So muss das System beispielsweise angemessen reagieren können, wenn Peers nicht mehr erreichbar sind oder neu hinzukommen; Netze können in kleinere Teilnetze zerfallen, oder mehrere Teilnetze zu einem werden.

1.1 Musskriterien

Im Folgenden werden die zentralen Features des Endprodukts beschrieben.

- Implementierung des gegebenen Protokolls
 - Aufbau des Ad-Hoc Netzwerkes
 - Erfassung benachbarter Peers
 - Erfassung und geeignete Behandlung von Änderungen in der Netzwerktopologie
 - Kryptologie
 - * X.509 Zertifikate (Authentifizierung von Nutzern/Nachrichten)
 - * Checksums (Überprüfung der Integrität von Nachrichten)
 - * Verschlüsselung
 - Realisierung verschiedener Channeltypen
 - * anonym
 - * privat
 - * öffentlich
 - Mitgliedschaft in Channeln
 - * Permanente Mitgliedschaft im einzigen anonymen Channel
 - * Teilnahme an beliebig vielen weiteren privaten und öffentlichen Channeln
 - * Senden/Empfangen von Nachrichten in diesen Channeln (diese können Texte oder Dateien begrenzter Größe sein)
 - Manuelles Überschreiben von Routinginformationen zum Infrastrukturbetrieb
- Interoperabilität mit standardkonformen Implementierungen des gegebenen Protokolls
- Plattformunabhängigkeit (mindestens unter Linux und MacOS X)
- Grafische Oberfläche

1.2 Wunschkriterien

Wenn die Zeit reicht, wird Wert auf die hier genannten Verbesserungen und Features gelegt.

- Optimiertes Routing zum Entlasten anderer Peers Skalierbarkeit im Bezug auf
 - Last
 - Durchsatz
- Durchdachte grafische Oberfläche
 - Erweiterte Benachrichtigungen über Änderungen der Netzwerktopologie, vor allem Merge- und Splitvorgänge

- Einfaches Interface zur Einsicht und Überschreibung der Routinginformationen (nur im Infrastrukturmodus editierbar)
- Schutz vor und Erkennung von Angriffen durch Peers
 - Flooding
 - DoS
 - Angriff auf chiffrierte Informationen

1.3 Abgrenzungskriterien

Die folgenden Merkmale werden definitiv nicht implementiert werden, sondern dienen dazu, die Grenzen des Projektes aufzuzeigen.

- Es handelt sich um ein Chatsystem. Der Austausch von Dateien ist eine sekundäre Funktion, es werden keine Mechanismen zur Vereinfachung von Filesharing implementiert.
- Kein Voice- oder Videochat.
- Keine Nutzerprofile mit Nicknames etc., die Identifikation von Nutzern findet über Zertifikate statt.
- Kein Instant-Messenger wie z.B. Jabber. Für die Kommunikation zwischen genau zwei Personen muss ein Channel geöffnet werden.

2 Produkteinsatz

In diesem Abschnitt werden Anwendungsgebiet, Zielgruppen und Betriebsbedingungen des Systems beschrieben.

2.1 Anwendungsbereiche

Protokoll, Planung, Implementierung und Optimierung des Produkts sind komplett darauf ausgelegt, dass es in Ad-Hoc Netzwerken verwendet wird.

Das heißt einerseits, dass das Produkt in Netzen, deren Topologie sich laufend ändert, gut funktionieren wird, andererseits wird es in Netzen mit statischer Topologie (z.B. Wired LAN) längst nicht eine so gute Performance liefern können, wie Chatsysteme, die auf eine feste Client-Server-Struktur ausgelegt sind.

2.2 Zielgruppen

Es gibt keine spezielle Zielgruppe. Jeder der sich berufen fühlt, in spontan gebildeten Netzwerken zu chatten, kann das Produkt nutzen.

2.3 Betriebsbedingungen

Das Produkt stellt wenig Anforderungen an die Hardware. Alles, was einen Netzwerkanschluss besitzt und in der Lage ist, die grafische Oberfläche darzustellen, kann das Produkt quasi im Dauerbetrieb verwenden. Dazu ist weder Beaufsichtigung noch Wartung nötig, noch werden extensiv Ressourcen beansprucht.

3 Produktübersicht

Die folgenden Use-Case Diagramme umreißen die Funktionen des Produktes. Da alle Nodes gleichberechtigt sind, gibt es in allen Diagrammen nur eine Art von Usern.

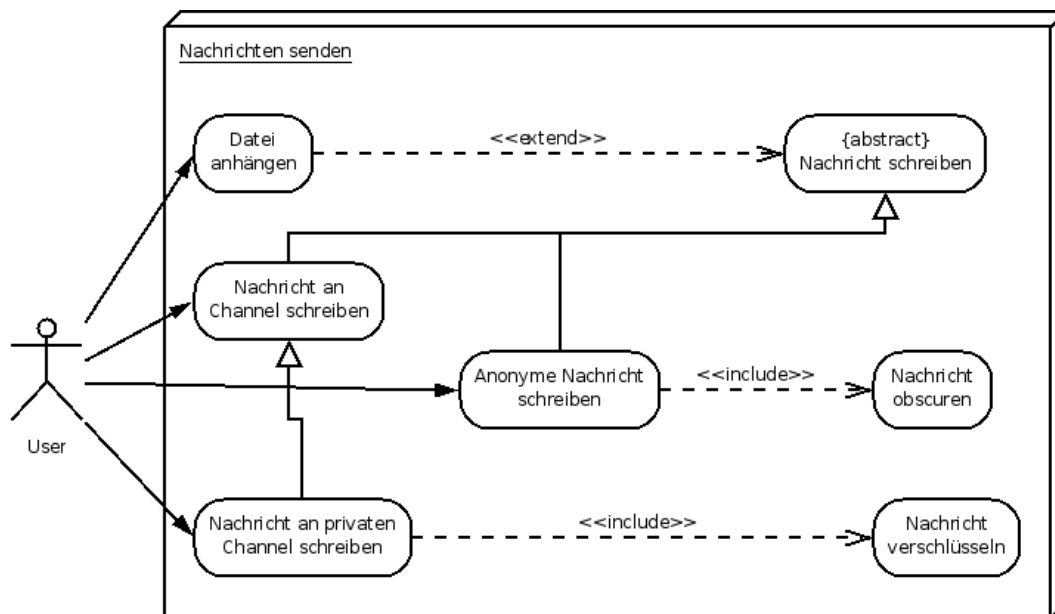


Abbildung 1: Use Case 1: Senden

Es können Nachrichten an die verschiedenen Channeltypen (anonym, öffentlich und privat) geschrieben werden. Die Herkunft anonymer Nachrichten werden per OBSCURE verschleiert, Nachrichten an private Kanäle werden mit dem Schlüssel des Kanals verschlüsselt. Es können Dateien begrenzter Größe an alle Nachrichten gehängt werden.

Anonyme Nachrichten können von allen gelesen werden, Nachrichten an öffentliche/private Kanäle nur von den Mitgliedern der jeweiligen Kanäle. Es besteht kein Abhörschutz für öffentliche Kanäle.

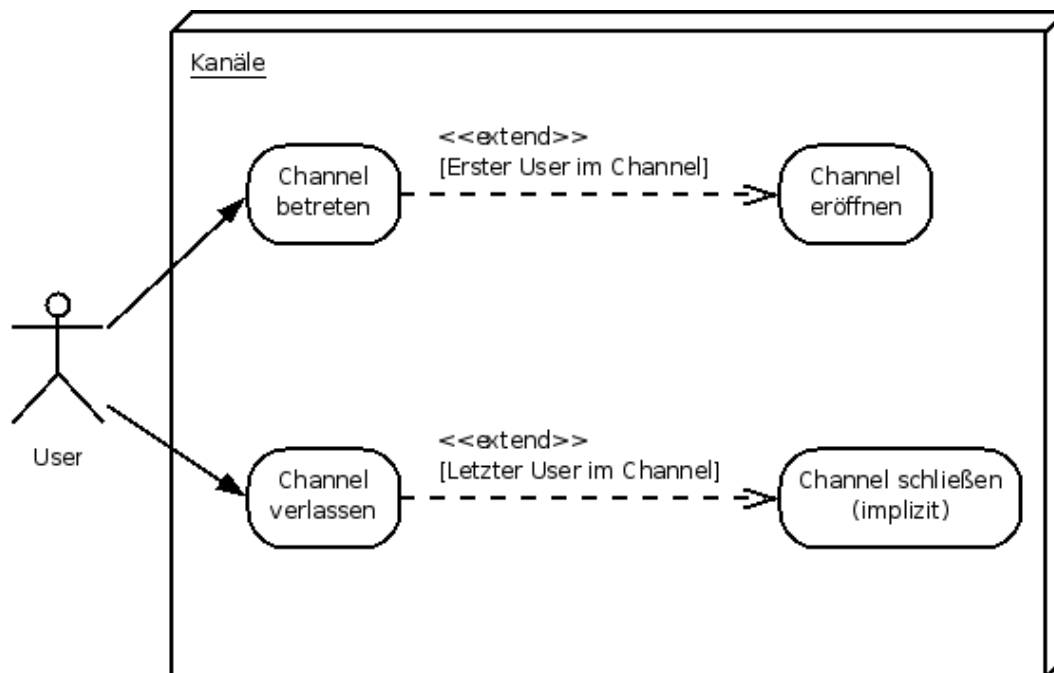


Abbildung 2: Use Case 2: Kanäle

Außerdem ist es möglich, ggf. verschlüsselten Kanälen beizutreten. Kanäle werden eröffnet, falls man der erste ist, der sie betritt, bzw. implizit dadurch geschlossen, dass man der letzte ist, der sie verlässt (die Kanäle hören dann auf zu existieren, werden aber nicht explizit geschlossen).

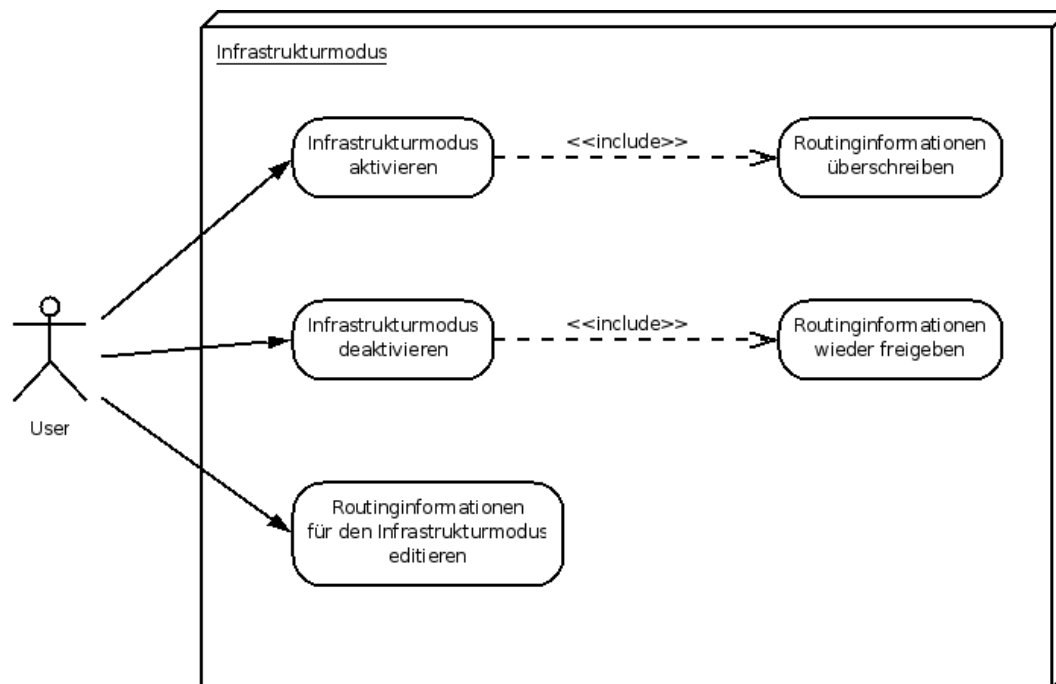


Abbildung 3: Use Case 3: Infrastrukturmodus

Der Infrastrukturmodus überschreibt die durch das Netz gewonnenen Routinginformationen durch manuell eingegebene und dient nicht zum Korrigieren fehlerhaft erkannter Netztopologie, sondern zum Testbetrieb.

4 Produktfunktionen

/F100/ Rendezvous

Geschäftsprozess: Rendezvous

Ziel: Kommen zwei Nodes in Übertragungreichweite, erfahren sie von der Präsenz des jeweils anderen

Kategorie: primär

Vorbedingung: Die Nodes müssen in Übertragungreichweite sein, so dass UDP-Pakete verschickt und empfangen werden können.

Nachbedingung Erfolg: Die Nodes kennen IP und Routing-Informationen des anderen Node

Nachbedingung Fehlschlag: Ein oder beide Nodes erkennen die Präsenz des anderen Node nicht

Akteure: Zwei Nodes

Auslösendes Ereignis: Durch eine in regelmäßigen Abständen verschickte HELLO-Nachricht erfährt ein Node von der Präsenz des anderen

Beschreibung:

1. Durch die HELLO Nachricht erfährt der Node IP-Adresse, sowie die User-ID des anderen Node
2. Durch eine darauf folgende ROUTING-Nachricht erfährt der Node, welche anderen Nodes über den Nachbarn zu erreichen sind
3. Die Informationen werden für zukünftige Verwendung gespeichert

/F200/ Nachricht an öffentlichen Kanal

Geschäftsprozess: Nachricht an öffentlichen Kanal

Ziel: Eine Nachricht wird in einem öffentlichen Kanal veröffentlicht und an alle Teilnehmer desselben weitergeleitet

Kategorie: primär

Vorbedingung: Ein Benutzer hat einen Kanal betreten und eine Nachricht in diesem veröffentlicht

Nachbedingung Erfolg: Die Nachricht wird an alle Mitglieder des Kanals zugestellt, die durch das Netzwerk erreichbar sind

Nachbedingung Fehlschlag: Die Nachricht wird an mindestens ein Mitglied, das durch das Netzwerk erreichbar ist, nicht zugestellt

Akteure: Ein Absender, mehrere Kanalmitglieder

Auslösendes Ereignis: Ein Node sendet, veranlasst durch den Benutzer eine MESSAGE-Nachricht an alle Mitglieder eines Kanals

Beschreibung:

1. Der Node des Senders ermittelt aufgrund von lokal gespeicherten Informationen alle Mitglieder des entsprechenden Kanals
2. Mithilfe der gespeicherten Routing-Informationen ermittelt der Node, wie die Nachricht im Netzwerk verschickt werden muss, um alle Mitglieder des Kanals zu erreichen
3. Die Nachricht wird wie oben ermittelt verschickt

Erweiterung:

1. Zu 2: Es wird eine Verteilung der Nachricht gesucht, die möglichst wenig Ressourcen des Netzwerks verwendet, also möglichst effizient zu allen Zielen gelangt

Alternativen:

1. Zu 3: Kommt es beim Versand zu Fehlern, wird gewartet, bis die Nachricht erneut verschickt werden kann

/F300/ Anonyme Kommunikation

Geschäftsprozess: Anonyme Kommunikation

Ziel: Eine Nachricht wird an den anonymen Kanal gesendet

Kategorie: primär

Vorbedingung: Ein Benutzer will eine anonyme Nachricht verschicken

Nachbedingung Erfolg: Die Nachricht wird so im anonymen Kanal veröffentlicht, dass es nicht möglich ist, den Urheber ausfindig zu machen

Nachbedingung Fehlschlag: Die Nachricht wird nicht veröffentlicht, oder der Urheber ist mit vertretbarem Aufwand ausfindig zu machen

Auslösendes Ereignis: Ein Benutzer veranlasst seinen Node, eine anonyme Nachricht zu verschicken

Beschreibung:

1. Der Node wählt 3-5 andere Nodes aus
2. Der Node verwendet die öffentlichen Schlüssel (erhalten über GETCERTIFICATE-Nachrichten) der Nodes, um die Nachricht hintereinander für jeden Node zu verschlüsseln und in eine neue OBSCURE-Nachricht zu verpacken
3. Der Node versendet die Nachricht an den Node, dessen Schlüssel zuletzt zum Verschlüsseln verwendet wurde
4. Jeder Node auf dem Pfad entschlüsselt die OBSCURE-Nachricht und leitet den Inhalt weiter, der letzte sendet die enthaltene MESSAGE-Nachricht an den anonymen Kanal

Alternativen:

1. Zu 1: Sind nicht genügend Nodes im Netzwerk verfügbar, kann Geheimhaltung nicht garantiert werden. Die Nachricht ist zu verwerfen.
2. Zu 4: Wird die Nachricht nach Ablauf des „Time To Live“-Wertes nicht im anonymen Kanal veröffentlicht, so muss die Nachricht erneut verschlüsselt und versendet werden.

/F400/ Partitionierung und Neusynchronisation

Geschäftsprozess: Partitionierung und Neusynchronisation

Ziel: Ein Ad-Hoc Netzwerk kann durch Entfernung von Nodes oder Verbindungen in zwei oder mehr Bereiche getrennt werden, die nicht mehr kommunizieren können. Bekommen die Teil-Netze wieder Kontakt, werden gleichnamige Kanäle zusammengeführt.

Kategorie: primär

Vorbedingung: Es existieren zwei nicht verbundene Netzwerkeile

Nachbedingung Erfolg: Alle öffentlichen Kanäle, die gleichnamig sind, werden zusammen geführt

Nachbedingung Fehlschlag: Mindestens ein öffentlicher Kanal wird nicht zusammen geführt

Auslösendes Ereignis: Die Netzwerkteile werden verbunden

Beschreibung:

1. Durch das Zusammenführen der Netzwerkteile, empfangen Nodes in beiden Teilen CHANNEL-Nachrichten, die unterschiedliche Channel-IDs haben
2. Jede Node führt die Nutzerlisten dieser Kanäle zusammen und wählt die kleinere Channel-ID als neue aus
3. Tut dies jede Node, verschwindet die höhere Channel-ID aus dem Netzwerk und die Kanäle sind zusammengeführt

/F500/ Infrastrukturmodus

Geschäftsprozess: Verwaltung des Infrastrukturmodus

Ziel: Der Infrastrukturmodus erlaubt das Testen des Chat-Clients in statischen Netzen durch Einschränkung der möglichen Kommunikationspartner

Kategorie: primär

Vorbedingung: keine

Nachbedingung Erfolg: Der Benutzer ist in der Lage, den Infrastrukturmodus zu aktivieren und Peers, mit denen kommuniziert werden darf, hinzuzufügen bzw. zu löschen

Nachbedingung Fehlschlag: Der Benutzer ist nicht in der Lage, einen dieser Aufgabenbereiche zu erfüllen

Auslösendes Ereignis: keins

Beschreibung:

1. Der Benutzer startet die Infrastrukturverwaltung
2. Der Benutzer aktiviert oder deaktiviert den Infrastrukturmodus der Node
3. Ist der Infrastrukturmodus aktiviert, können Peers auf die Liste gesetzt oder von ihr entfernt werden

5 Produktdaten

/D10/ Konfigurationsdaten

In einer Konfigurationsdatei können folgende Daten angepasst werden:

- Die eigene UserID
- Pfad zu eigenem Zertifikat
- Pfad zu eigenem Secret Key
- Evtl. durch den Benutzer vorgenommene Einstellungen zum Aussehen der GUI

/D20/ IBR-Zertifikat

Hier wird der Public Key des IBR gespeichert, um die Gültigkeit der Benutzerzertifikate überprüfen zu können.

/LD20/ Zertifikate

Hier werden zu einer Liste bestimmter UserID's die zugehörigen, bekannten Zertifikate verwaltet.

5.1 Allgemeine Bemerkungen

Da die meisten Daten über das Netzwerk, die Topologie, die Teilnehmer, etc. den empfangenen Nachrichten entnommen werden und sich aufgrund der zu erwartenden starken Fluktuationen dieser Daten kein zumindest mittelfristig bleibender Zustand einpendeln wird, werden über obige Daten hinausgehend keine weiteren gespeichert.

6 Produktleistungen

/L10/ Weitgehende Erhaltung des Chat-Kontexts

Nachrichten sollen unabhängig von ihrer Zustellungsdauer weitestgehend – sofern dies die Informationen, die dem Protokoll entnommen werden können, zulassen – in den Gesprächskontext eingepasst werden.

/L20/ Zeitnahe Zustellung von Nachrichten

Die Wahl der verschiedenen Timeouts und das Routing sollen möglichst gut einem zeitnahen Chat-Erlebnis zuarbeiten.

7 Qualitätsanforderungen

7.1 Übersicht

Produktqualität	sehr gut	gut	normal	nicht relevant
<i>Funktionalität</i>				
Richtigkeit	X			
Interopabilität	X			
<i>Sicherheit</i>				
Zuverlässigkeit			X	
Reife		X		
Fehlertoleranz		X		
Wiederherstellbarkeit		X		
<i>Benutzbarkeit</i>				
Verständlichkeit			X	
Erlernbarkeit			X	
Bedienbarkeit		X		
Effizienz			X	
Zeitverhalten		X		
Verbrauchsverhalten		X		
<i>Änderbarkeit</i>				
Analysierbarkeit		X		
Modifizierbarkeit			X	
Stabilität		X		
Prüfbarkeit		X		
<i>Übertragbarkeit</i>				
Anpassbarkeit			X	
Installierbarkeit		X		
Konformität	X			
Austauschbarkeit	X			

7.2 Erläuterungen

Zuverlässigkeit und Reife: Die Software wendet zuverlässig durch das Protokoll vorgesehene Sicherheitskonzepte (Signaturen, Verschlüsselung) an.

Fehlertoleranz: Die Software soll Fehler – soweit im Umfang des Protokolls möglich – korrigieren bzw. tolerieren.

Wiederherstellbarkeit: Nach dem unerwarteten oder absichtlichen Beenden des Programms kann der vorherige Zustand in vertretbarer Zeit wiederhergestellt werden. Dies ist auch darin begründet, dass bei erneuter Teilnahme am Ad-Hoc-Netz die Informationen über das Netz leicht wieder gewonnen werden können.

Verständlichkeit und Erlernbarkeit: Die Verständlichkeit und die Erlernbarkeit der Benutzeroberfläche sind normal, d. h. die Benutzeroberfläche ist mit ähnlichem Aufwand zu erlernen und zu benutzen wie bei vergleichbarer Software.

Effizienz: Es wird kein besonderes Augenmerk auf die Effizienz, mit der die Benutzeroberfläche bedient werden kann, gelegt werden.

Zeitverhalten und Verbrauchsverhalten: Durch sorgfältige Programmierung soll ein gutes Zeit- und Verbrauchsverhalten der Software im Betrieb erreicht werden.

Analysierbarkeit und Prüfbarkeit: Durch gute Dokumentation und die Wahl einer geeigneten Programmiersprache, die dies gut unterstützt, ist das Programm gut analysierbar und prüfbar.

Modifizierbarkeit: Die Software wird mit dem üblichen Aufwand änder-, weiterentwickel- und erweiterbar sein.

Stabilität: Zum einen sollen, wenn Module der Software hinzugefügt werden, die bisherigen Module stabil weiter funktionieren. Zum anderen sollen alle Schnittstellen stabil gehalten werden.

Anpassbarkeit: Die Software wird mit normalem Aufwand anpassbar sein, d. h. es wird z. B. keine Plug-In-Struktur für bestimmte Komponenten geben.

Installierbarkeit: Die fertige Software wird gut in den unterstützten Zielbetriebssystemen zu installieren sein.

Konformität: Die Software wird sich an die durch das Protokoll gegebenen Standards halten.

Austauschbarkeit: Die Software kann auf den einzelnen Nodes durch andere protokollkonforme Implementierungen ausgetauscht werden.

7.3 Weitere Anforderungen

Außerdem werden folgende weiteren Qualitätsanforderungen explizit gestellt:

- Die gesendeten Daten sollen gegen das vorgegebene XML Schema validieren.
- Optimale Ausnutzung der durch das Protokoll ermöglichten Erkennung von unzustellbaren Nachrichten.

8 Benutzeroberfläche

8.1 Entwurf

Die folgende Grafik stellt einen ersten Entwurf der GUI, wie sie einmal aussehen könnte, dar.

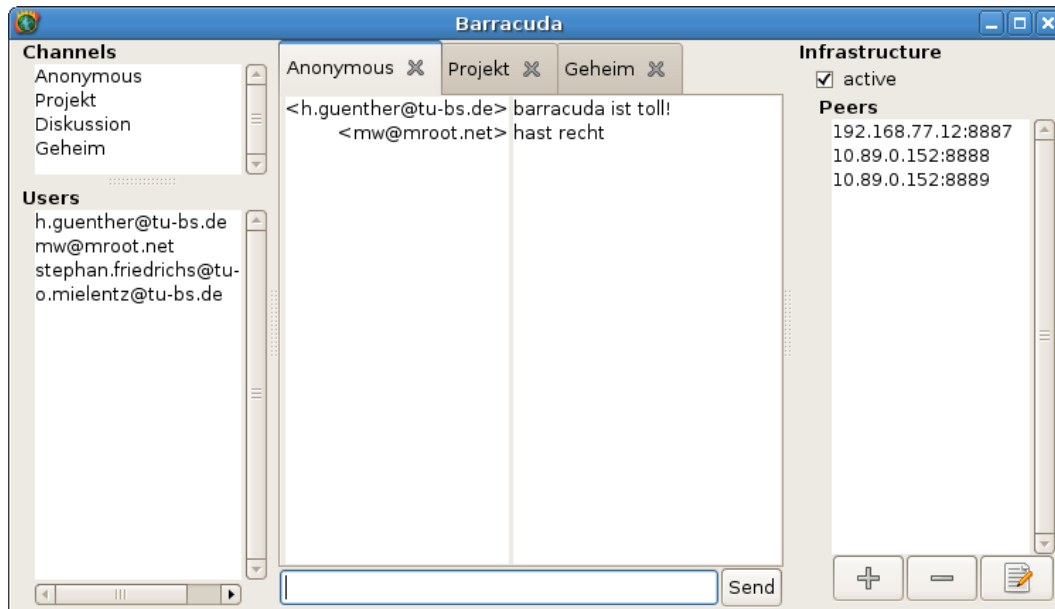


Abbildung 4: Erster Entwurf der GUI

Bei allen folgenden Ausführungen ist zu beachten, dass während der Implementierungs- und Testphase einzelne GUI-Komponenten durch andere ersetzt werden können, um die Benutzerfreundlichkeit zu optimieren. Dabei wird die Funktionalität jedoch nicht eingeschränkt werden, zum Beispiel könnten Registerreiter durch eine Liste ersetzt werden.

/B10/

Standardmäßig sind die Gnome Human Interface Guidelines in der Benutzeroberfläche zu beachten.

/B20/

Die GUI beinhaltet ein Chatfenster in welchem die Channels als Registerreiter ihren Platz finden. Jeder Reiter zeigt die Messages sowie die Teilnehmer des entsprechenden Channels. Desweiteren können Listen mit allen bekannten Teilnehmern sowie allen bekannten Channels des Ad-Hoc Netzwerkes angezeigt werden.

Zur Steuerung des Infrastrukturmodus gibt es ein Panel, in dem der Modus aktiviert, sowie eine Liste benachbarter Peers editiert werden kann. Dieser wird ggf. noch in ein externes Fenster ausgelagert.

/B30/

Die GUI unterstützt weitestgehend die Steuerung mit einer Maus bzw. einem mausähnlichem Eingabegerät.

9 Nichtfunktionale Anforderungen

/Inf10/

Die Software soll es ermöglichen, dass User, welche keine fundierten Informatik- bzw. Computerkenntnisse besitzen, problemlos in einem Ad-Hoc Netzwerk miteinander chatten können, ohne sich oft bewusst um die Rahmenbedingungen (wie z. B. User gerade nicht erreichbar, Netzwerkabbruch) kümmern zu müssen.

/Inf20/

Die Software soll mit geringem Aufwand weiterentwickelbar und wartbar sein.

/Inf30/

Die Software soll fehlertolerant bezüglich der Benutzereingabe sein.

10 Technische Produktumgebung

10.1 Software

Prinzipiell kann die Software auf jedem Betriebssystem kompiliert werden, unter dem ein Haskell-Compiler zur Verfügung steht und ein grafisches Frontend vorhanden ist. Primär entwickelt wird sie jedoch für Linux und andere Unix-Derivate, darunter auch MacOS X. Binaries sind nur für MacOS X und GNU Linux (x86/x86_64) geplant. Für die GUI müssen die GTK+ Librarys installiert sein.

Desweiteren wird der Einsatz einer Software zur Zeitsynchronisation empfohlen, wie z.B. NTP. Dadurch kann ein angenehmerer Chat-Ablauf möglich werden, denn das Protokoll sieht eine zeitliche Sortierung der Nachrichten vor.

10.2 Hardware

Als Hardwarevoraussetzungen benötigt man nur eine Möglichkeit an einem Ad-Hoc Netzwerk teilnehmen zu können. Die Teilnahme an einem „simulierten“ Ad-Hoc Netzwerk in einer festen Netzwerkumgebung ist über den Infrastrukturmodus der Software möglich.

Es werden prinzipiell alle Architekturen unterstützt, unter denen GHC und GTK+ vorhanden sind, Binaries werden jedoch nur für ausgewählte Plattformen verteilt, siehe oben.

10.3 Orgware

Die Software setzt ein vorhandenes Ad-Hoc Netzwerk, bzw. ein bestehendes LAN (für den Infrastrukturmodus) voraus.

10.4 Produktschnittstellen

Die Kommunikation mit anderen Ad-Hoc Chat-Systemen erfolgt über das vorgegebene, spezifizierte *draft-strauss-p2p-chat-09* Protokoll. Die Verbindung zu den anderen Systemen wird über das UDP-Protokoll hergestellt.

11 Glossar

(Net-)Merge Zusammenführen vorher getrennter Ad-hoc Netze

(Net-)Split Aufteilen eines Ad-hoc Netzes

Node/Knoten Ein Teilnehmer im Netzwerk. Mehrere Nodes können auf dem selben Rechner aktiv sein.

Peer Die Node am anderen Ende einer Verbindung.

GTK+ Gimp-ToolKit, eine freie Komponentenbibliothek unter LGPL, mit der es möglich ist, grafische User Interfaces (GUI) für Software zu erstellen.

GHC Glasgow Haskell Compiler, ein Compiler für die funktionale Programmiersprache Haskell.

UDP Das User Datagram Protocol (Abk. UDP) ist ein minimales, verbindungsloses Netzprotokoll, das zur Transportschicht der Internetprotokollfamilie gehört.

GUI Das *Graphical User Interface* ist die grafische Bedienoberfläche eines Programms.

12 Referenzen

1. Lastenheft „Ad-hoc Chatsystem für mobile Netze“ von Oliver Wellnitz, <http://www.ibr.cs.tu-bs.de/courses/ss07/sep-cm/lastenheft.html>
2. Protokoll-Spezifikation: „draft-strauss-p3p-chat-09 – P2P CHAT - A Peer-to-Peer Chat Protocol“ von F. Strauss (u. a.), <http://www.ibr.cs.tu-bs.de/courses/ss07/sep-cm/protocol/draft-strauss-p2p-chat-09.txt>
3. „GNOME Human Interface Guidelines 2.0“ von Calum Benson, Adam Elman, Seth Nickell, Colin Z Robertson, <http://developer.gnome.org/projects/gup/hig/2.0/>
4. „NTP – Network Time Protocol“, <http://www.ntp.org/>

Abbildungsverzeichnis

1	Use Case 1: Senden	5
2	Use Case 2: Kanäle	6
3	Use Case 3: Infrastrukturmodus	7
4	Erster Entwurf der GUI	13