A d - h o c C h a t s y s t e m f ü r m o b i l e N e t z e

G r u p p e 3 (A d B e e)

Softwareentwicklungspraktikum
Sommersemester 2007

G r o b e n t w u r f

Auftraggeber:
Technische Universität Braunschweig
Institut für Betriebssysteme und Rechnerverbund
Prof. Dr.-Ing. Lars Wolf
Mühlenpfordtstraße 23, 1. OG
38106 Braunschweig
Deutschland

Betreuer: Sven Lahde, Oliver Wellnitz
Hiwi: Wolf-Bastian Pöttner

Auftragnehmer: Gruppe 3(AdBee)

Name	E - Mail
Ekrem Özmen	e.oezmen@gmail.com
Celal Özyalcin	c.oezyalcin@tu-bs.de
Thorben Schulze	thorben.schulze@tu-bs.de
Danny Melching	danny.melching@tu-bs.de

Phasenverantwortlicher: Ekrem Özmen

Braunschweig, 30.04.2007

Versionsübersicht

Version	Datum	Autor	Status	Kommentar
1.0	30.04	Ekrem, Celal, Danny, Thorben		Erste Version des Grobentwurfs
2.0	07.07	Ekrem, Celal, Danny, Thorben		**1** Überarbeitung in der Beschreibung der Funktionen **2** Ausbesserung der Diagramme **3** Hinzufügen von zwei neuen Funktionen aus dem überarbeiteten Pflichtenheft /F95/ /F175/ **4** Deutlichere Beschreibung der Schnittstellen **5** Entfernen der unpassenden Diagramme aus Kapitel 3

I n h a l t s v e r z e i c h n i s

A b b i l d u n g s v e r z e i c h n i s

1 Einleitung

Die Aufgabe ist, ein Ad-hoc Chatprogramm zu erstellen, wobei jeder Client ein Knoten im Netz darstellt. Der Nachrichtenaustausch soll innerhalb von Kanälen nach einen vorgegebenen Protokoll stattfinden. Folglich kann das Chatprogramm Kanäle verwalten und Nachrichten senden und auf selbige entsprechend reagieren.

1.1 Projektdetails

Kanalverwaltung:

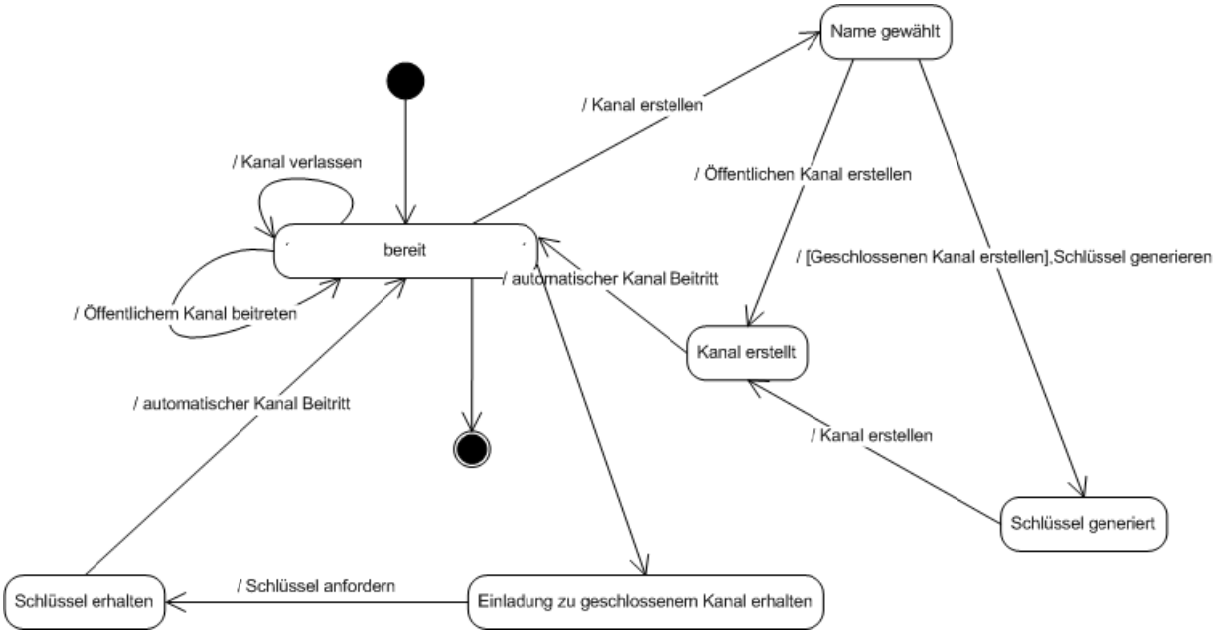

Abbildung 1: StateChart zu Kanalverwaltung

Das StateChart beschreibt die Interaktionen zwischen Benutzer und Programm, beim Erstellen, Beitreten und Verlassen von Kanälen. Startet der Benutzer das Programm, so ist er automatisch Mitglied des anonymen Kanals, daher ist dieser hier nicht aufgeführt. Befindet man sich in dem Zustand „bereit“ kann man unbegrenzt Kanäle beitreten und erstellen. Daher sind alle Aktionen Schleifen, die wieder in diesen Zustand zurückführen. Möchte der Benutzer einen Kanal erstellen, muss zunächst ein Name gewählt werden. Dieser darf noch nicht im Netz vorhanden sein. Als nächstes muss entschieden werden, ob es ein öffentlicher oder ein geschlossener Kanal sein soll. Bei letzterem muss zunächst noch ein Schlüssel für die Verschlüsselung der Nachrichten generiert werden, dann wird der Kanal erstellt und der Ersteller tritt automatisch bei. Erhält man eine Einladung in einen geschlossenen Kanal von einem Teilnehmer eines solchen, wird vom eigenen Programm der Schlüssel des Kanals angefordert und der Benutzer tritt automatisch bei. Kanal verlassen und öffentlichen Kanal beitreten, sind einzelne Anweisungen und bedürfen keiner weiteren Aktionen.

Nachrichtenverwaltung:

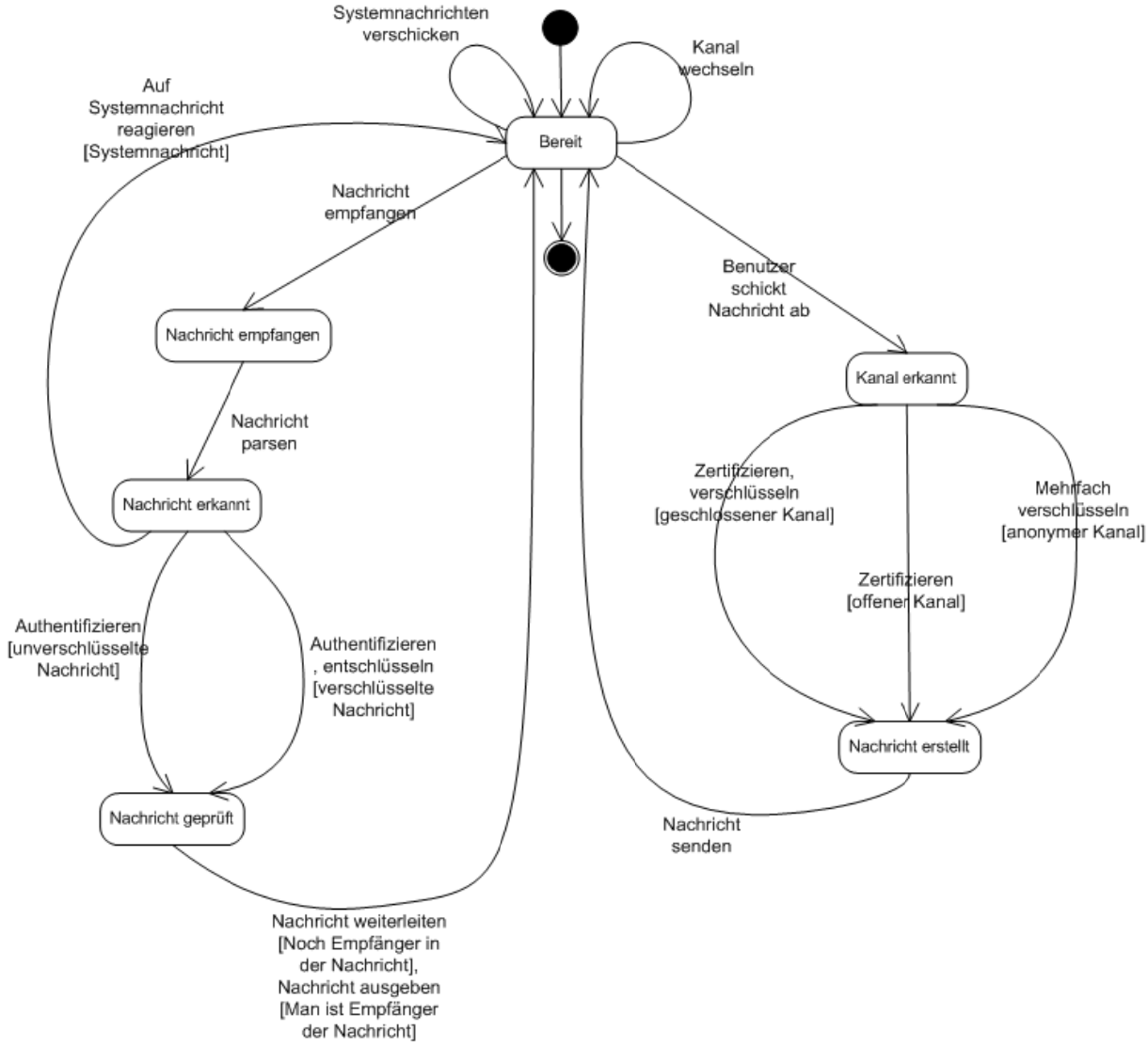

Abbildung 2: StateChart zu Nachrichtenverwaltung

Das StateChart beschreibt grob die Nachrichten-Funktionen, die das Programm nach dem Start, im Bereit-Zustand, ausführen kann. Zunächst kann der Benutzer den Kanal wechseln und je nachdem, welcher Kanal gerade aktiv ist, wird das Abschicken der Nachrichten in diesem Kanal durchgeführt. Nach dem Abschicken einer Nachricht erkennt das Programm, welcher Kanal gerade aktiv ist, erstellt die Nachricht dementsprechend und schickt sie dann an die richtigen Empfänger.

Des Weiteren kann das Programm Nachrichten empfangen. Sobald dies passiert ist, wird die Nachricht eingelesen und das Programm handelt dementsprechend. Eine angekommene Nachricht kann z.B. entschlüsselt und weitergeleitet werden. Außerdem kann es auf Systemnachrichten reagieren, selbige verschicken und so das korrekte Verhalten des Programms gewährleisten.

2 Analyse der Produktfunktionen

Im Folgenden werden die Produktfunktionen aus dem Pflichtenheft analysiert. Dabei sind die Aktivitätsdiagramme in der Grobanalyse zum Beschreiben des Ablaufs der Funktion da. Außerdem zeigen sie teilweise die Verteilung in der Architektur des Programms.
In der Feinanalyse werden Sequenzdiagramme verwendet, um die Interaktion von Objekten darzustellen. Die Objekte sind aber noch nicht die späteren Klassen. Diese werden erst im Feinentwurf entwickelt, weil sonst die Komplexität der Diagramme zu groß gewesen wäre. Die Objekte geben aber einen guten Einblick in die Interaktionen, da sie so gewählt sind, dass sie später nur verfeinert und nicht mehr verteilt werden müssen. Bei den Sequenzdiagrammen ist noch zu beachten, dass der Benutzer kein Objekt ist, sondern nur zur Veranschaulichung der Interaktion zwischen dem Programm und dem menschlichen Benutzer da ist.

2.1 Analyse von Funktionalität /F10/ : Nachricht im geschlossenen Kanal senden

Die Funktion führt das Senden einer Nachricht im geschlossenen Kanal durch.

2.1.1 Grobanalyse

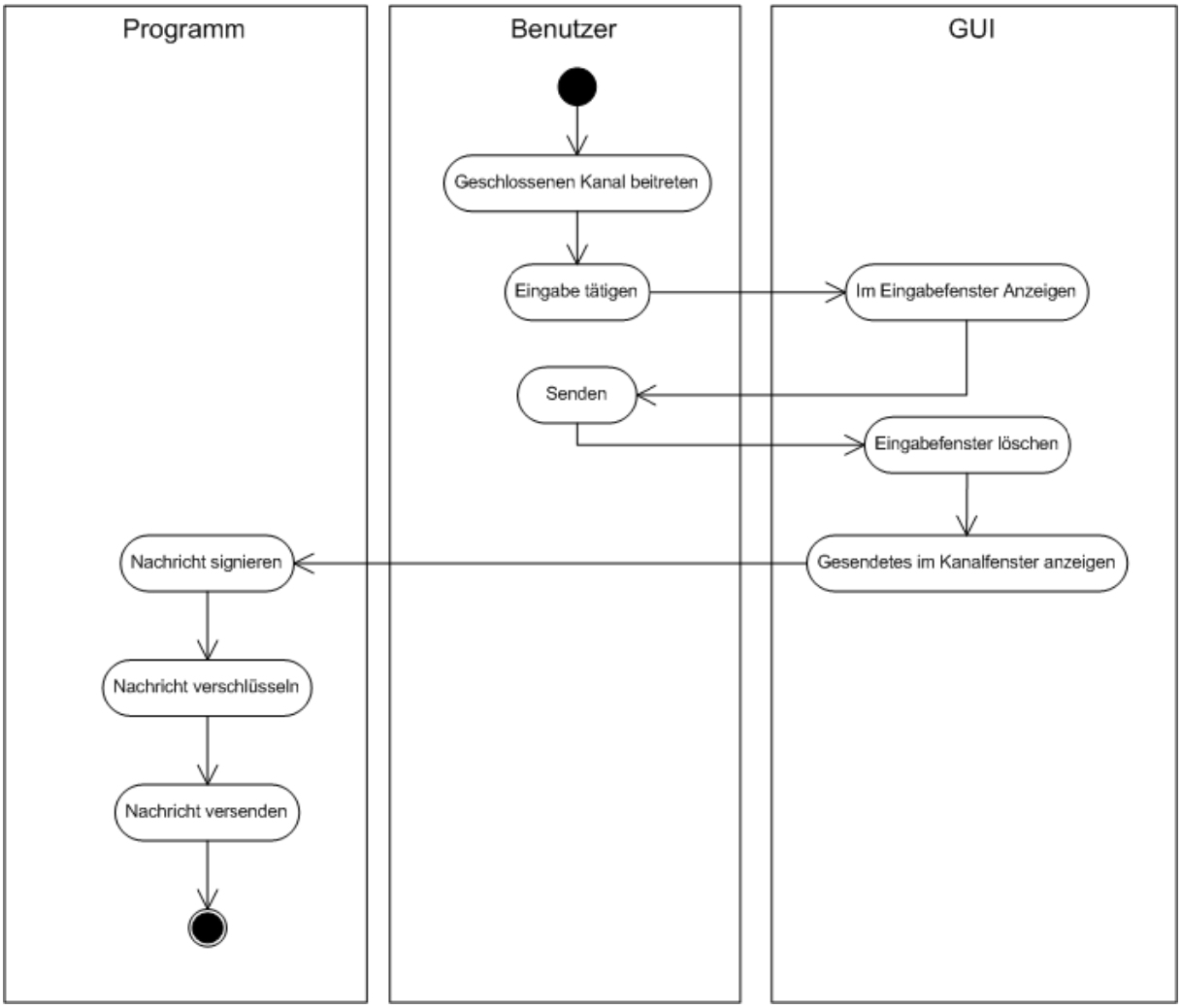

Abbildung 3: Aktivitätsdiagramm zum Senden in geschlossenen Kanal

Das Diagramm zeigt den Ablauf zum Senden in geschlossenen Kanälen. Bevor dies durchgeführt werden kann, muss sich der Benutzer in einem Kanal, entweder durch beitreten oder erstellen, befinden. Anschließend tätigt er eine Eingabe in einem Textfeld, welche er in Echtzeit sieht. Nachdem er diese gesendet hat, wird die Eingabe gelöscht und in ein Kanalfenster übertragen, damit der Benutzer sieht, dass seine Nachricht verschickt worden ist. Im Programm selbst findet dann zunächst das signieren, dann das verschlüsseln und letztendlich das versenden der Nachricht statt.

2.1.2 Feinanalyse

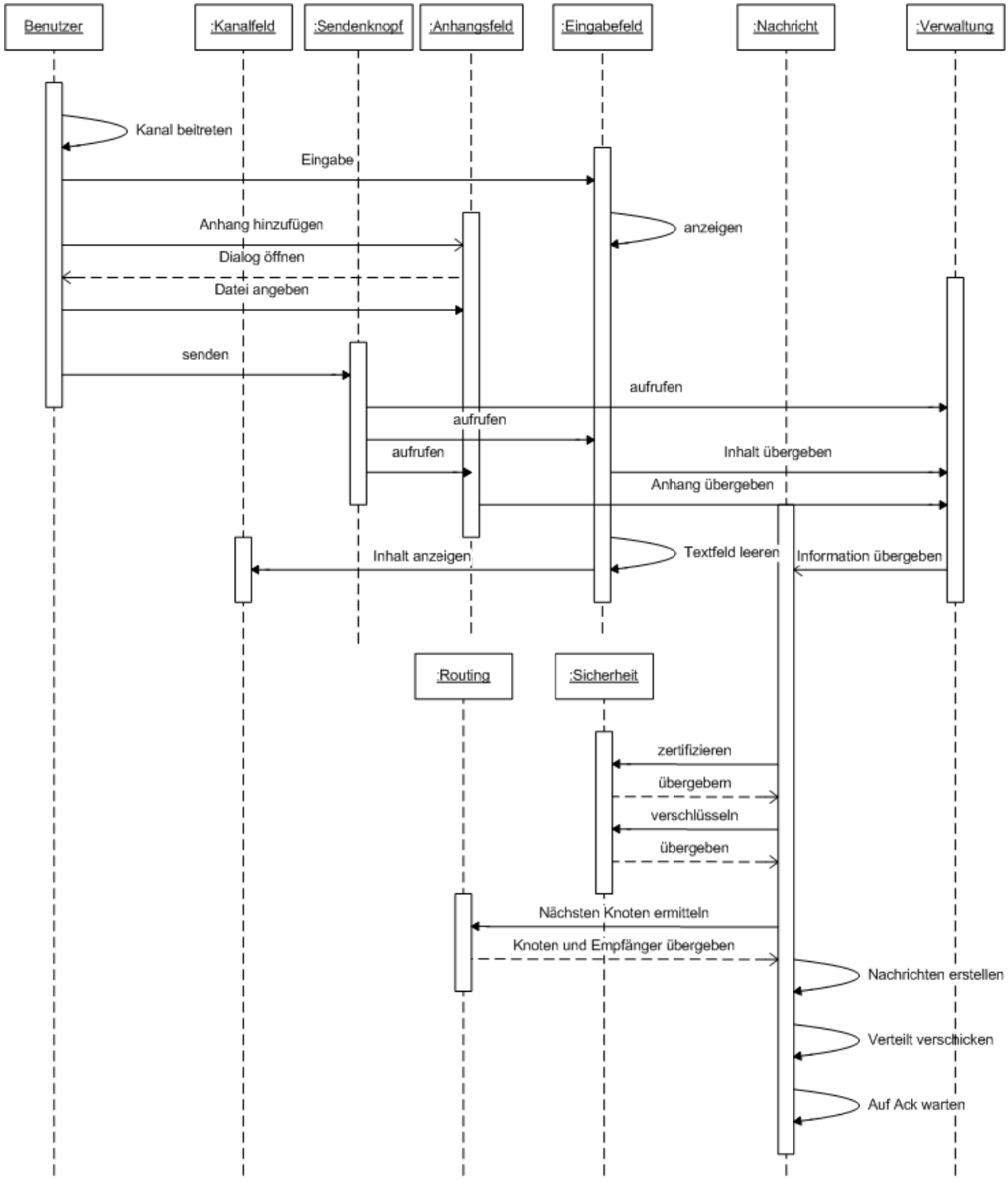

Abbildung 4: Sequenzdiagramm zum Senden im geschlossenen Kanal

Das Diagramm zeigt die Interaktion zwischen den Objekten beim Senden in geschlossenen Kanälen von Text mit Anhang. Der Benutzer interagiert mit dem Eingabefeld und dem Anhangsfeld, um seine Sendewünsche festzulegen. Nachdem er diese über ein Knopf oder Kommando versendet hat, wird je nachdem, ob ein Anhang angeben worden ist oder nicht, der Inhalt an die Verwaltung übergeben. Dieses gibt dann alle nötigen Informationen an das Nachrichtenobjekt. Anschließend zertifiziert und verschlüsselt es den Inhalt der Nachricht mit

Hilfe des Sicherheitsobjekts und fragt beim Routingobjekt an, wohin die Nachricht verschickt werden soll. Dann werden die Nachrichten erstellt und verschickt. Letztendlich wird noch auf eine Bestätigung für die Nachricht von dem nächsten Knoten gewartet.

2.2 Analyse von Funktionalität /F20/ : Nachricht im anonymen Kanal senden

Die Funktion führt das Senden einer Nachricht im anonymen Kanal durch.

2.2.1 Grobanalyse

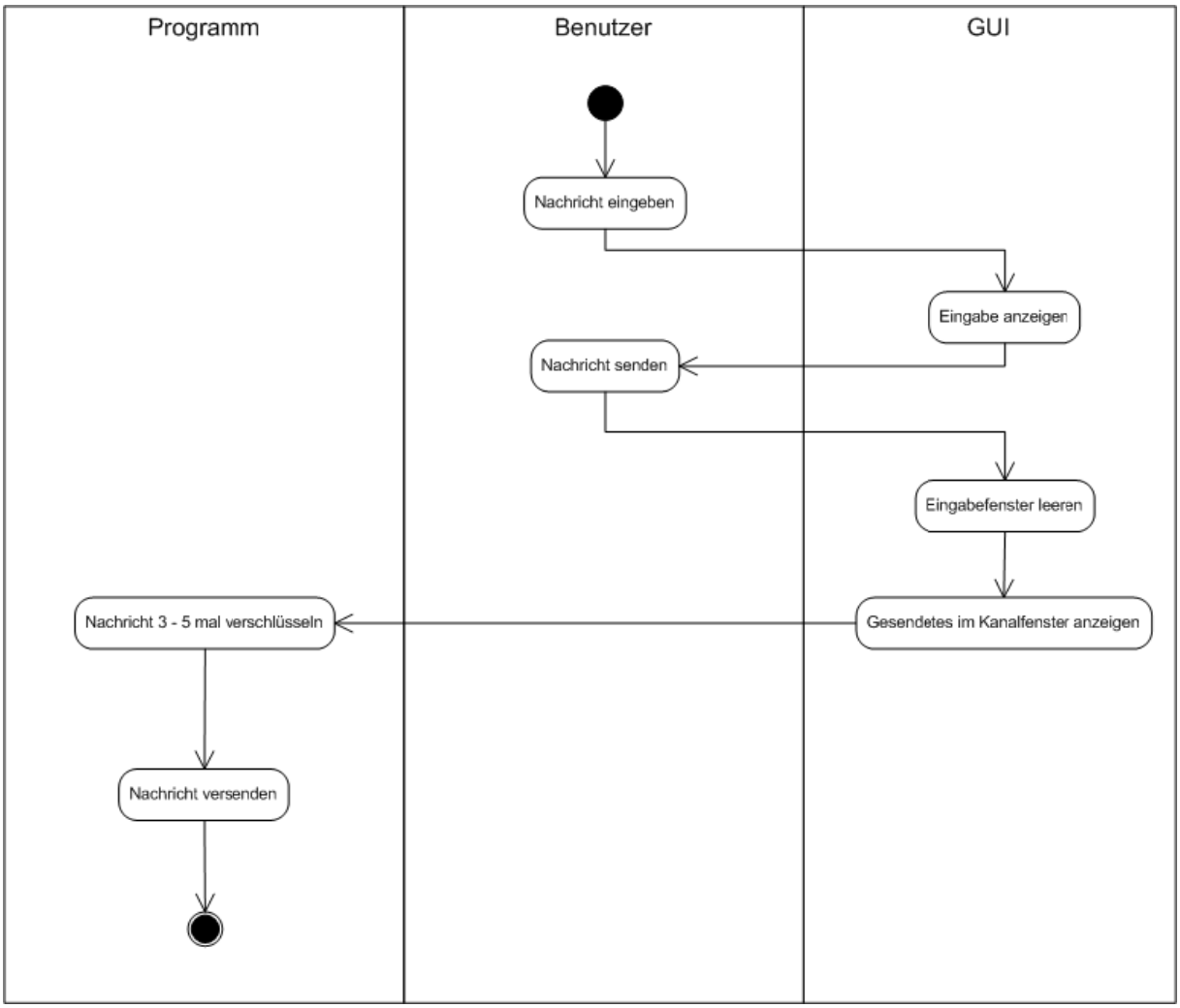

Abbildung 5: Aktivitätsdiagramm zum Senden im anonymen Kanal

Das Diagramm beschreibt den Ablauf zum Senden im anonymen Kanal. Der Benutzer gibt seine Nachricht ein und diese wird ihm von der GUI im Textfeld angezeigt. Anschließend gibt der Benutzer dem Programm den Auftrag, die Nachricht zu versenden, vorzugsweise über einen einfachen „Send“-Button, die GUI leert das Textfeld, schreibt den Text ins Kanalfenster und übergibt diesen an das Hauptprogramm. Dieses verschlüsselt die Nachricht mehrfach mit den öffentlichen Schlüsseln der nächsten 3-5 Benutzer, falls so viele Benutzer im Netz vorhanden sind. Das verschlüsseln erfolgt via Zwiebelprinzip: Die Verschlüsselung erfolgt wenn sich mindestens 4 Benutzer im Kanal befinden, die Nachricht wird dann mit den Schlüsseln der anderen 3 Benutzer verschlüsselt. Das Programm erstellt aus dem eingegeben Inhalt eine vollständige XML-Nachricht, in der das „flood“ auf „true“ gesetzt wird. Anschließend wird die komplette XML-Nachricht mit dem Schlüssel von Benutzer1 verschlüsselt und gilt nun als Inhalt einer neuen Nachricht, die wieder in das XML-Format gebracht wird und Benutzer1 wird als Empfänger eingetragen. Diese XML-Nachricht wird mit dem Schlüssel von Benutzer2 verschlüsselt, gilt nun wieder als Nachrichten Inhalt, wird ins XML-Format gebracht und Benutzer2 wird als Empfänger eingetragen, usw. Idealerweise wird die Nachricht 5-mal verschlüsselt, hierfür müssen sich aber mindestens 6 Teilnehmer im Kanal befinden.

2.2.2 Feinanalyse

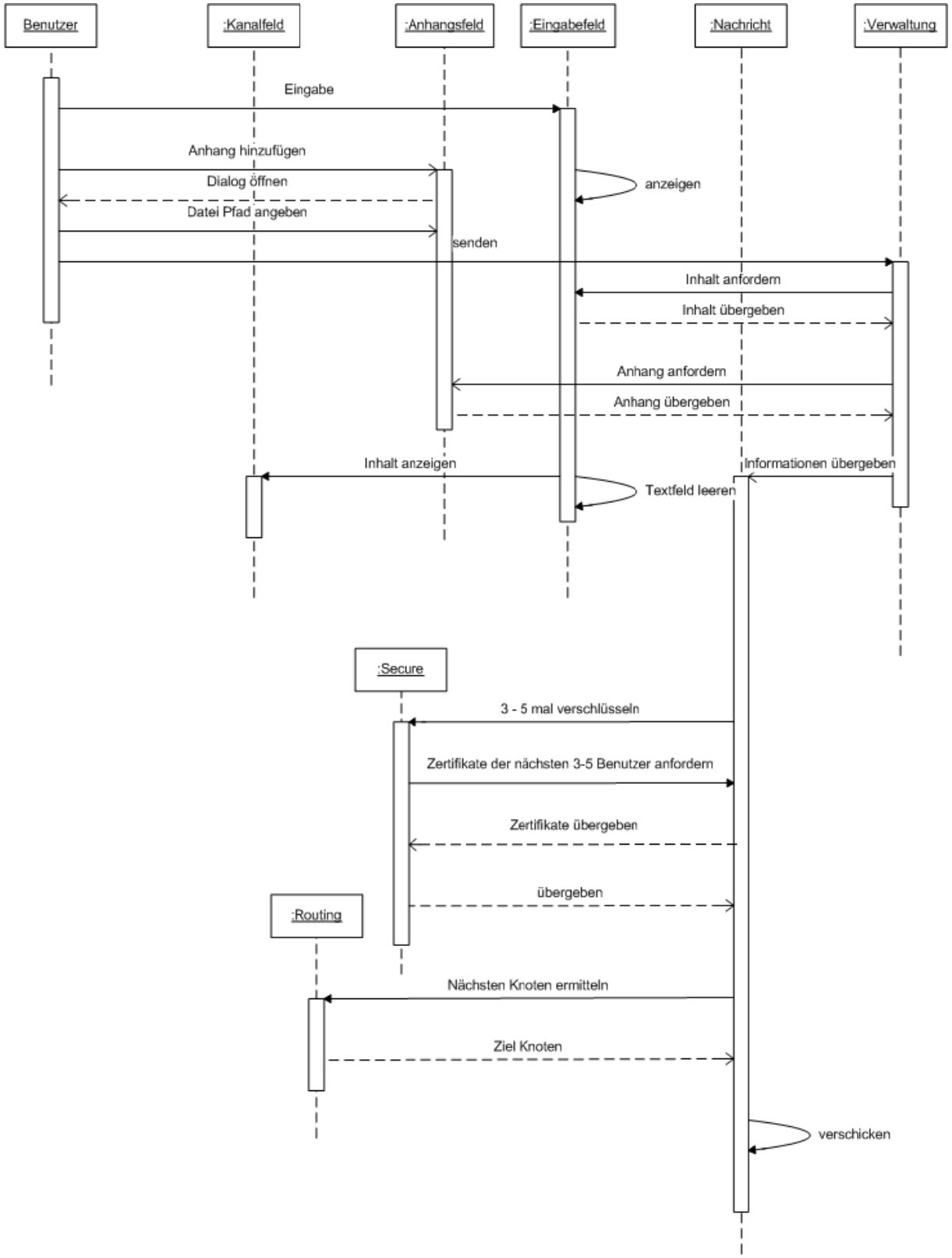

Abbildung 6: Sequenzdiagramm zum Senden im anonymen Kanal

Das Diagramm zeigt die Interaktion zwischen den Objekten beim senden im anonymen Kanal von Text mit Anhang. Der Benutzer ist beim Programmstart automatisch dem anonymen Kanal beigetreten und möchte nun eine Nachricht in diesem verschicken. Der Benutzer gibt seine Nachricht ein, diese wird ihm im Eingabefeld angezeigt, zusätzlich hat er die Möglichkeit eine Datei anzuhängen, diese Aktion spiegelt sich im Diagramm wieder, ist aber keine notwendige. Der Benutzer klickt auf senden, darauf hin schickt das Eingabefeld den eingegebenen Text und das Anhangsfeld die Pfadangabe an das Verwaltungsobjekt. Das Eingabefeld wird geleert und die eingegebene Nachricht dem Benutzer im Kanalfeld angezeigt. Die Verwaltung übergibt die gesammelten Informationen an das Nachrichtenobjekt. Von hier werden die Benutzer bestimmt, mit deren Schlüsseln die Nachricht via oben beschriebenes Zwiebelprinzip verschlüsselt werden. Die Verschlüsselung erfolgt dann im Secureobjekt, sollten hierfür die Zertifikate noch nicht vorhanden sein, werden sie via Nachricht angefordert. Danach wird der Inhalt wieder an das Nachricht-Objekt gegeben wo sie in das XML Nachrichten Format geparst werden. Zum Schluss fragt das Nachrichtenobjekt beim Routing-Objekt an, wohin die Nachricht verschickt werden soll, und verschickt diese.

2.3 Analyse von Funktionalität /F30/ : Nachricht im offenen Kanal senden

Die Funktion führt das Senden einer unverschlüsselten Nachricht im offenen Kanal durch.

2.3.1 Grobanalyse

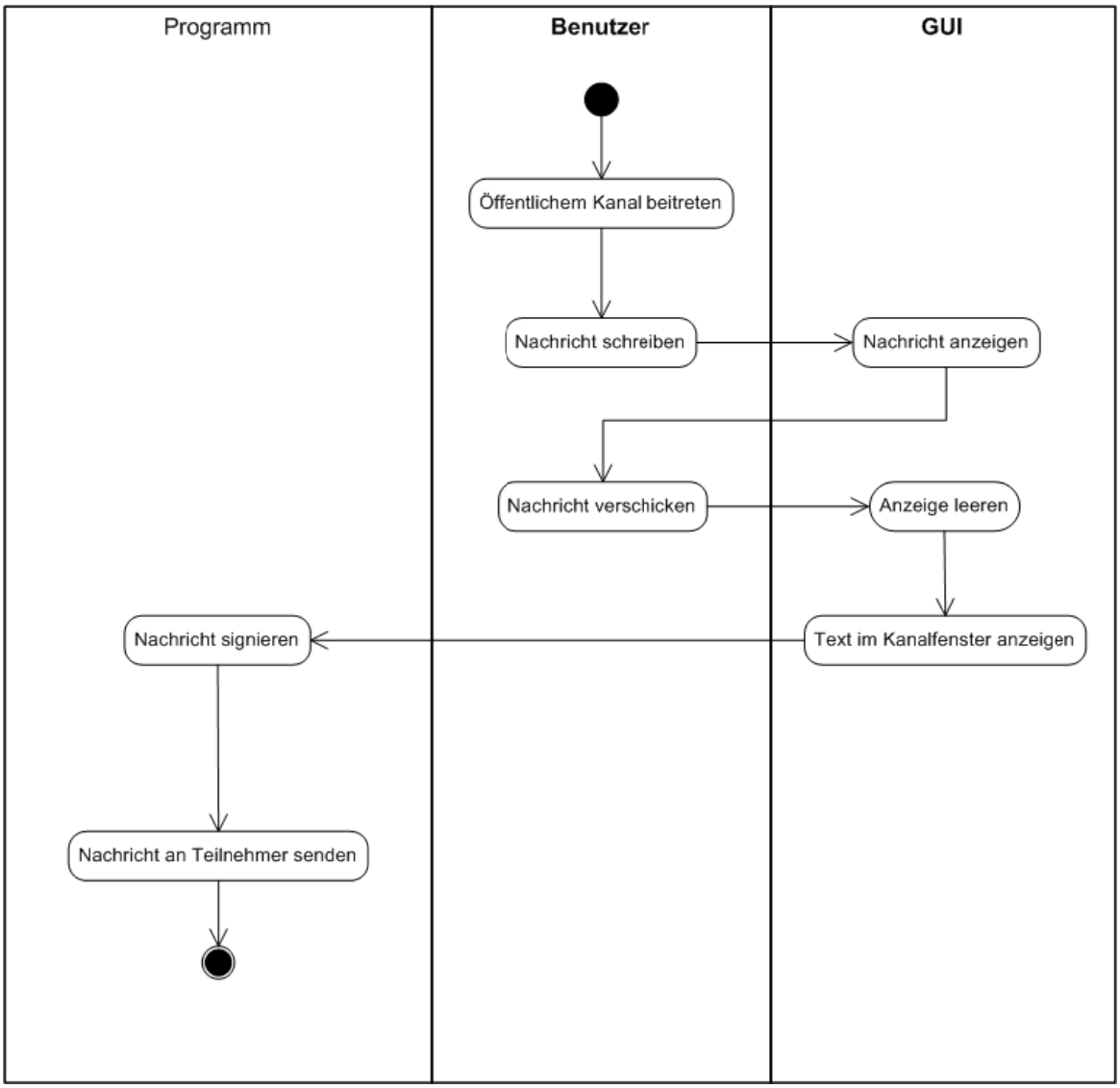

Abbildung 7: Aktivitätsdiagramm zum Senden im offenen Kanal

Das Diagramm zeigt den Ablauf zum Senden in offenen Kanälen. Bevor dies durchgeführt werden kann, muss sich der Benutzer in einem Kanal, entweder durch beitreten oder erstellen, befinden. Anschließend tätigt er eine Eingabe in einem Textfeld, welche er in Echtzeit sieht. Nachdem er diese gesendet hat, wird die Eingabe gelöscht und in ein Kanalfenster übertragen, damit der Benutzer sieht, dass seine Nachricht verschickt worden ist. Im Programm selbst findet dann zunächst das signieren und letztendlich das versenden der Nachricht statt.

2.3.2 Feinanalyse

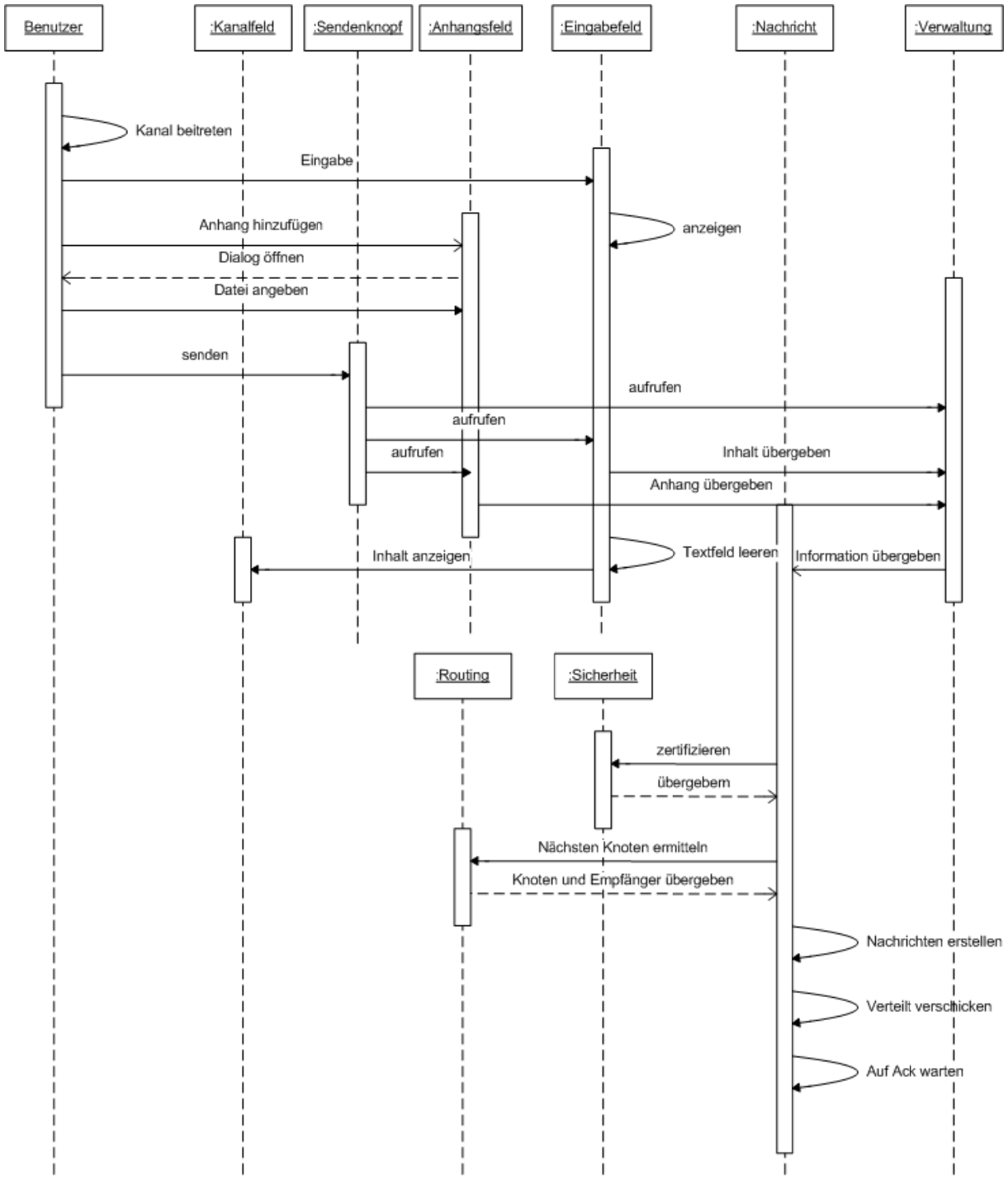

Abbildung 8: Sequenzdiagramm zum unverschlüsselten Senden im offenen Kanal

Das Diagramm zeigt die Interaktion zwischen den Objekten beim Senden in offenen Kanälen von Text mit Anhang. Der Benutzer interagiert mit dem Eingabefeld und dem Anhangsfeld, um seine Sendewünsche festzulegen. Nachdem er diese über ein Knopf oder Kommando versendet hat, wird je nachdem, ob ein Anhang angeben worden ist oder nicht, der Inhalt an die Verwaltung übergeben. Dieses gibt dann alle nötigen Informationen an das Nachrichtenobjekt. Anschließend zertifiziert es die Nachricht mit Hilfe des Sicherheitsobjekts und fragt beim Routingobjekt an, wohin die Nachricht verschickt werden soll. Dann werden die Nachrichten erstellt und verschickt. Letztendlich wird noch auf eine Bestätigung für die Nachricht von dem nächsten Knoten gewartet.

2.4 Analyse von Funktionalität /F40/ : Nachricht im geschlossenen Kanal empfangen

Die Funktion führt das Empfangen von Nachrichten durch, falls sie für einen geschlossenen Kanal bestimmt sind.

2.4.1 Grobanalyse

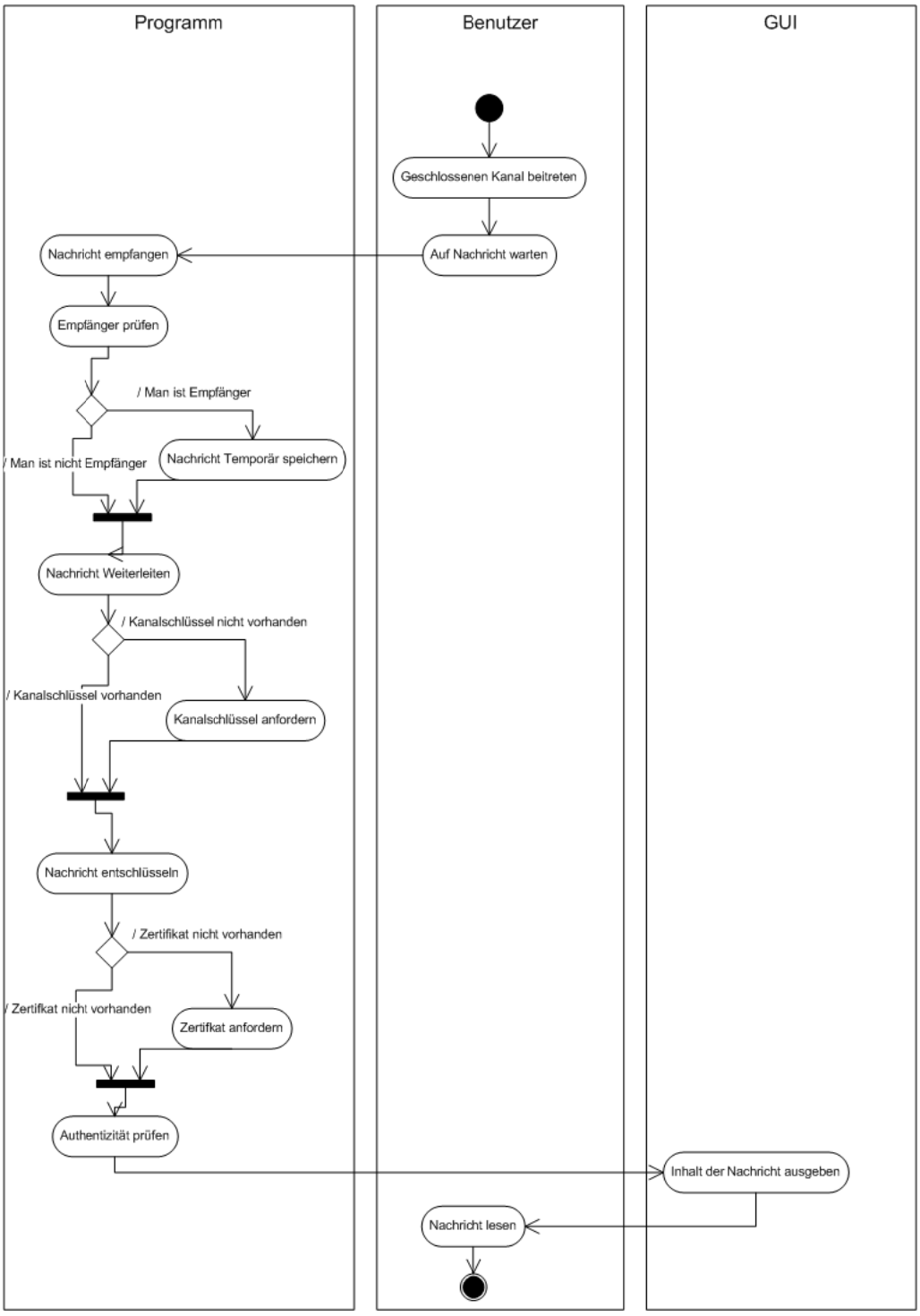

Abbildung 9: Aktivitätsdiagramm zum Empfangen im geschlossenen Kanal

Das Diagramm zeigt den Ablauf für das Empfangen von Nachrichten im geschlossenen Kanal. Bevor eine Nachricht Empfangen und ausgegeben wird, muss der Benutzer sich in dem Kanal befinden. Sobald eine Nachricht eintrifft, wird geprüft, ob man Empfänger der Nachricht ist. Wenn man Empfänger ist, wird sie gespeichert und dann gleich weitergesendet (/F70), um keine Zeit zu verschwenden. Ist der Kanalschlüssel vorhanden wird sie entschlüsselt, andernfalls muss der Schlüssel angefordert werden. Der gleiche Ablauf wird auch zum Authentifizieren durchgeführt. Dann wird die Nachricht ausgegeben.

2.4.2 Feinanalyse

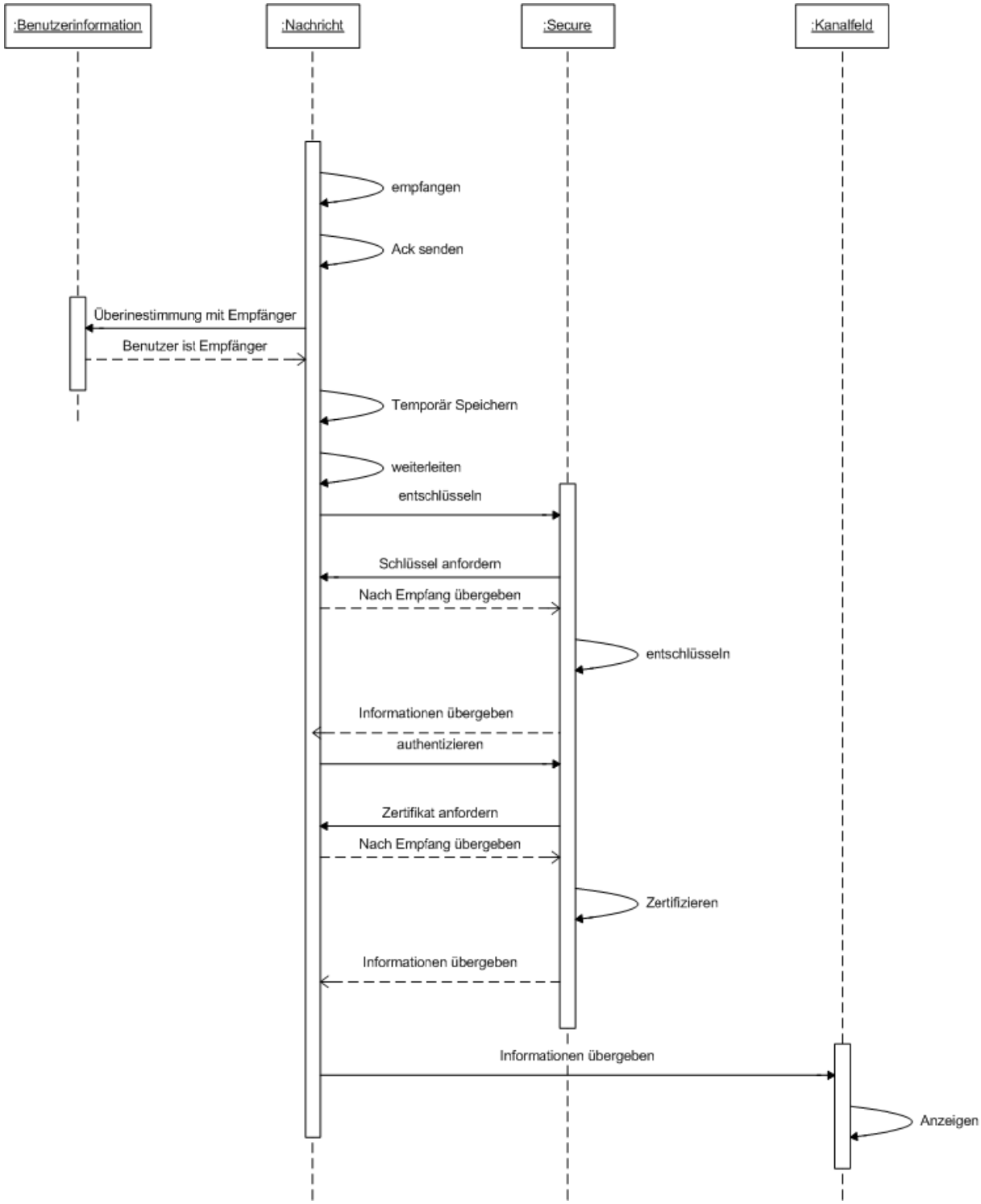

Abbildung 10: Sequenzdiagramm zum Empfangen im geschlossenen Kanal

Das Diagramm beschreibt das Empfangen einer Nachricht für einen geschlossenen Kanal, wobei gemeinsamer Schlüssel und Zertifikat nicht vorhanden sind. Nachdem eine Nachricht empfangen worden ist, wird eine Bestätigung(Ack) an den vorherigen Knoten gesendet. Dann wird in den Benutzerinformationen abgefragt, ob die Nachricht für einen bestimmt ist. Wenn das der Fall ist, wird die Nachricht temporär gespeichert und weitergeleitet (/F70),.

Dann gibt das Nachrichtobjekt dem Sicherheitsobjekt die Aufgabe die Nachricht zu entschlüsseln bzw. zu authentifizieren. Dies fordert über ein anderes Nachrichtenobjekt den gemeinsamen Schlüssel bzw. das Zertifikat an und führt dann die Authentifizierung und Entschlüsselung durch. Anschließend werden die neuen Daten wieder übergeben und an das Kanalfeld weitergeleitet, welches dann die Nachricht entsprechend ausgibt.

2.5 Analyse von Funktionalität /F50/ : Nachricht im anonymen Kanal empfangen

Die Funktion führt das Empfangen einer Nachricht im anonymen Kanal durch.

2.5.1 Grobanalyse

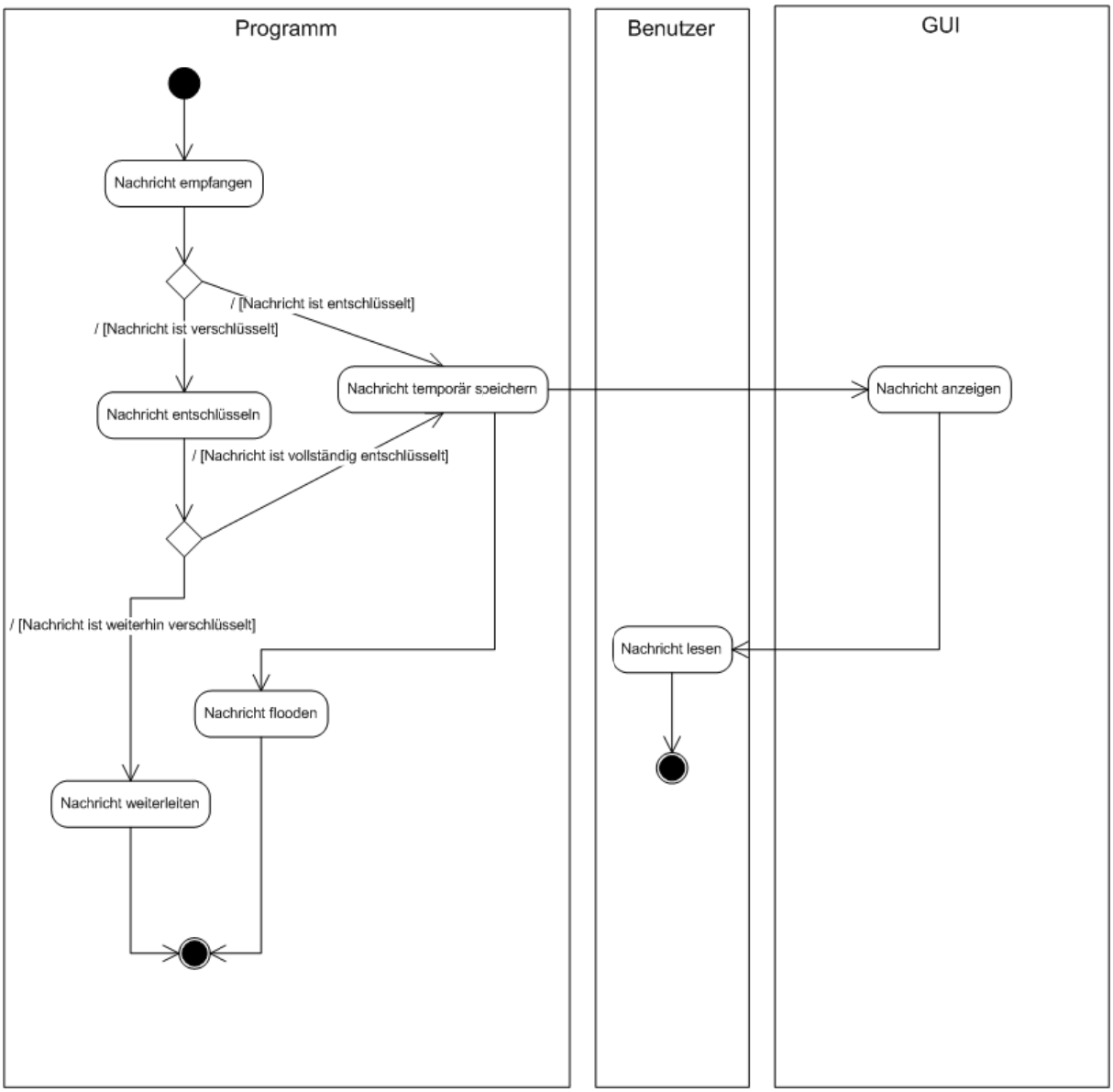

Abbildung 11: Aktivitätsdiagramm zum Empfangen im anonymen Kanal

Das Diagramm beschreibt den Ablauf beim Empfangen einer Nachricht im anonymen Kanal. Die Entscheidung ob eine Nachricht entschlüsselt ist oder nicht, erfolgt über den Nachrichten Typ. Ist sie verschlüsselt, liegt sie als OBSCURE Nachricht vor, ist sie Entschlüsselt, handelt es sich um eine normale MESSAGE Nachricht.

Trifft eine bereits entschlüsselte Nachricht ein, so wird sie wird per Flooding weitergeleitet und dem Benutzer zum Lesen im Kanalfenster angezeigt. Trifft eine verschlüsselte Nachricht ein, wird sie mit dem privaten Schlüssel entschlüsselt, sollte sie dann endgültig entschlüsselt sein, liegt sie nun als normale MESSAGE Nachricht vor. Die Nachricht wird zunächst temporär gespeichert, dann im Kanal geflutet und dem Benutzer angezeigt. Ist die Nachricht jedoch immer noch verschlüsselt, wird sie direkt an den nächsten Empfänger weitergeleitet.

2.5.2 Feinanalyse

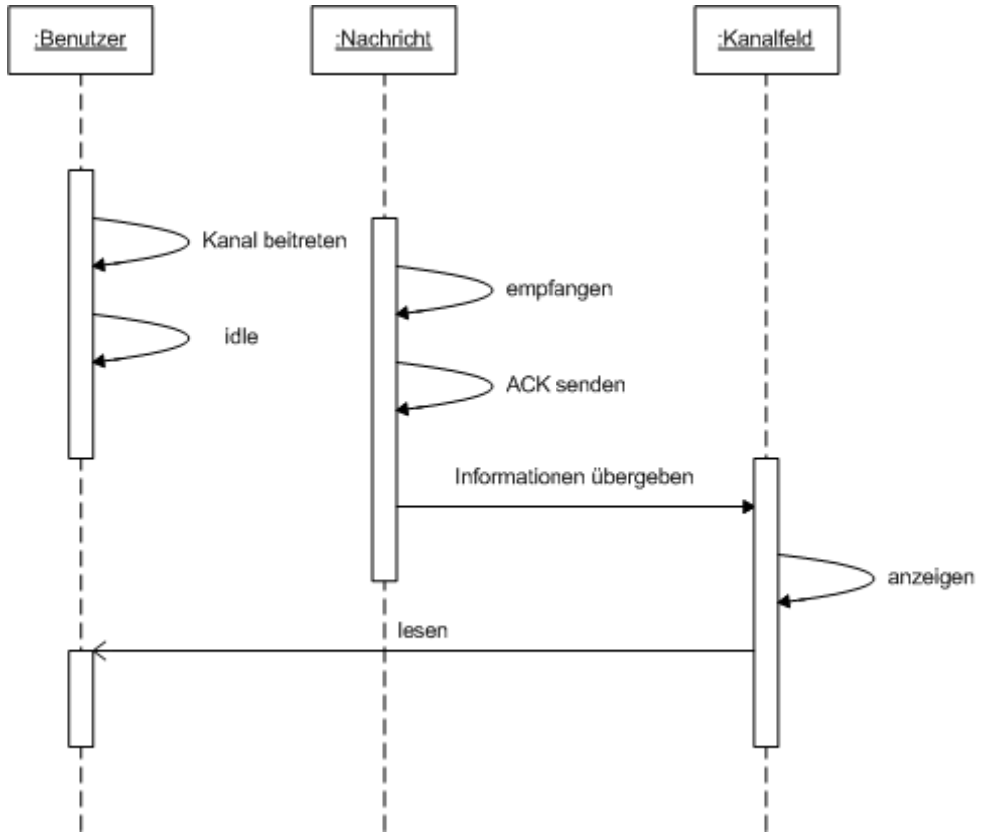

Abbildung 12: Sequenzdiagramm zum Empfangen im anonymen Kanal, falls die Nachricht entschlüsselt eintrifft

Das Diagramm beschreibt das Nachrichten empfangen im anonymen Kanal, sofern diese bereits vollständig entschlüsselt wurde. Der Benutzer befindet sich beim Programmstart automatisch im anonymen Kanal und ist zunächst nicht aktiv beteiligt. Eine Nachricht trifft ein, dies wird von dem Nachrichtobjekt erkannt, es wird ein ACK an den Sender geschickt und die Nachricht wird im Kanal weiter geflutet. Anschließend wird die Nachricht aus dem XML Format geparst und dem Benutzer zum Lesen im Kanalfeld angezeigt.

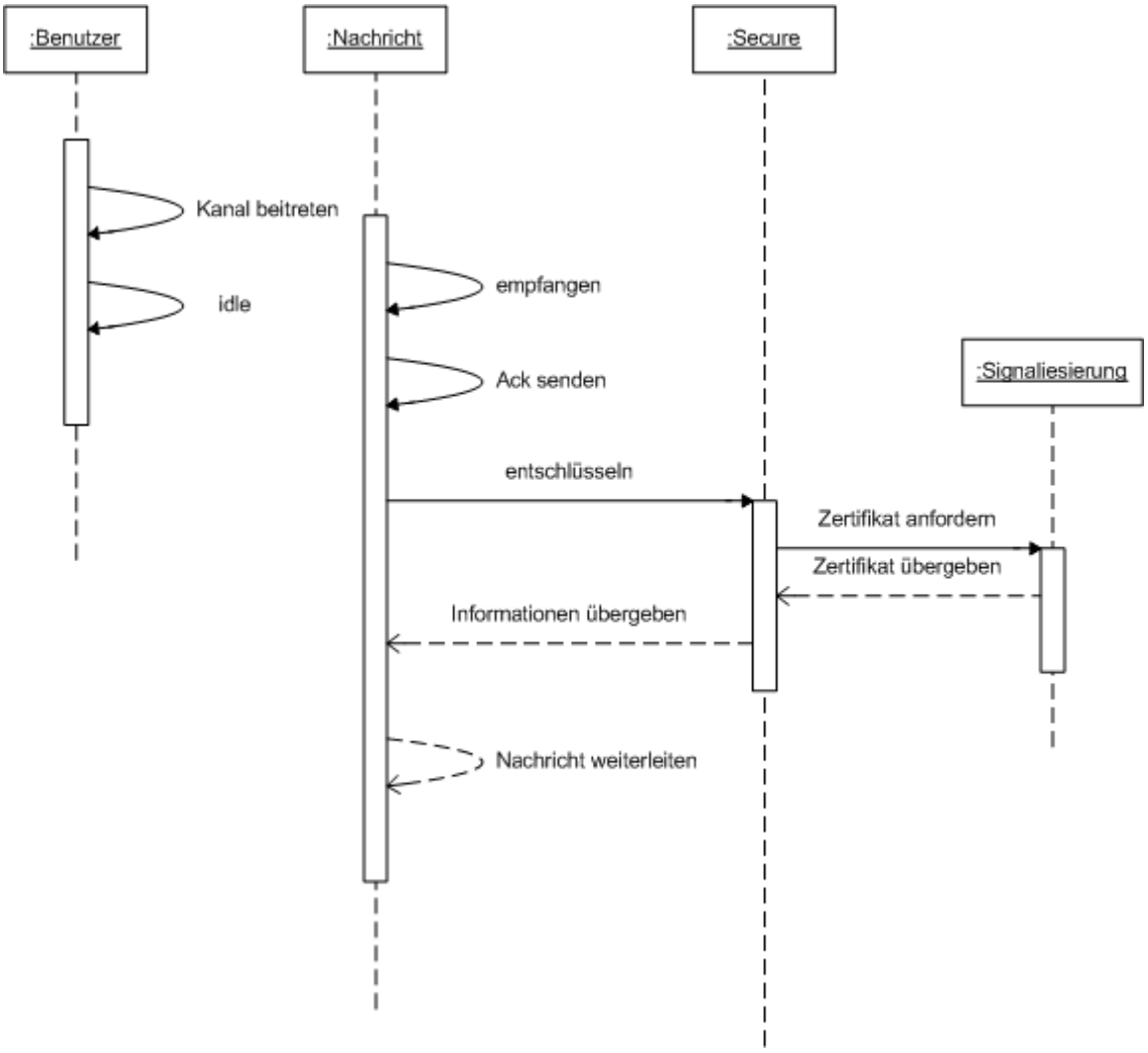

Abbildung 13: Sequenzdiagramm zum Empfangen im anonymen Kanal, falls die Nachricht verschlüsselt eintrifft

Das Diagramm beschreibt das Nachrichten empfangen im anonymen Kanal, sofern diese noch nicht entschlüsselt wurde. Der Benutzer befindet sich beim Programmstart automatisch im anonymen Kanal und ist nicht aktiv beteiligt. Eine Nachricht trifft ein, dies wird von dem Nachrichtobjekt erkannt und es wird ein ACK an den Sender geschickt. Danach wird die Nachricht mit dem eigenen Zertifikat entschlüsselt, aus dem entschlüsselten Inhalt entsteht eine neue OBSCURE Nachricht die den nächsten Empfänger enthält, an diesen wird die Nachricht dann gesendet.

2.6 Analyse von Funktionalität /F60/ : Nachricht im offenen Kanal empfangen

Die Funktion führt das Empfangen von Nachrichten durch, falls sie für einen offenen Kanal bestimmt sind.

2.6.1 Grobanalyse

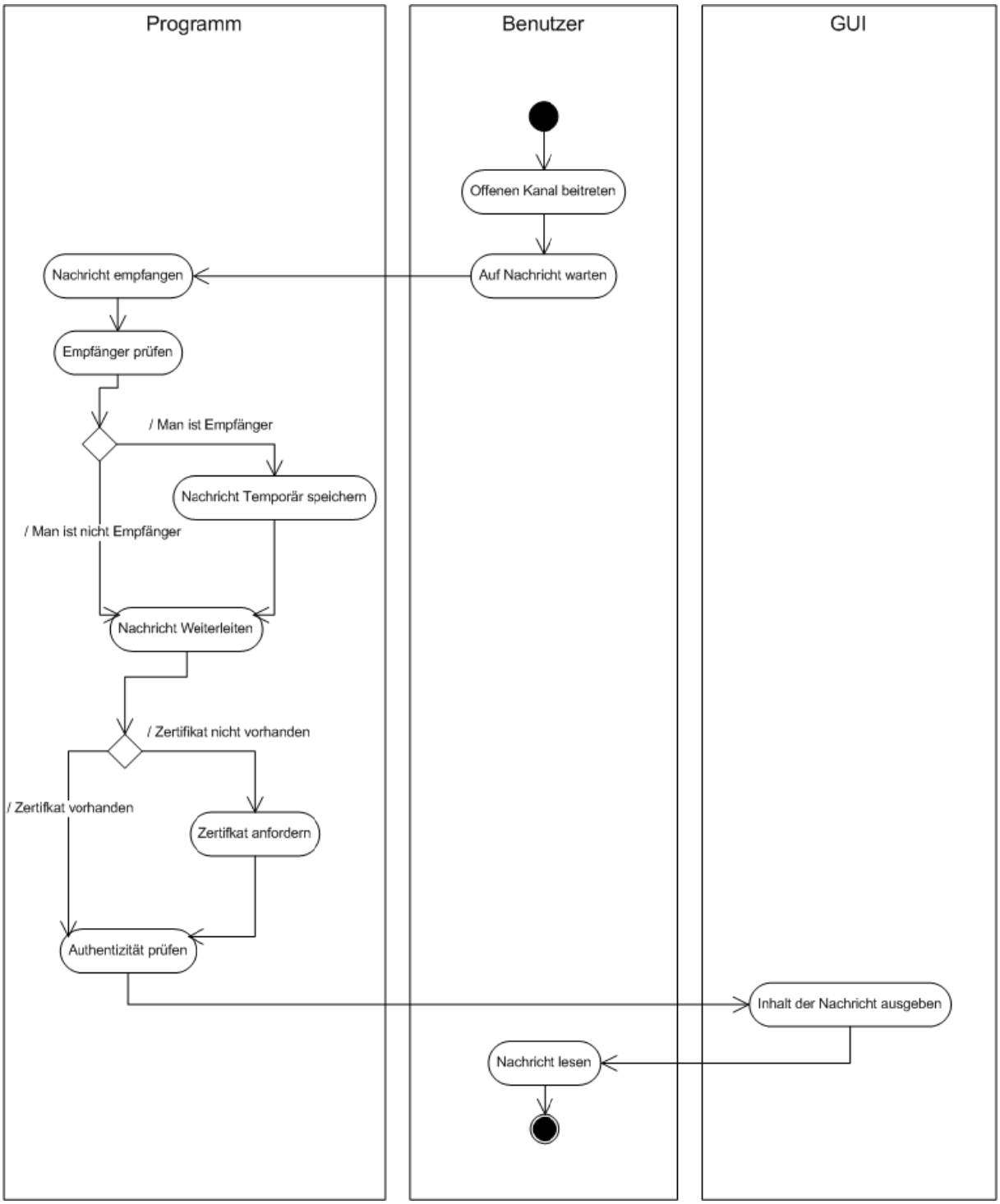

Abbildung 14: Aktivitätsdiagramm zum Empfangen im offenen Kanal

Das Diagramm zeigt den Ablauf für das Empfangen von Nachrichten im offenen Kanal. Bevor eine Nachricht Empfangen und ausgegeben wird, muss der Benutzer sich in dem Kanal befinden. Sobald eine Nachricht eintrifft, wird geprüft, ob man Empfänger der Nachricht ist. Wenn man Empfänger ist, wird sie gespeichert und dann gleich weitergesendet, um keine Zeit zu verschwenden. Ist das Zertifikat vorhanden wird sie authentifiziert, andernfalls muss das Zertifikat angefordert werden. Letztendlich wird sie dem Benutzer über die GUI ausgegeben.

2.6.2 Feinanalyse

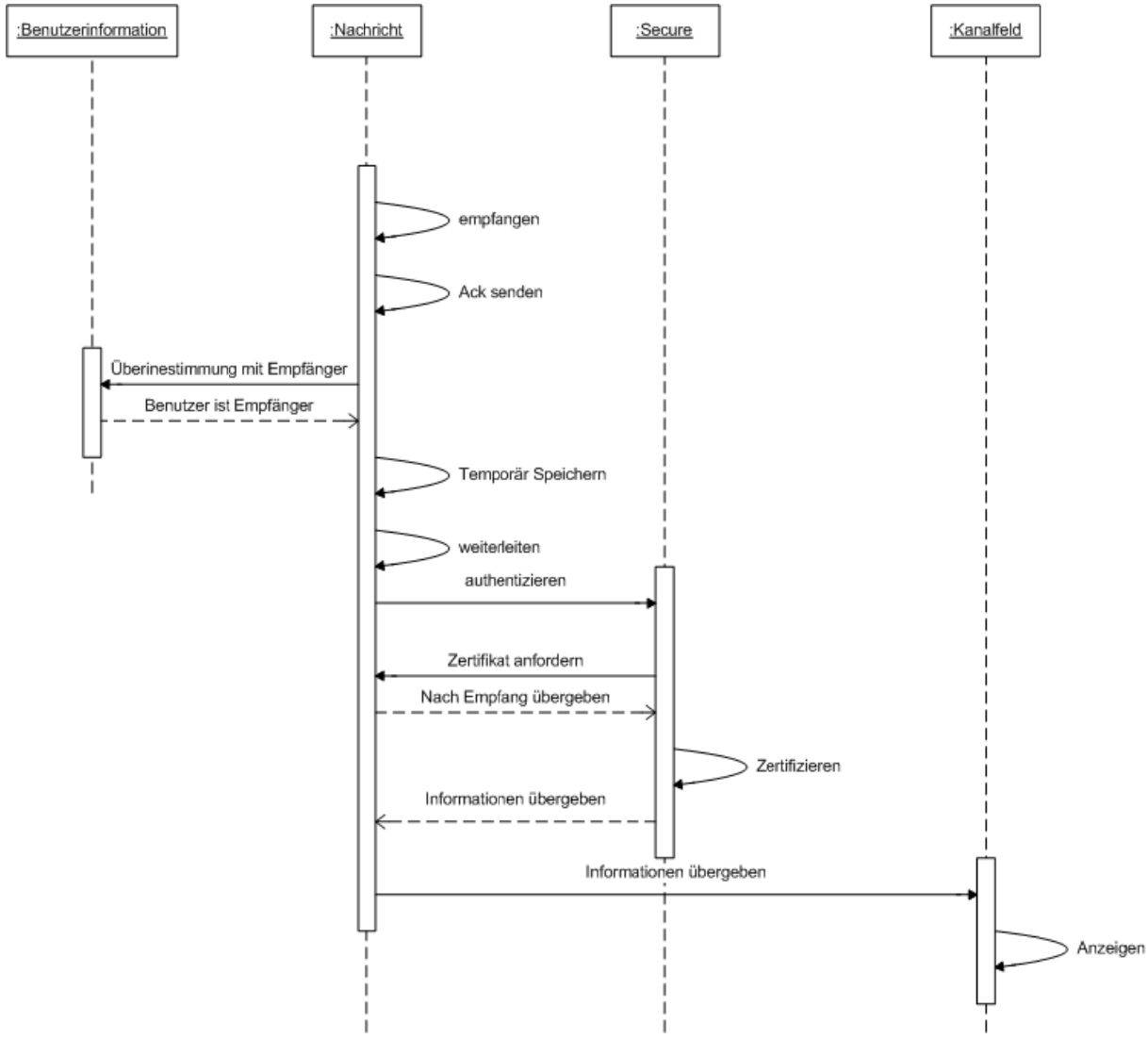

Abbildung 15: Sequenzdiagramm zum Empfangen im offenen Kanal

Das Diagramm beschreibt das Empfangen einer Nachricht für einen offenen Kanal, wobei das Zertifikat nicht vorhanden ist. Nachdem eine Nachricht empfangen worden ist, wird eine Bestätigung(Ack) an den vorherigen Knoten gesendet. Dann wird in den Benutzerinformationen abgefragt, ob die Nachricht für einen bestimmt ist. Wenn das der Fall ist, wird die Nachricht temporär gespeichert und weitergeleitet. Dann gibt das Nachrichtobjekt dem Sicherheitsobjekt die Aufgabe die Nachricht zu authentifizieren. Dies fordert über ein anderes Nachrichtenobjekt das Zertifikat an und führt dann die Authentifizierung durch. Anschließend werden die Daten wieder übergeben und an das Kanalfeld weitergeleitet, welches dann die Nachricht entsprechend ausgibt.

2.7 Analyse von Funktionalität /F70/ : Nachricht weiterleiten

Die Funktion hat die Aufgabe eintreffende Nachrichten weiterzuleiten, falls noch andere die Nachricht erhalten sollen.

2.7.1 Grobanalyse

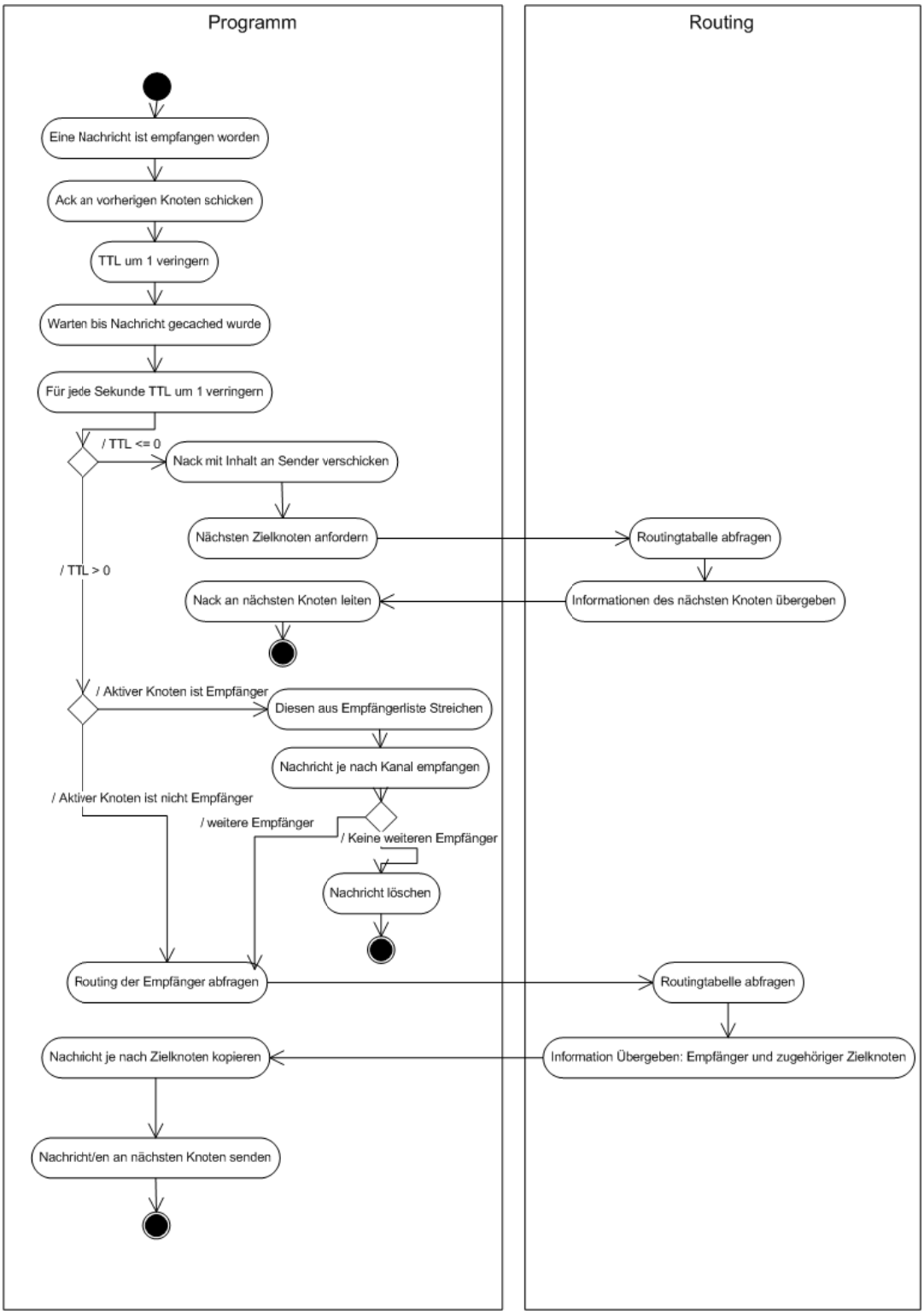

Abbildung 16: Aktivitätsdiagramm zum Weiterleiten von Nachrichten

Das Diagramm zeigt wie das Programm reagiert, wenn eine Nachricht weitergeleitet werden soll. Nachdem eine Nachricht empfangen worden ist, wird zunächst eine Bestätigung an den vorherigen Knoten geschickt (Ack). Dann wird die TTL um 1 verringert und für jede weitere Sekunde, die die Nachricht nicht weitergesendet wurde, ebenfalls um 1 verringert. Entweder die TTL ist abgelaufen, dann wird eine negative Bestätigung an den Sender geschickt, oder sie ist nicht abgelaufen, dann wird sie im Bezug auf die Empfänger aktualisiert. Die Nachricht wird dann je nach Kanal, wie die Funktionen /F40/, /F50/ und /F60/ zeigen, empfangen. Dann wird die Nachricht an die weiteren Empfänger durch Abrufen der Routinginformationen weitergeleitet.

2.7.2 Feinanalyse

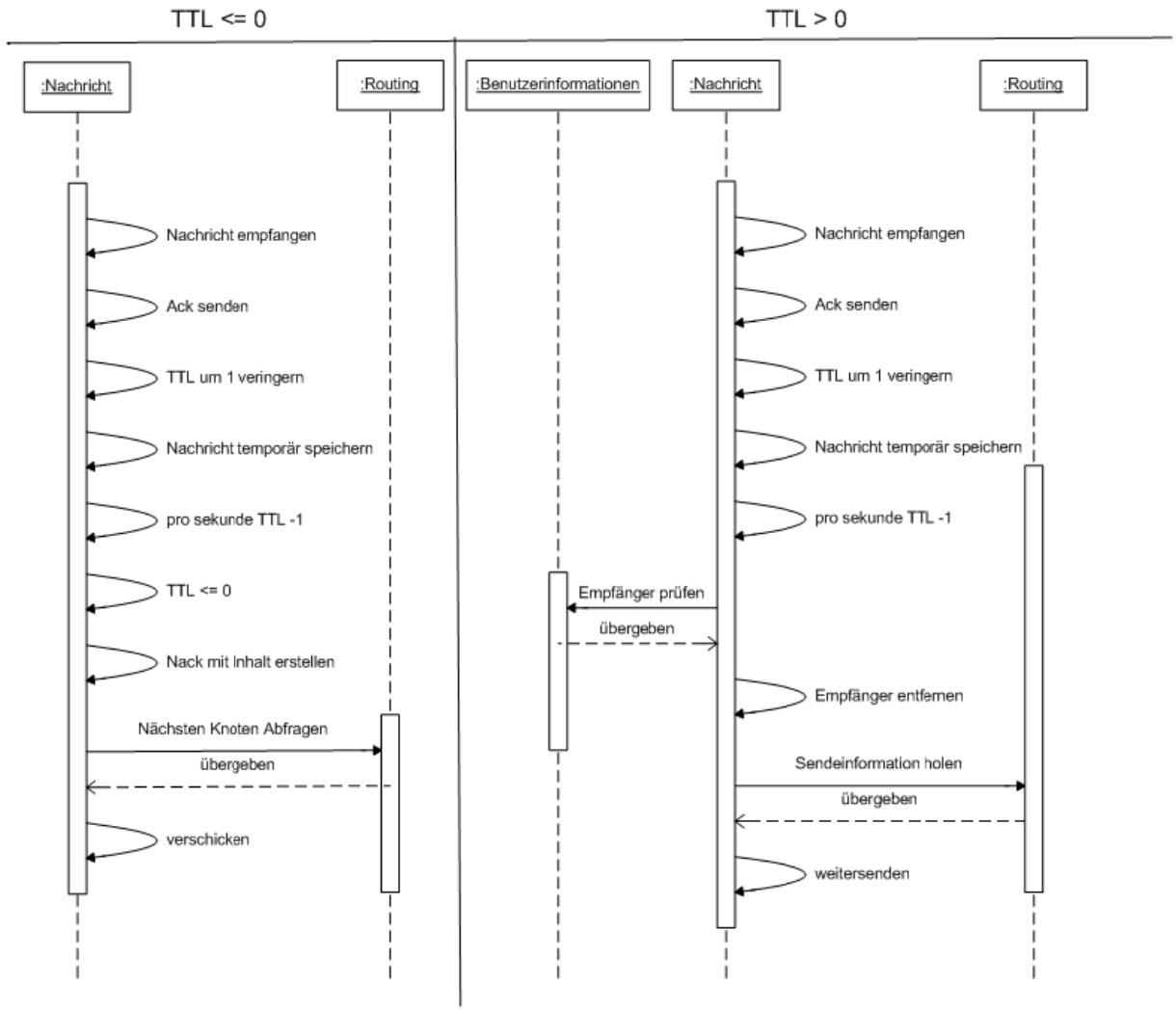

Abbildung 17: Sequenzdiagramm zum Weiterleiten von Nachrichten

Die Diagramme zeigen die Interaktion der Objekte zum Weiterleiten einmal mit und einmal ohne Ablauf der TTL. Das Senden der Bestätigung und das Verringern der TTL passiert innerhalb des Nachrichtenobjekts. Im Fall TTL <=0 wird eine negative Bestätigung erstellt, die Routinginformation abgefragt und dann verschickt. Im anderen Fall wird geprüft, ob der aktive Knoten ein Empfänger ist. Dann wird dieser aus der Nachricht entfernt und je nach Kanal(F40, F50, F60) empfangen. Für die restlichen Empfänger werden die Routinginformationen geholt und dementsprechend Nachrichten erstellt und verschickt. Bei beiden Fällen wird am Ende auf den Empfang einer Bestätigung gewartet. Falls diese nach einem Timeout nicht ankommt, wird sie erneut gesendet. Kommt die Bestätigung an brauch sie nicht länger gespeichert zu werden.

2.8 Analyse von Funktionalität /F80/ : Geschlossenen Kanal erstellen

Die Funktion ist für das Erstellen eines geschlossenen Kanals zuständig.

2.8.1 Grobanalyse

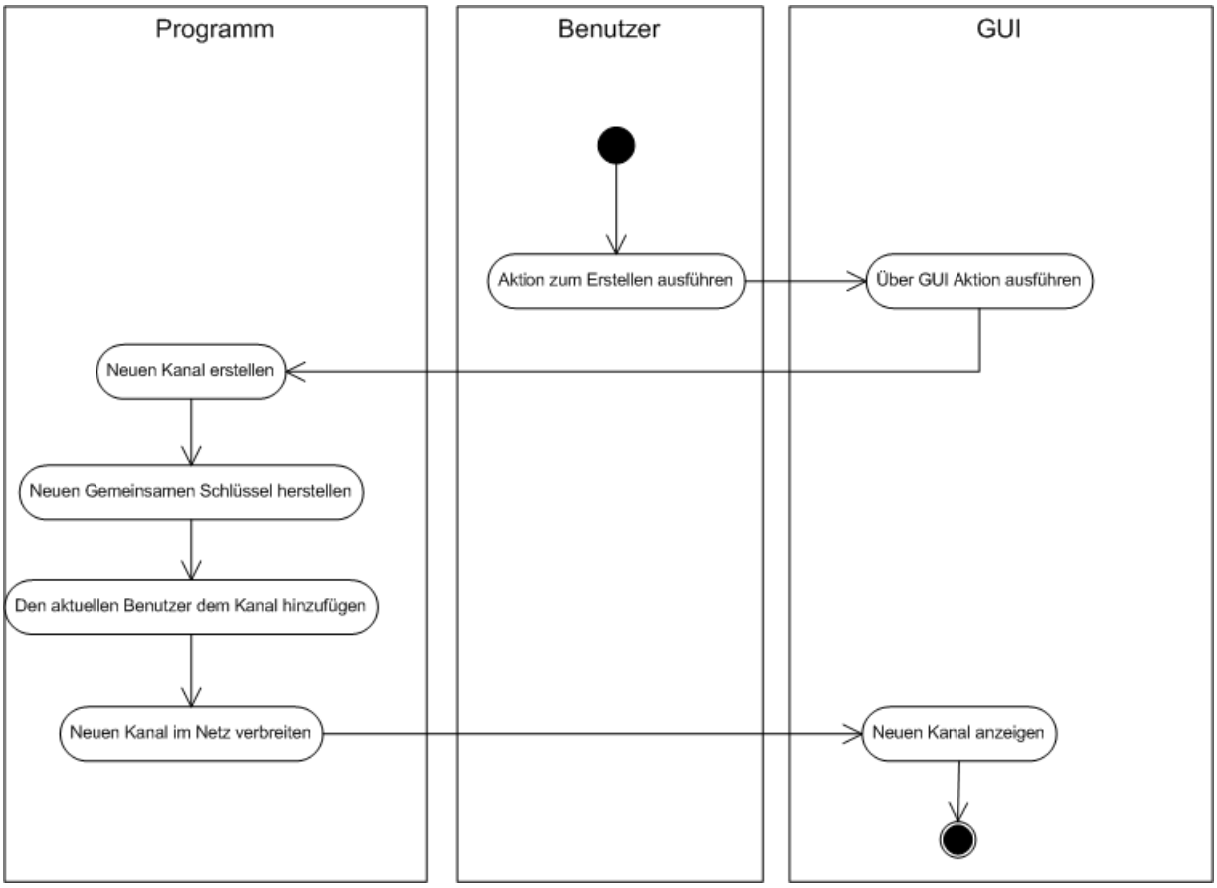

Abbildung 18: Aktivitätsdiagramm zum Erstellen eines geschlossenen Kanals

Das Diagramm zeigt was beim Erstellen eines geschlossenen Kanals abläuft. Nachdem der Benutzer über die GUI den Wunsch mitteilt, einen geschlossenen Kanal zu erstellen, wird innerhalb des Programms auch ein Kanal erstellt. Dann wird für den Kanal ein gemeinsamer Schlüssel generiert. Der Ersteller wird automatisch dem Kanal hinzugefügt und anschließend wird der Kanal im Netz verbreitet, damit die anderen Knoten wissen, dass ein neuer Kanal erstellt worden ist. Letztendlich wird durch neues aktives Kanalfenster und Reiter der Kanal dem Benutzer angezeigt.

2.8.2 Feinanalyse

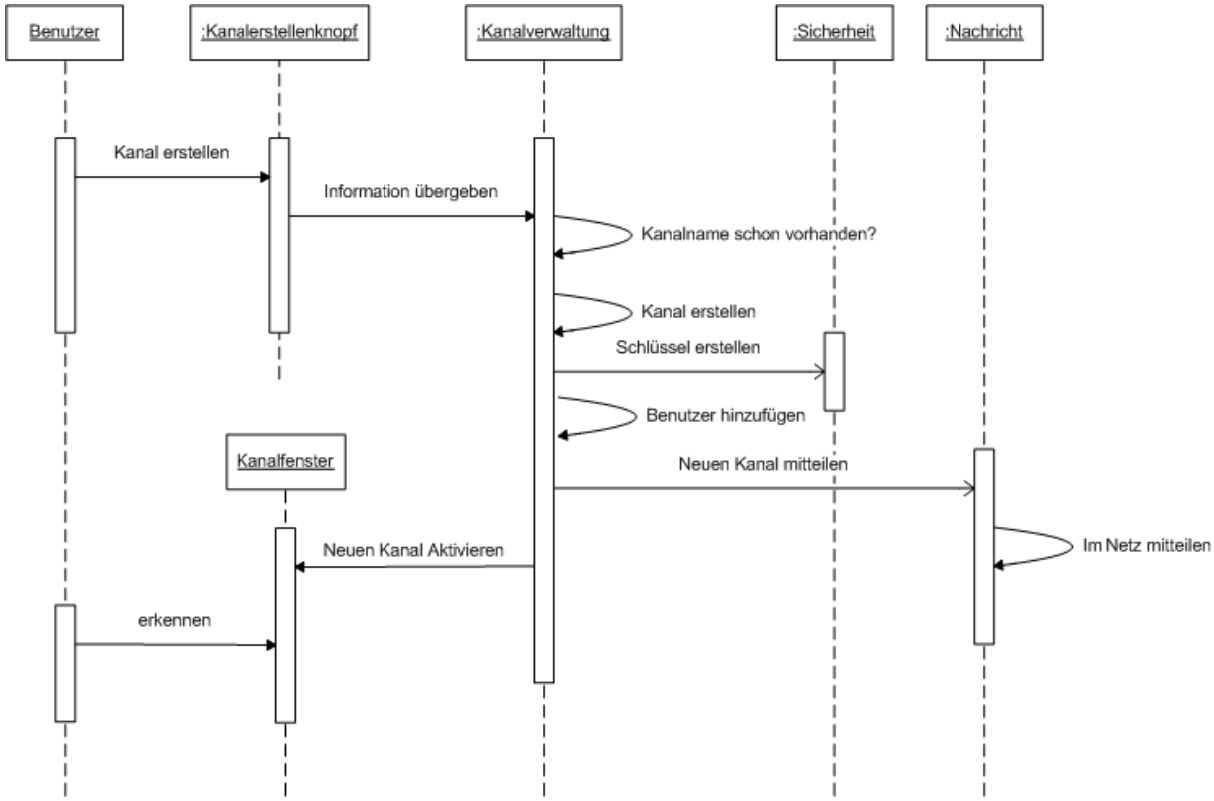

Abbildung 19: Sequenzdiagramm zum Erstellen eines geschlossenen Kanals

Das Diagramm zeigt die Interaktion der Objekte zum Erstellen eines geschlossenen Kanals. Über einen Knopf oder ein Kommando wird der Kanalverwaltung die Information übergeben, dass ein Kanal erstellt werden sollen. Dazu gehört auch der vom Benutzer gewählte Namen. Falls dieser Name schon vorhanden ist, wird die Erstellung verweigert und das dem Benutzer angezeigt. Die Verwaltung erstellt dann einen Kanal, generiert die zugehörige ID, lässt einen zugehörigen Schlüssel erstellen, und fügt den Benutzer hinzu, der den Kanal erstellt hat. Der Schlüssel wird bei dem Benutzer selbst gespeichert. Anschließend teilt die Verwaltung den anderen Knoten im Netz über das Nachrichtenobjekt durch eine Channel-Nachricht mit, das ein neuer Kanal existiert. Letztendlich wird einem Kanalfensterobjekt aufgetragen, dass es einen neuen Kanal erstellen soll, der vom Benutzer erkennbar ist.

2.9 Analyse von Funktionalität /F90/ : Offenen Kanal erstellen

Die Funktion ist für das Erstellen eines offenen Kanals zuständig.

2.9.1 Grobanalyse

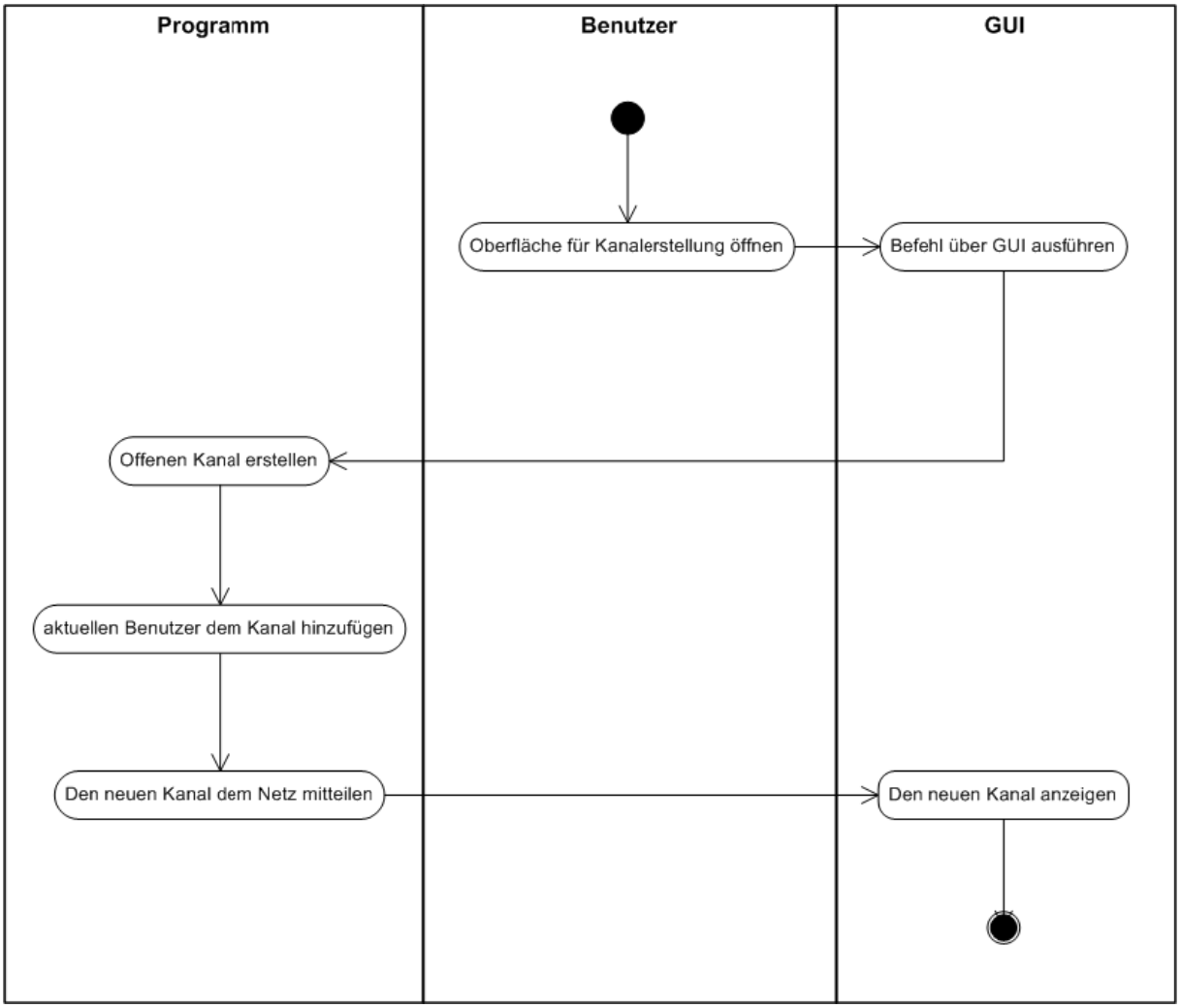

Abbildung 20: Aktivitätsdiagramm zum Erstellen eines offenen Kanals

Das Diagramm /F90/ zeigt was beim Erstellen eines offenen Kanals abläuft. Nachdem der Benutzer über die GUI den Wunsch mitteilt, einen offenen Kanal zu erstellen, wird innerhalb des Programms auch ein Kanal erstellt. Der Ersteller wird automatisch dem Kanal hinzugefügt und anschließend im Netz verbreitet, damit die anderen Knoten wissen, dass ein neuer Kanal erstellt worden ist. Letztendlich wird durch ein neues aktives Kanalfenster und Reiter der Kanal dem Benutzer angezeigt.

2.9.2 Feinanalyse

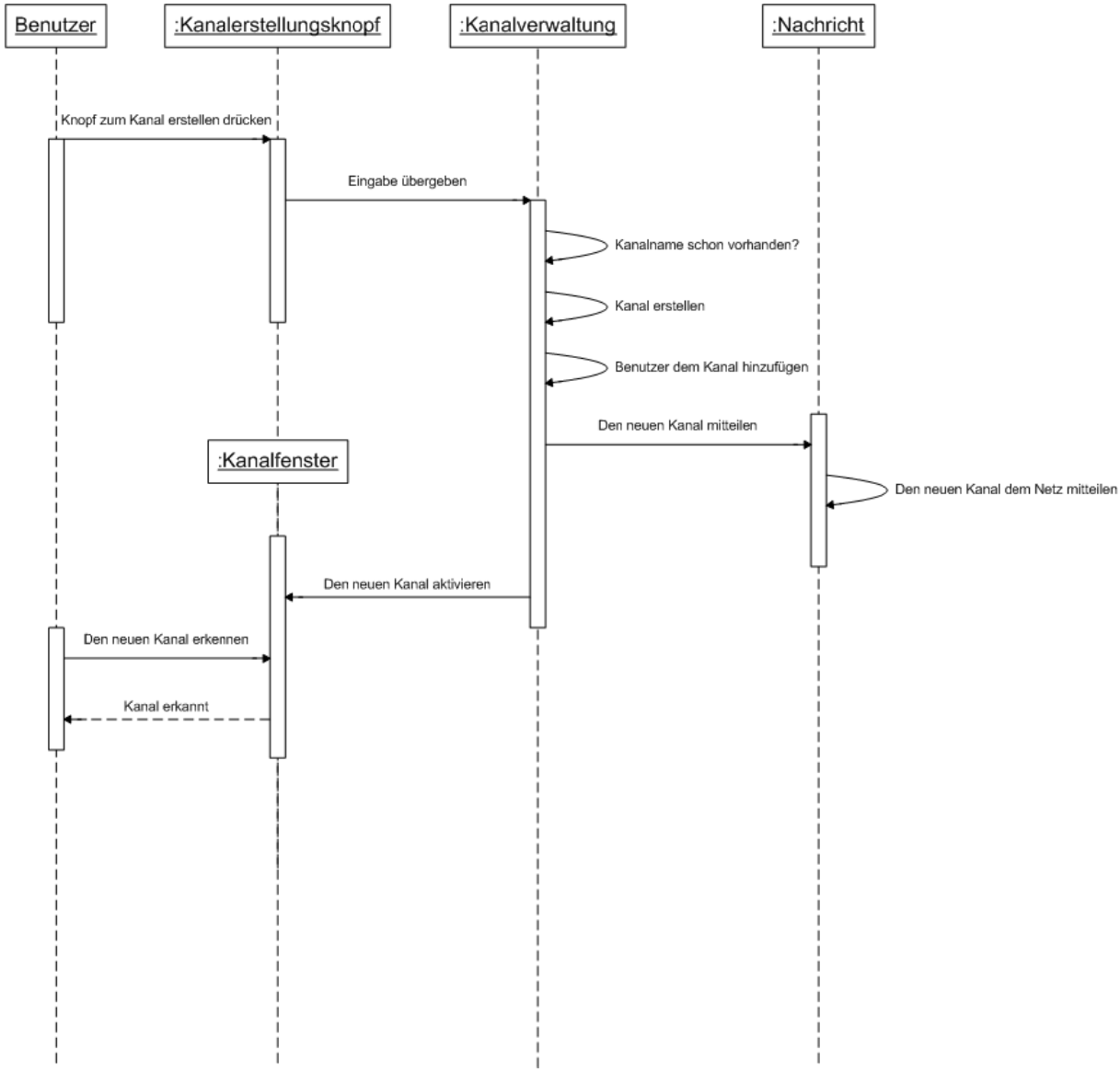

Abbildung 21: Sequenzdiagramm zum Erstellen eines offenen Kanals

Das Sequenzdiagramm veranschaulicht die Interaktion der Objekte beim Erstellen eines offenen Kanals. Nachdem der Benutzer eine Aufforderung zum Kanal erstellen aufgegeben hat, wird geprüft ob der angegebene Name schon vorhanden ist. Falls ja wird es dem Benutzer angezeigt. Wenn dies nicht der Fall ist, wird der Kanal durch die Kanalverwaltung erstellt und der Benutzer automatisch hinzugefügt. Der neue Kanal wird dann mit Hilfe des Nachrichtenobjektes durch eine Channel-Nachricht im Netz verbreitet. Außerdem wird ein neues Kanalfenster erstellt und aktiv geschaltet, damit es der Benutzer erkennt.

2.10 Analyse von Funktionalität /F95/ : Kanäle anzeigen

Die Funktion ist für das Anzeigen der Kanäle zuständig.

2.10.1 Grobanalyse

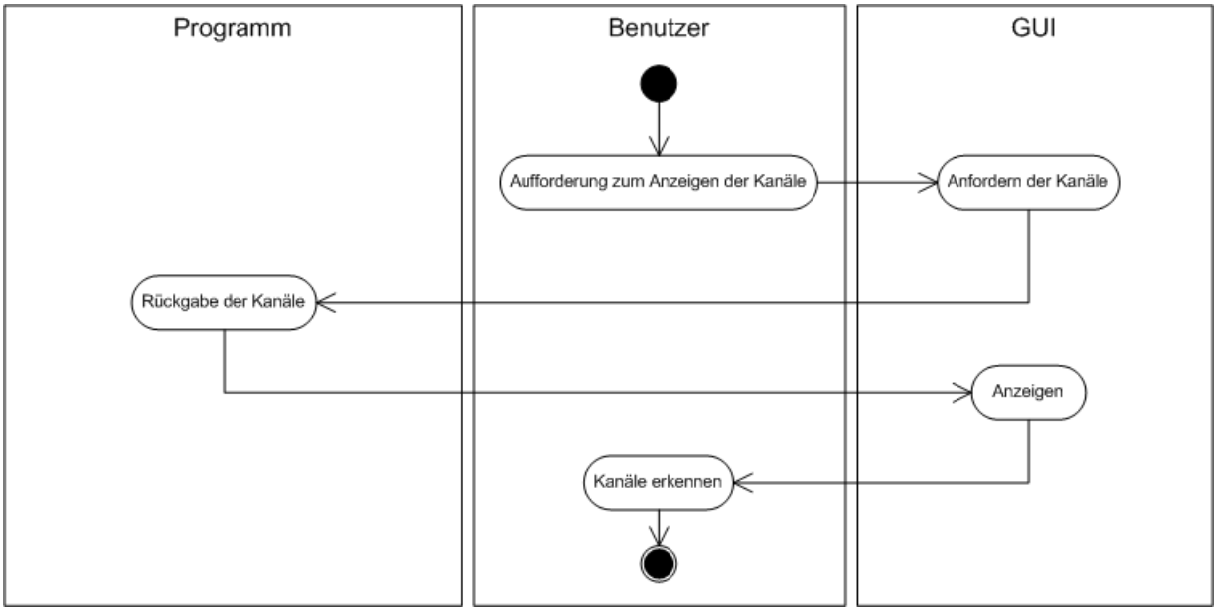

Abbildung 22: Aktivitätsdiagramm zum Anzeigen der Kanäle

Das Diagramm zeigt den Ablauf beim Anzeigen der Kanäle. Nach der Aufforderung des Benutzers über die GUI die Kanäle anzuzeigen, werden die aktuellen Kanäle von Programm an die GUI übergeben und dann angezeigt.

2.10.2 Feinanalyse

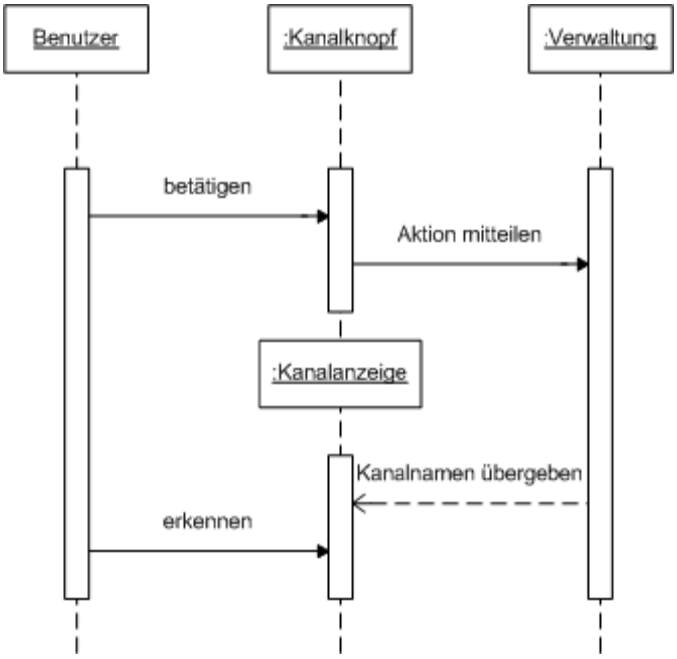

Abbildung 23: Sequenzdiagramm zum Anzeigen der Kanäle

Das Diagramm zeigt die Interaktion der Objekte beim Anzeigen von Kanälen. Nachdem der Benutzer über einen Knopf oder Kommando die Kanäle angezeigt bekommen möchte, werden die aktuellen Kanäle in der Verwaltung abgefragt und zurückgegeben. Sie werden dann in einer Liste angezeigt.

2.11 Analyse von Funktionalität /F100/ : Kanal verlassen

Die Funktion ist für das Verlassen eines Kanals zuständig.

2.11.1 Grobanalyse

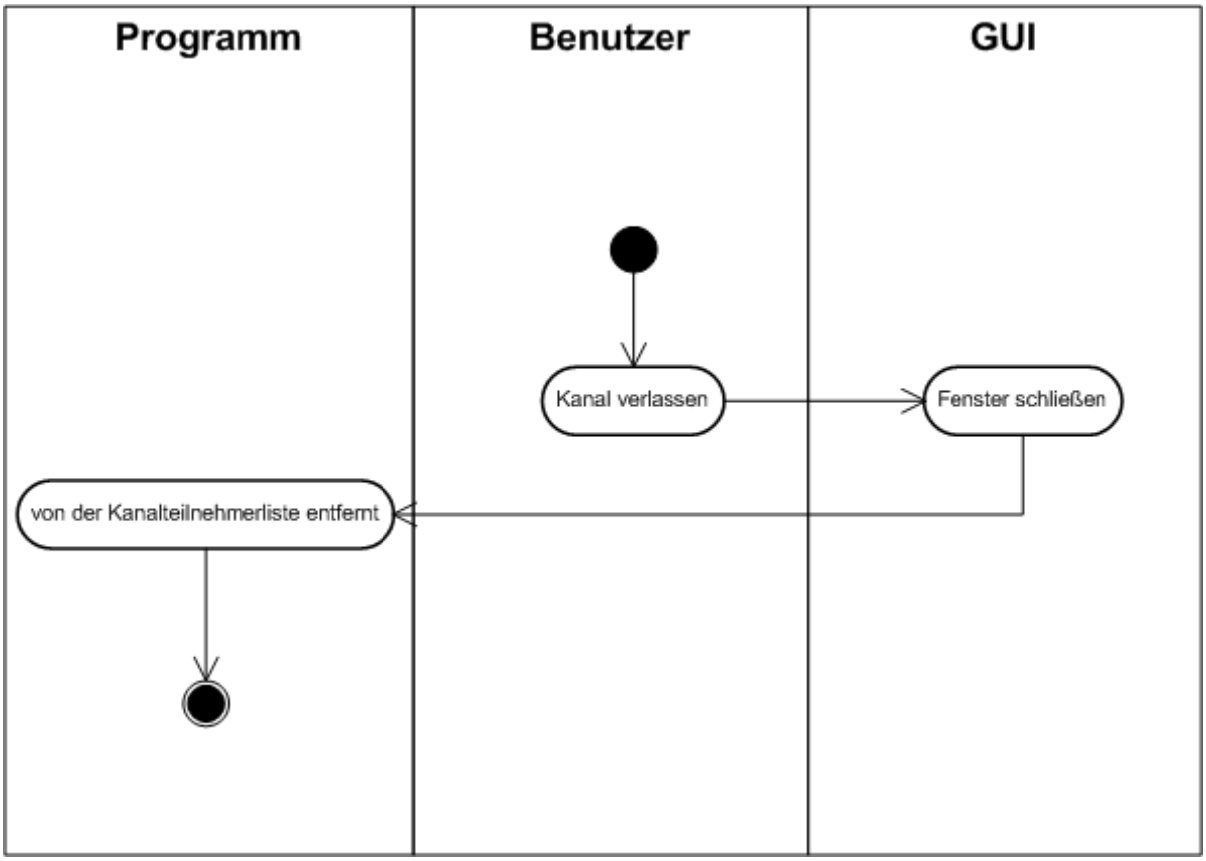

Abbildung 24: Aktivitätsdiagramm zum Verlassen eines Kanals

Das Aktivitätsdiagramm zeigt, wie die Benutzer einen Kanal verlassen können. Der Benutzer verlässt den Kanal, wobei das Fenster geschlossen wird. Das Programm entfernt den Teilnehmer aus der Kanalteilnehmerliste und er verlässt so den Kanal.

2.11.2 Feinanalyse

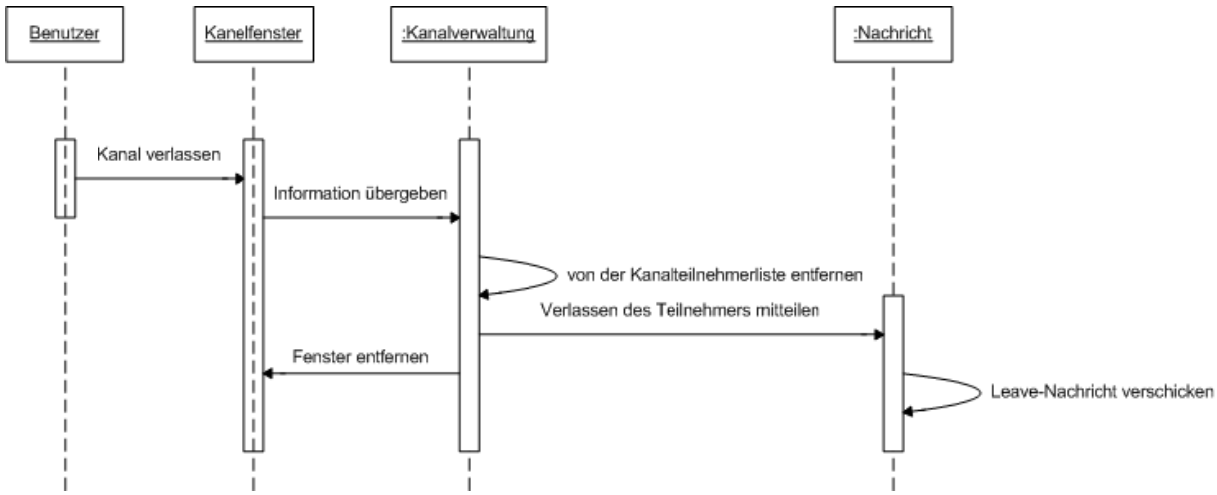

Abbildung 25: Sequenzdiagramm zum Verlassen eines Kanals

Das Sequenzdiagramm zeigt, wie die Objekte miteinander kommunizieren, damit ein Benutzer einen Kanal verlässt. Der Benutzer teilt mit, dass er den Kanal verlassen möchte, indem er das Kanalfenster schließt. Dies wird der Kanalverwaltung mitgeteilt und sie entfernt den Teilnehmer von der Kanalteilnehmerliste. Anschließend wird dies durch eine Leave-Nachricht an die anderen Benutzer im Netz propagiert. Letztendlich wird das Kanalfenster entfernt.

2.12 Analyse von Funktionalität /F110/ : Geschlossenen Kanal beitreten

Die Funktion ist für das Beitreten in einen geschlossenen Kanal zuständig.

2.12.1 Grobanalyse

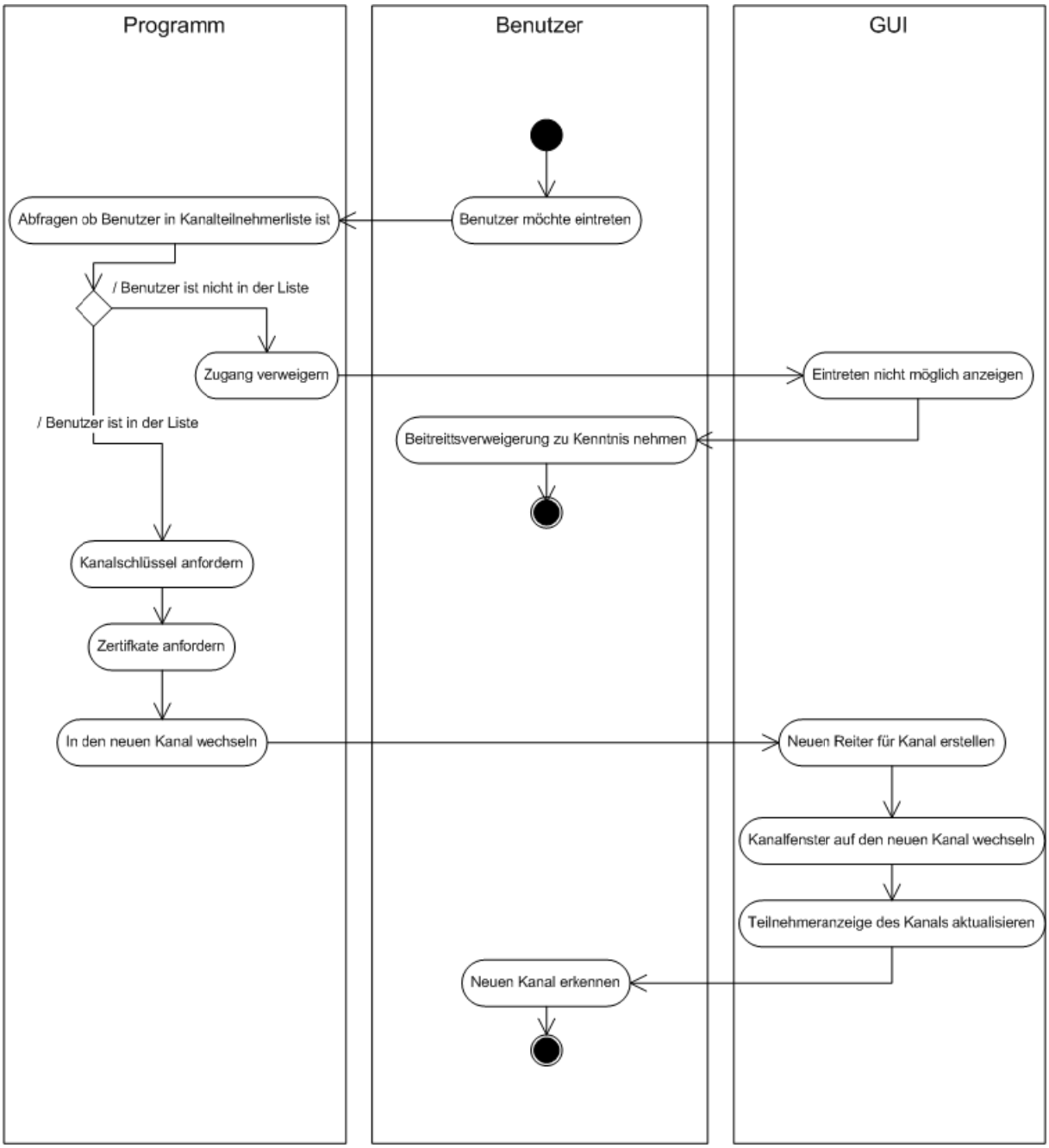

Abbildung 26: Aktivitätsdiagramm zum Beitreten in einen geschlossenen Kanal

Das Diagramm zeigt schrittweise, was beim Beitreten in einen geschlossen Kanal passiert. Nach dem Wunsch des Benutzers einen Kanal beitreten zu möchten, prüft das Programm, ob der Benutzer vorher durch jemanden, der bereits Teilnehmer dieses Kanal ist, eingeladen und somit der Kanalteilnehmer hinzugefügt worden ist. Falls das nicht der Fall ist, wird dem Benutzer der Zutritt verweigert und ihm dies in geeigneter Art und Weise angezeigt. Anderenfalls wird ihm der Zutritt gewährt und das Programm fordert den gemeinsamen Kanalschlüssel und die Zertifikate der Teilnehmer in diesem Kanal an. Der Kanal wird anschließend in der GUI erstellt, indem ein neues Kanalfenster und ein dazugehöriger Reiter hinzugefügt wird und diese aktiv geschaltet werden.

2.12.2 Feinanalyse

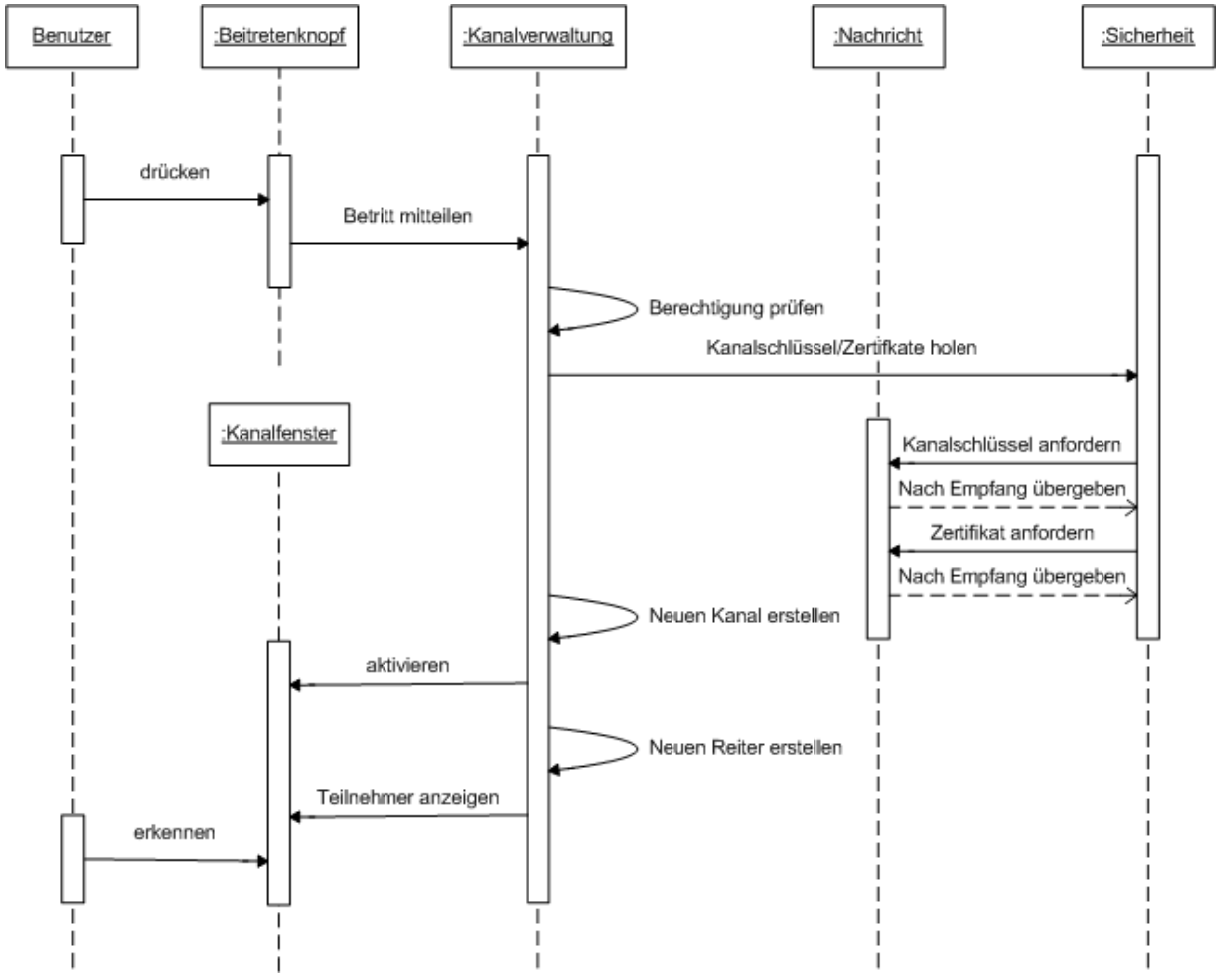

Abbildung 27: Sequenzdiagramm zum Beitreten in einen geschlossenen Kanal

Das Diagramm zeigt die Interaktion von Objekten beim erfolgreichen Beitreten eines geschlossenen Kanals. Über einen Knopf oder ein Kommando gibt der Benutzer an, dass er einen bestimmten Kanal beitreten möchte. Anschließend wird dies der Kanalverwaltung mitgeteilt, die prüft, ob der Benutzer beitreten darf. Nachdem dem Sicherheitsobjekt eine Anforderung des gemeinsamen Schlüssels und der nötigen Zertifikate übergeben worden ist, führt es dies mit Hilfe des Nachrichten Objektes durch, damit diese im Sicherheitsobjekt bei einem späteren Empfang und Senden von Nachrichten vorliegen. Das Kanalverwaltungsobjekt übergibt dann noch die Aufforderung ein neues Kanalfenster zu erstellen, aktiv zu schalten und die derzeitigen Teilnehmer anzuzeigen.

2.13 Analyse von Funktionalität /F120/ : Offenen Kanal beitreten

Die Funktion ist für das Beitreten eines offenen Kanals zuständig.

2.13.1 Grobanalyse

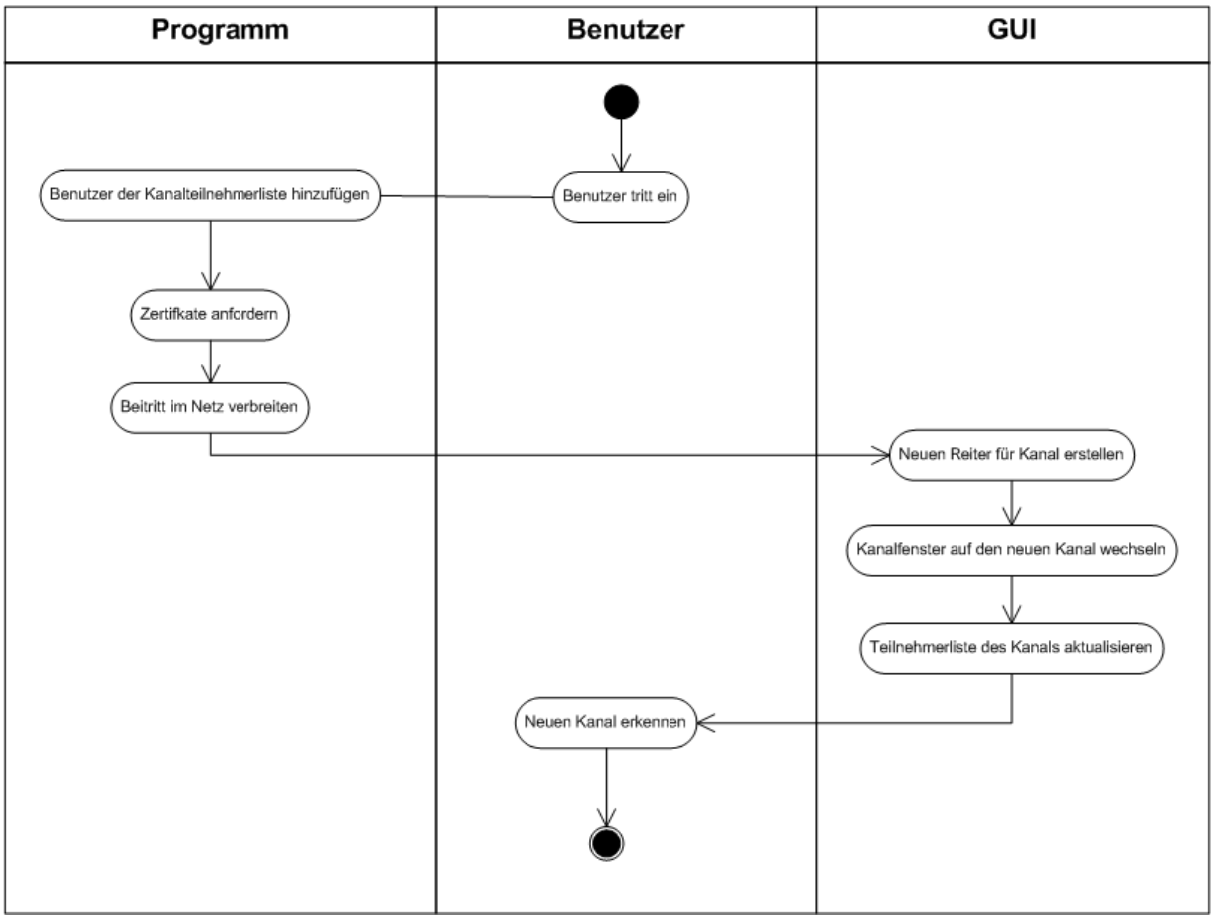

Abbildung 28: Aktivitätsdiagramm zum Beitreten in einen offenen Kanal

Durch das Aktivitätsdiagramm der /F120/ wird beschrieben, wie ein Benutzer einem offenen Kanal beitreten kann. Der Benutzer tritt in den Kanal ein und er wird der Kanalteilnehmerliste hinzugefügt. Dann werden die Zertifikate der anderen Benutzer angefordert und der Kanalbeitritt den im Netz mitgeteilt. Letztendlich wird ein neues Kanalfenster mit zugehörigem Reiter erstellt, welche dann der Benutzer zur Kenntnis nimmt.

2.13.2 Feinanalyse

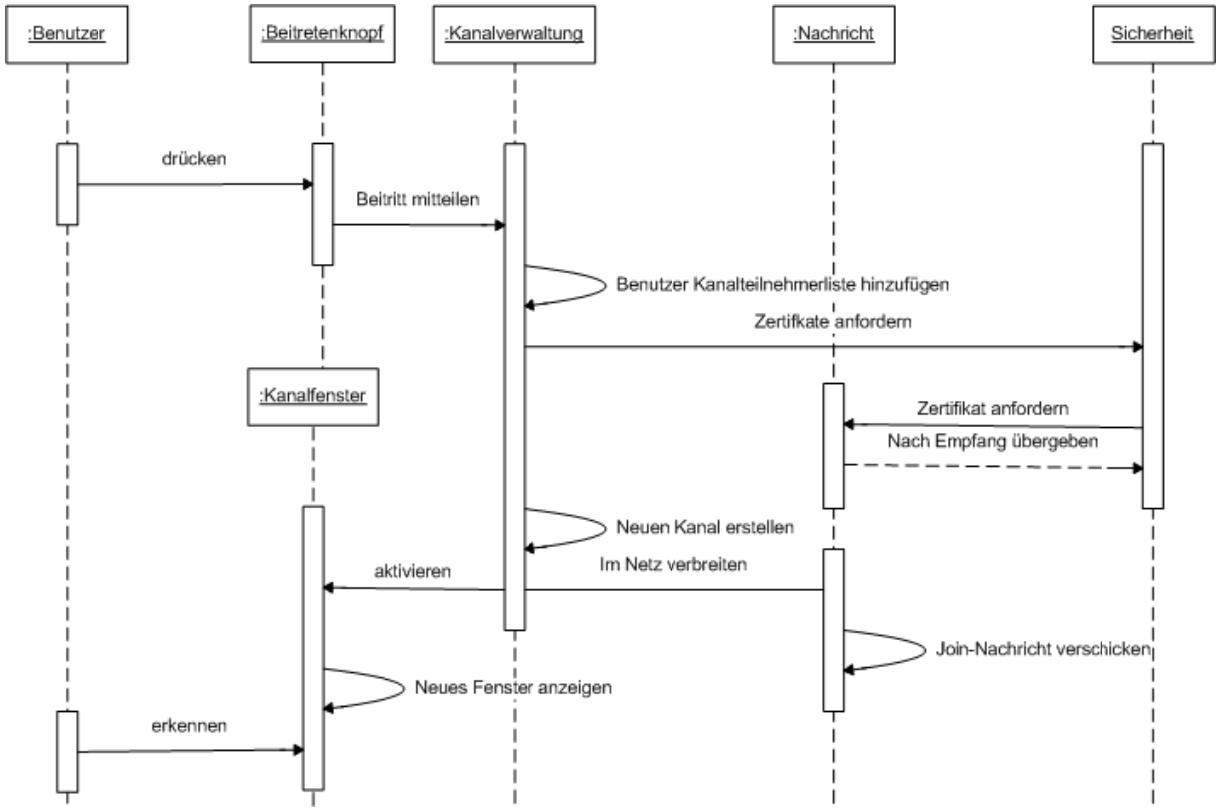

Abbildung 29: Sequenzdiagramm zum Beitreten in einen offenen Kanal

Durch das Sequenzdiagramm der /F120/ wird beschrieben, wie die Objekte beim Beitreten eines Benutzers in einen offenen Kanal kommunizieren. Durch einen Knopf oder ein Kommando möchte der Benutzer einem offenen Kanal beitreten. Dies wird der Kanalverwaltung mitgeteilt und sie fügt den Benutzer der Kanalteilnehmerliste hinzu. Anschließen fordert sie das Sicherheitsobjekt auf die Zertifikate der anderen Benutzer in diesem Kanal anzufordern, was es mit Hilfe des Nachrichtenobjekts durchführt. Außerdem gibt die Kanalverwaltung dem Kanalfenster den Auftrag einen neuen Kanal mit zugehörigen Reiter zu erstellen, damit dies vom Benutzer erkannt wird. Ebenfalls wird eine Join-Nachricht verschickt, um im Netz mitzuteilen das der Benutzer beigetreten ist.

2.14 Analyse von Funktionalität /F130/ : Jemand anderen in einen geschlossenen Kanal einladen

Die Funktion ist dafür zuständig, damit man andere in einen geschlossenen Kanal einladen kann.

2.14.1 Grobanalyse

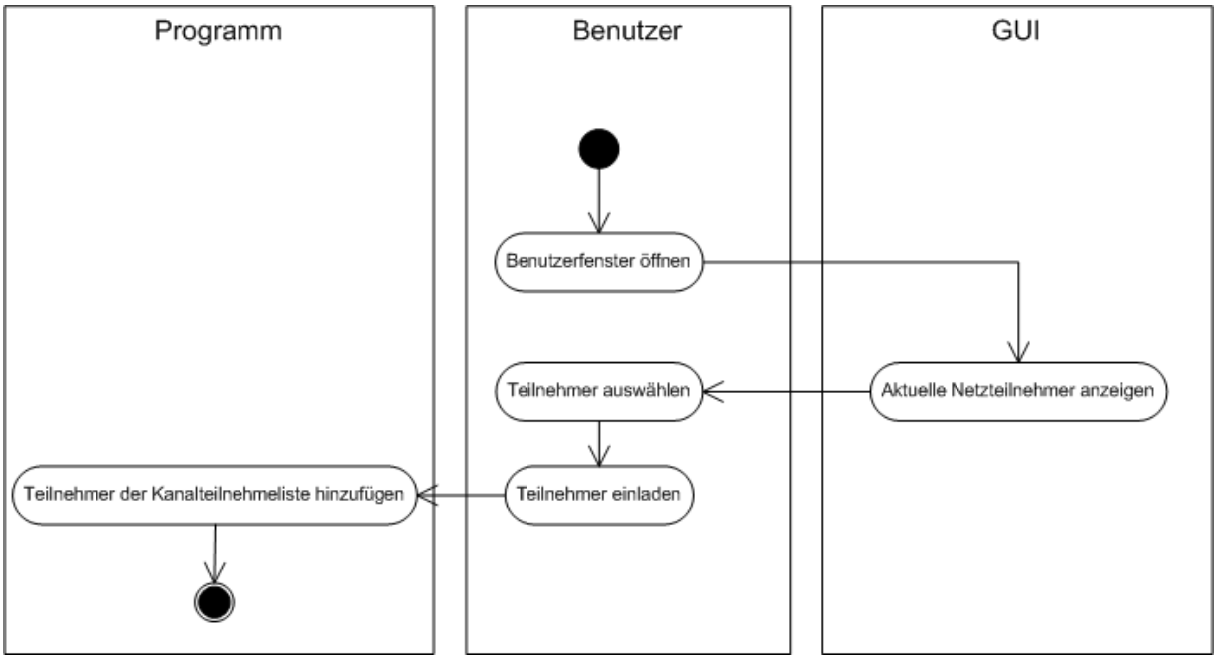

Abbildung 30: Aktivitätsdiagramm zum Einladen in einen geschlossenen Kanal

Das Diagramm beschreibt den Ablauf für das Einladen eines anderen Benutzers in einen geschlossenen Kanal. Der Benutzer ruft ein Fenster auf, in dem alle aktiven Teilnehmer angezeigt werden. Darüber kann er einen Teilnehmer auswählen und ihn für einen geschlossenen Kanal freischalten. Im Programm wird der Teilnehmer der Kanalteilnehmerliste hinzugefügt und hat so die Berechtigung für den Kanal.

2.14.2 Feinanalyse

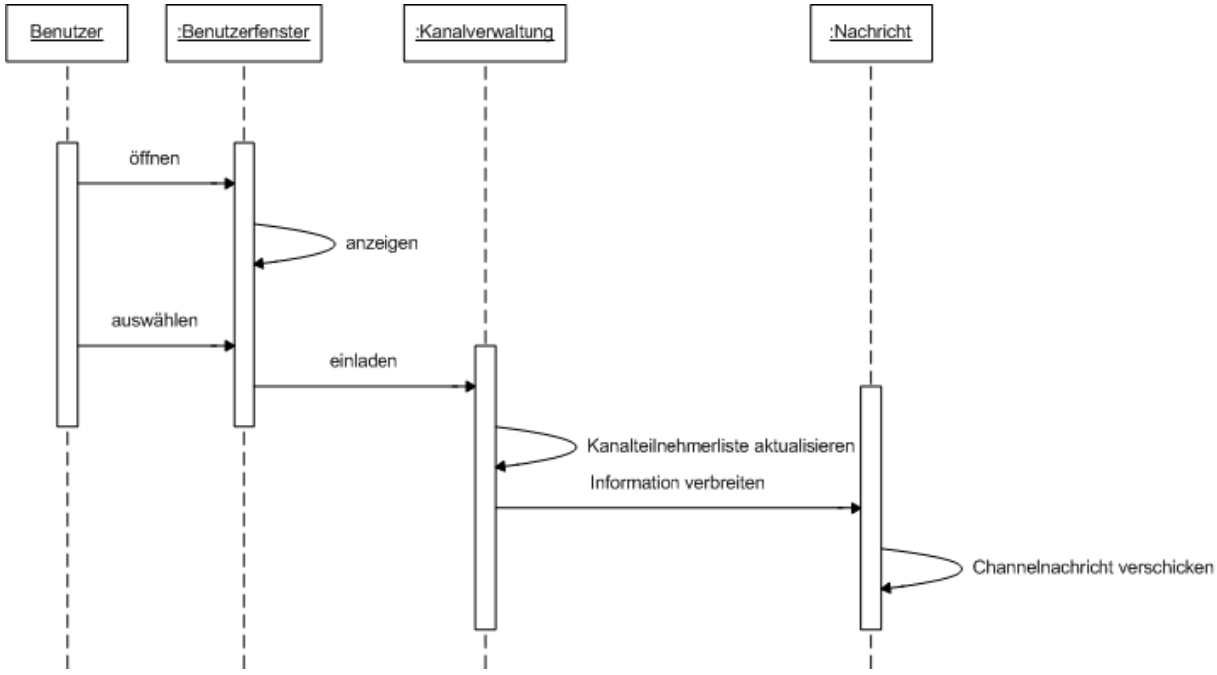

Abbildung 31: Sequenzdiagramm zum Einladen in einen geschlossenen Kanal

Das Diagramm zeigt die Interaktion der Objekte, um einen Benutzer in einen geschlossenen Kanal einladen zu können. Der Benutzer öffnet zunächst ein Benutzerfenster und wählt dann Benutzer, den er einladen möchte, aus. In der Kanalverwaltung wird dann die Kanalteilnehmerliste aktualisiert und letztendlich die Information über das Nachrichtenobjekt mit Hilfe einer Channel-Nachricht verteilt. Dadurch das der Benutzer in der Channel-Nachricht hinzugefügt ist, ist es ihm nun möglich in den Kanal beizutreten.

2.15 Analyse von Funktionalität /F140/ /150/ : Zertifikat anfordern, versenden

Die Funktionen sind dafür da, um Zertifikate anfordern und versenden zu können.

2.15.1 Grobanalyse

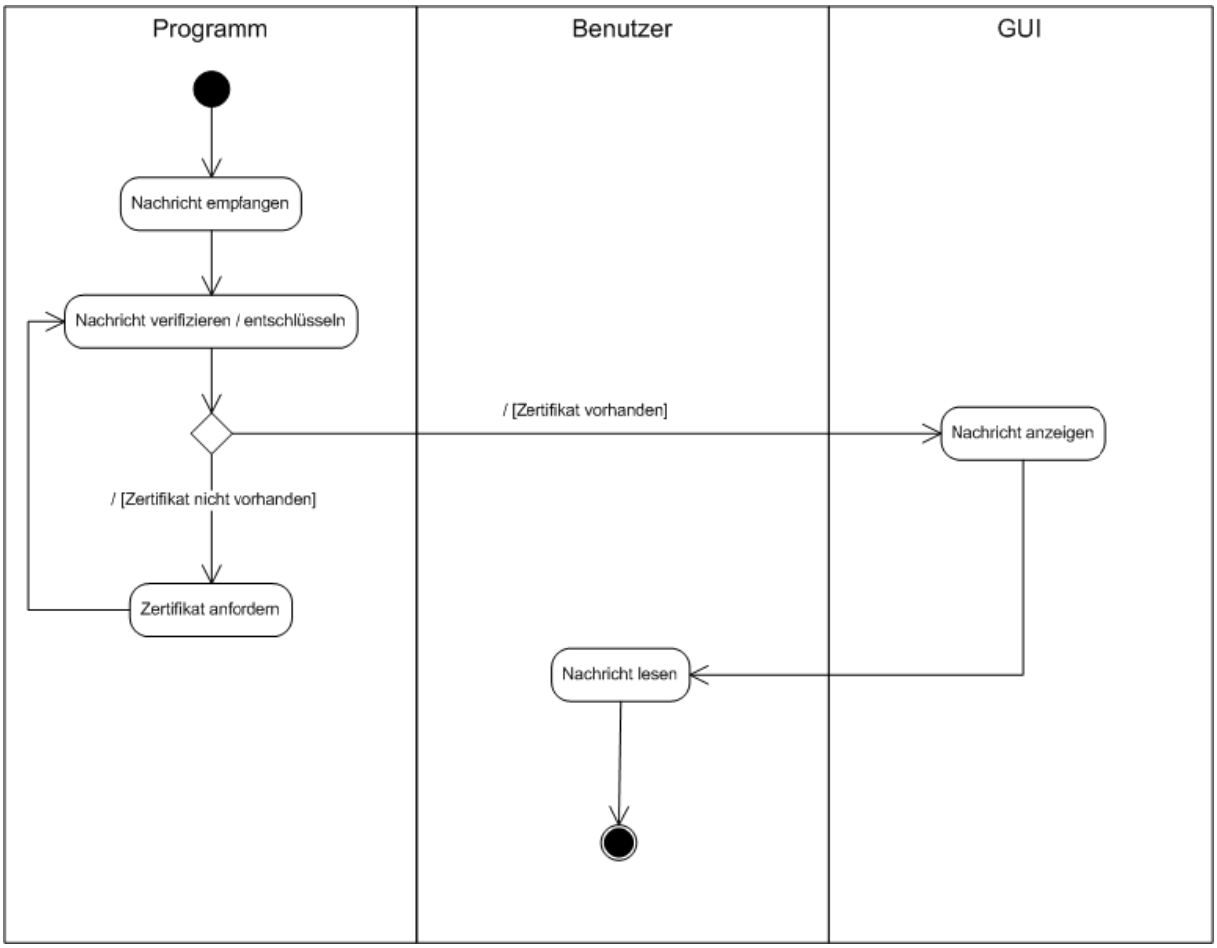

Abbildung 32: Aktivitätsdiagramm zum Anfordern eines Zertifikats

Das Diagramm beschreibt das Anfordern eines Zertifikats, anhand des Beispiels, dass eine signierte Nachricht in einem öffentlichen Kanal eintrifft. Diese muss verifiziert werden, ist hierfür das Zertifikat des Senders vorhanden, wird die Nachricht verifiziert und dem Benutzer zum Lesen im Kanalfenster angezeigt, andernfalls muss das Zertifikat vorher vom Sender angefordert werden. Das Anfordern des Zertifikats ruft beim Sender der Nachricht die Funktion /F150/ Zertifikat senden auf und das Zertifikat wird automatisch versendet. Ein anderes, hier nicht aufgeführtes Beispiel, wäre ein Zertifikat anfordern um damit eine Nachricht im Anonymen Kanal zu verschlüsseln.

2.15.2 Feinanalyse

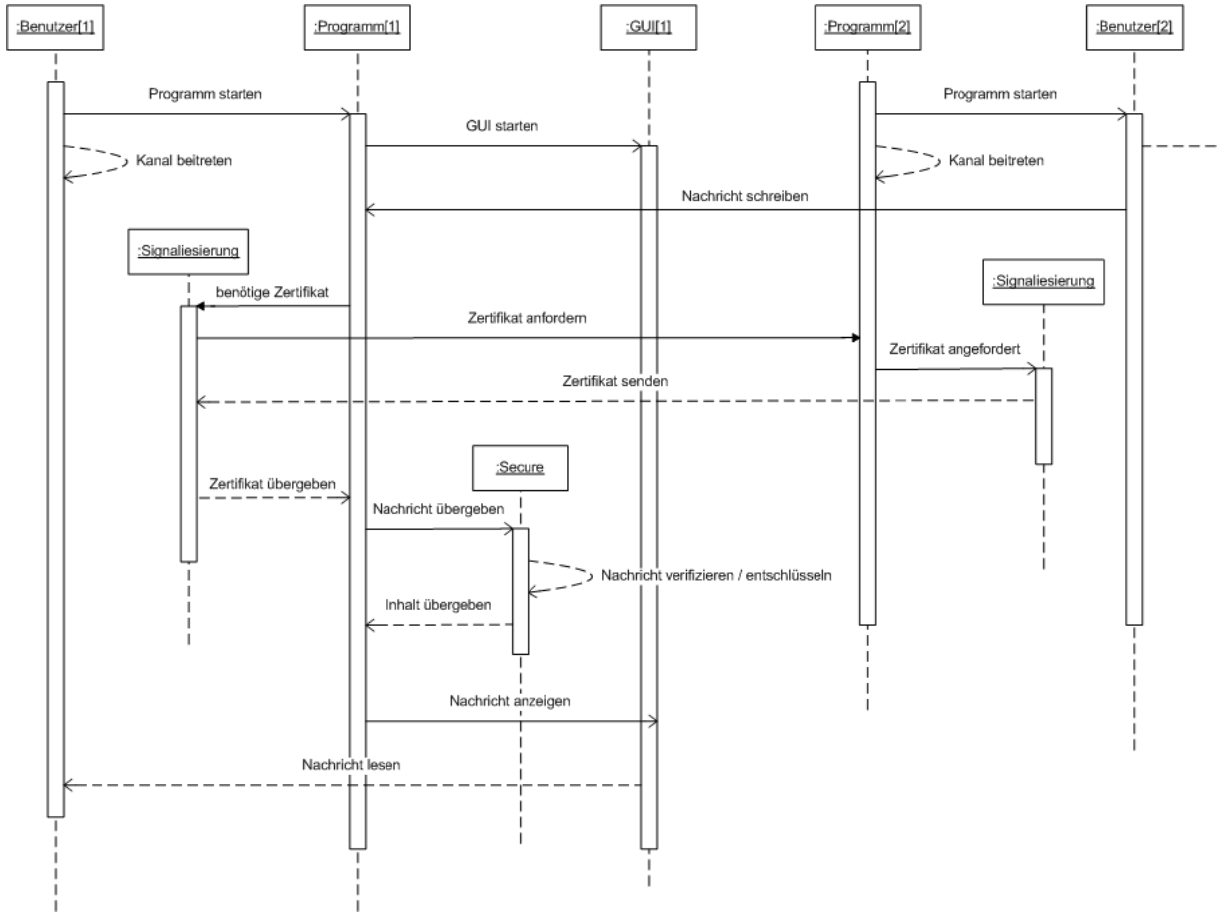

Abbildung 33: Sequenzdiagramm zum Anfordern und versenden eines Zertifikats

Das Diagramm beschreibt die Interaktionen beim Anfordern und Senden eines Zertifikates. Textfeld, Kanalfeld, ... wurden hier zum Objekt GUI zusammengefasst und bei Benutzer2, der Übersichthalber, gar nicht mehr aufgeführt. Benutzer1 und Benutzer2 treten dem gleichen Kanal bei. Benutzer2 schreibt eine Nachricht an Benutzer1, diese wird vom Nachrichtobjekt, von Benutzer1, empfangen und weitergegeben an das Secureobjekt, hier soll die Nachricht verifiziert werden. Ist zum verifizieren das Zertifikat von Benutzer2 noch nicht vorhanden, wird es über das Nachrichtobjekt angefordert, in Form einer GETCERTIFICATE Nachricht. Das Nachrichtenobjekt von Benutzer2 bekommt diese GETCERTIFICATE Nachricht und aus diesem Objekt wird das Zertifikat automatisch an Benutzer1 gesendet. Das Nachrichtenobjekt von Benutzer1 empfängt dies und gibt es weiter an Secure wo die Nachricht nun verifiziert werden kann und weiter behandelt wird wie bereits beschrieben.

2.16 Analyse von Funktionalität /F160/ /170: Gemeinsamen Schlüssel für geschlossenen Kanal anfordern senden

Die Funktionen sind dafür zuständig Zertifikate anzufordern und zu versenden.

2.16.1 Grobanalyse

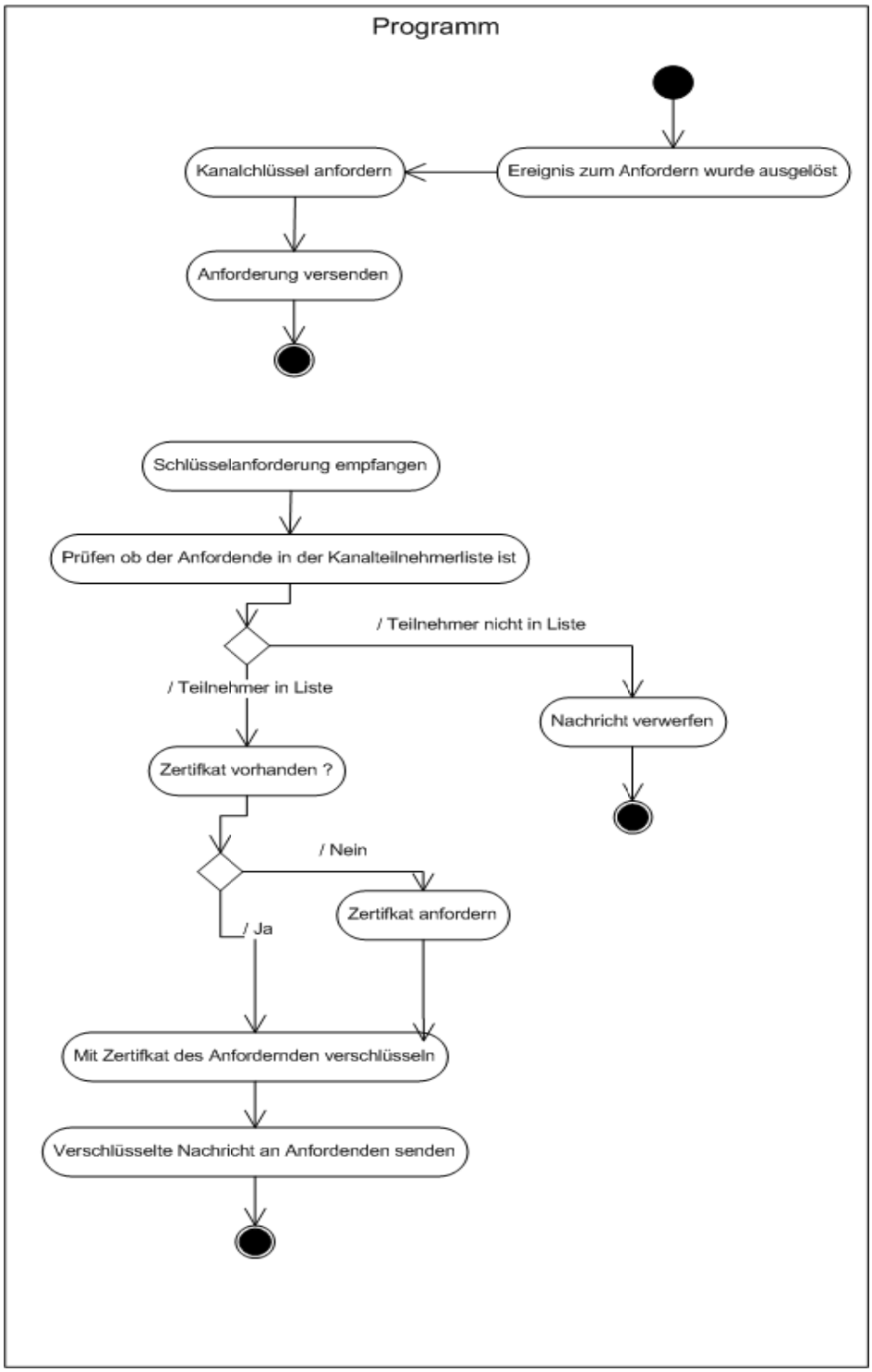

Abbildung 34: Aktivitätsdiagramme zum Anfordern und Senden des Kanalschlüssels

Die Diagramme zeigen die Schritte, die beim Anfordern und Senden des gemeinsamen Schlüssels, durchgeführt werden. Eine Anforderung wird gesendet sobald ein auslösendes Ereignis, wie z.B. das Beitreten eines Kanals oder die Nichtverfügbarkeit des Schlüssels, aufgetreten ist. Das Versenden des Schlüssels selbst erfolgt nur direkt nach einer Anforderung. Es wird zuerst geprüft, ob der Anfordernde den Schlüssel erhalten darf. Um den Schlüssel auch verschlüsselt senden zu können, benötigt man das Zertifikat des Anfordernden, welches angefordert werden muss, falls es nicht vorhanden ist. Nach dem verschlüsseln wird es letztendlich versendet.

2.16.2 Feinanalyse

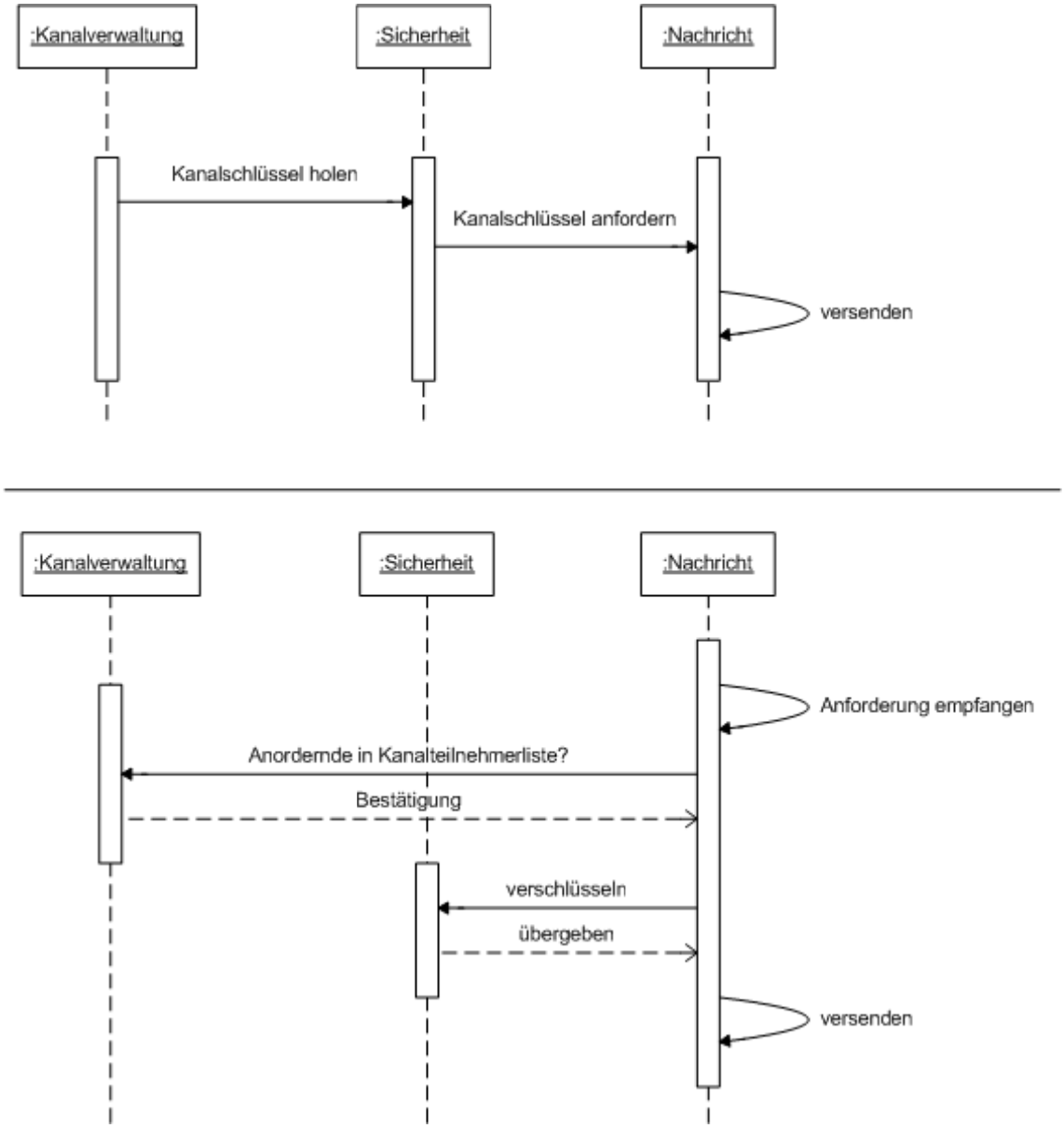

Abbildung 35: Sequenzdiagramme zum Anfordern und Senden des Kanalschlüssels

Die Diagramme zeigen die Interaktion von Objekten zum Anfordern und zum Versenden eines gemeinsamen Schlüssels. Im ersten Diagramm wird die Anforderung durch die Kanalverwaltung ausgelöst, indem z.B. ein Benutzer einen Kanal beigetreten ist. Das Sicherheitsobjekt fordert dann den Schlüssel über das Nachrichtenobjekt an.
Im zweiten Diagramm wird eine Anforderung empfangen und dementsprechend reagiert. Es wird über die Kanalverwaltung geprüft, ob der Anfordernde berechtigt für den Schlüssel ist. Danach wird mit Hilfe des Sicherheitsobjekt die Nachricht mit dem Schlüssel verschlüsselt, wobei das Zertifikat des Anfordernden in diesem Diagramm vorliegt. Zuletzt wird dann die Nachricht versendet.

2.17 Analyse von Funktionalität /F175/ : Speichern und späteres erneutes Senden von Nachrichten

Die Funktion ist für das Speichern und spätere erneute Senden von Nachrichten zuständig.

2.17.1 Grobanalyse

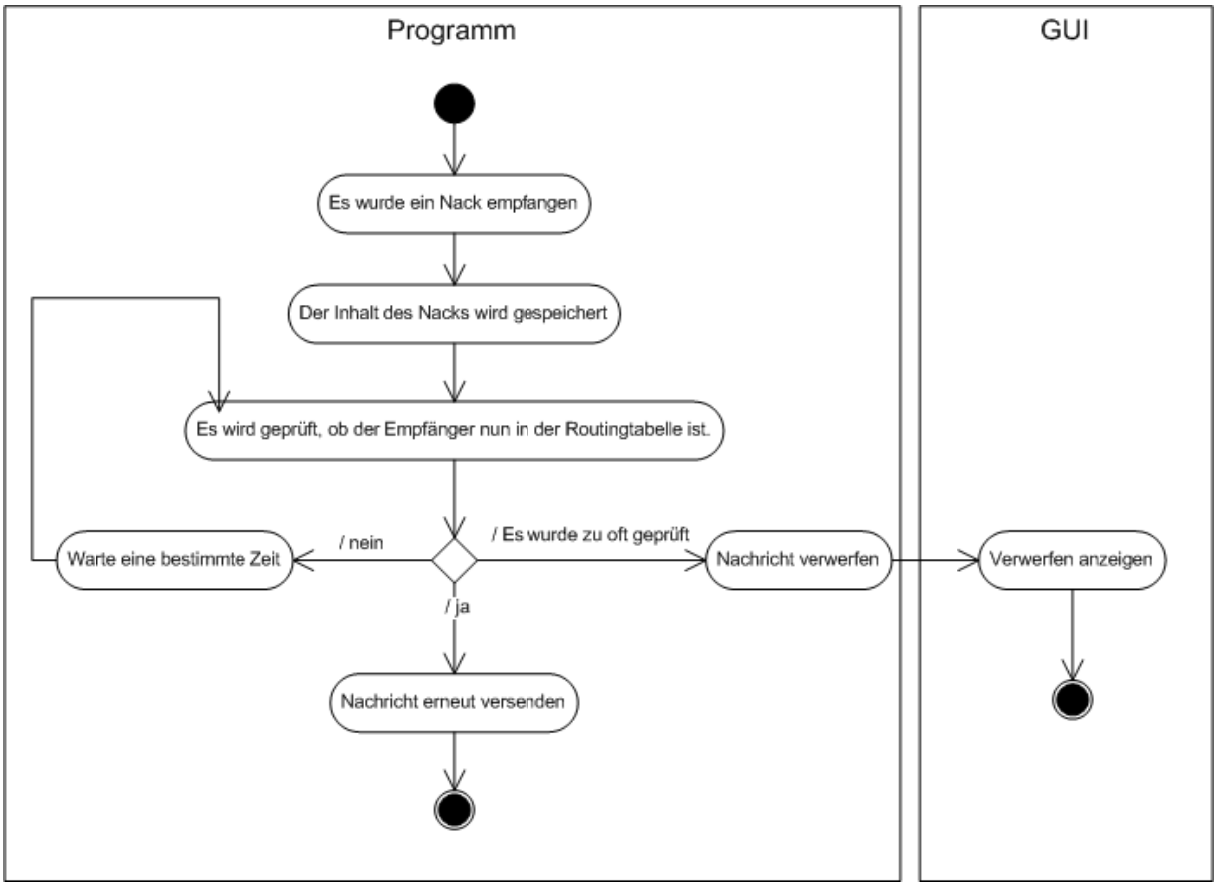

Abbildung 36: Aktivitätsdiagramm zum Speichern und späteren erneuten Senden von Nachrichten

Das Diagramm zeigt den Ablauf, was beim Speichern und späteren erneuten Senden von Nachrichten genau passiert. Zunächst muss eine negative Empfangsbestätigung erhalten worden sein. In dieser befindet sich die gesendete Nachricht die nicht angekommen ist. Über die Routingtabelle wird periodisch geprüft, ob der Empfänger nun erreichbar ist. Sobald er erreichbar ist, wird die Nachricht erneut losgeschickt. Falls er längere Zeit nicht erreichbar ist wird dem Sender angezeigt, dass die Nachricht verworfen wurde.

2.17.2 Feinanalyse

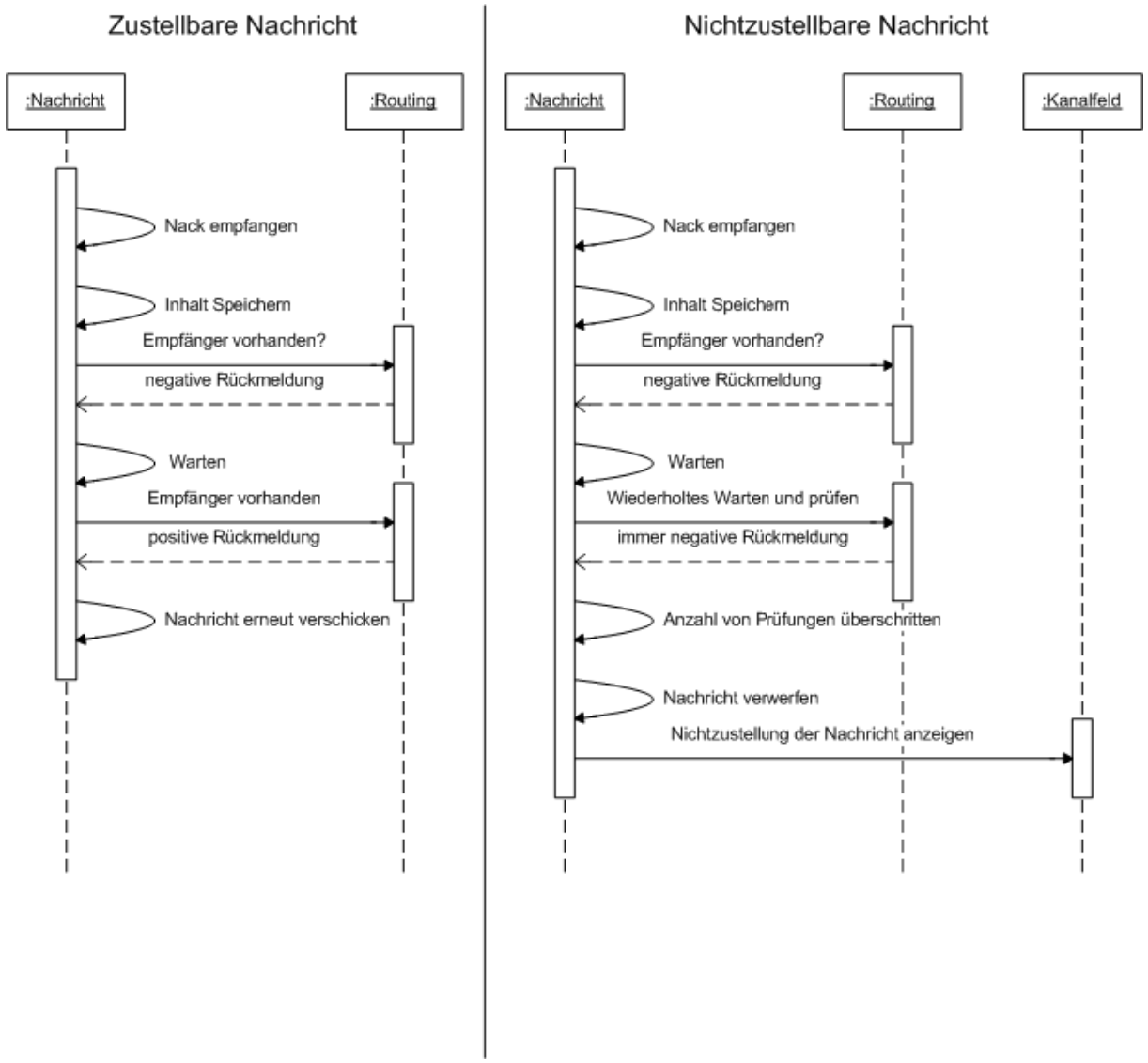

Abbildung 37: Sequenzdiagramm zum Speichern und späteren erneuten Senden von Nachrichten

Die Diagramme zeigen die Interaktion der Objekte beim Speichern und späteren erneuten Senden von Nachrichten. Das erste Diagramm zeigt eine zustellbare Nachricht, die letztendlich den Empfänger erreicht. Nachdem dort ein Nack empfangen worden ist wird, der Inhalt gespeichert und dann mit Hilfe des Routingobjektes geprüft, ob der Empfänger vorhanden ist. Weil der Empfänger zunächst nicht in der Routingtabelle ist wird gewartet. Bei einer späteren neuen Anfrage an das Routingobjekt, ist dieser nun im Netz und die Nachricht wird erneut losgeschickt. Im zweiten Diagramm erhält das Programm immer wieder negative Rückmeldungen, so dass die Nachricht verworfen und das dem Benutzer angezeigt wird.

2.18 Analyse von Funktionalität /F180/ : Aktualisieren der Netzstruktur

Die Funktion ist für die Aktualisierung der Netzstruktur zuständig.

2.18.1 Grobanalyse

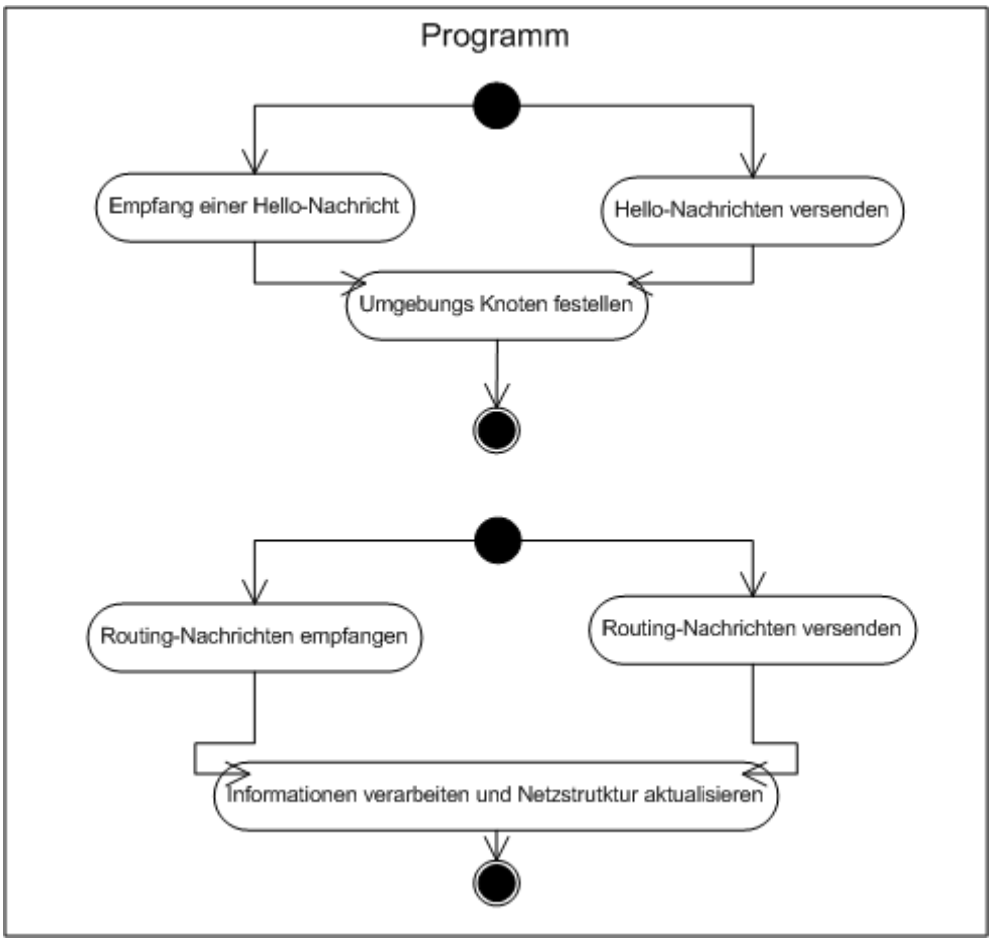

Abbildung 38: Aktivitätsdiagramm zur Aktualisierung der Netzstrukturen

Das Diagramm zeigt wie die Netzstruktur aktualisiert wird. Zunächst werden Hello-Nachrichten versendet. Durch den Empfang weiß das Programm, welche anderen Knoten in seiner Umgebung sind. An diese schickt man dann jeweils Routingnachrichten, um denen mitzuteilen, welche man selbst erreichen kann. Durch den Empfang von diesen erhält man dann Informationen, um die Netzstruktur aufzubauen und zu aktualisieren.

2.18.2 Feinanalyse

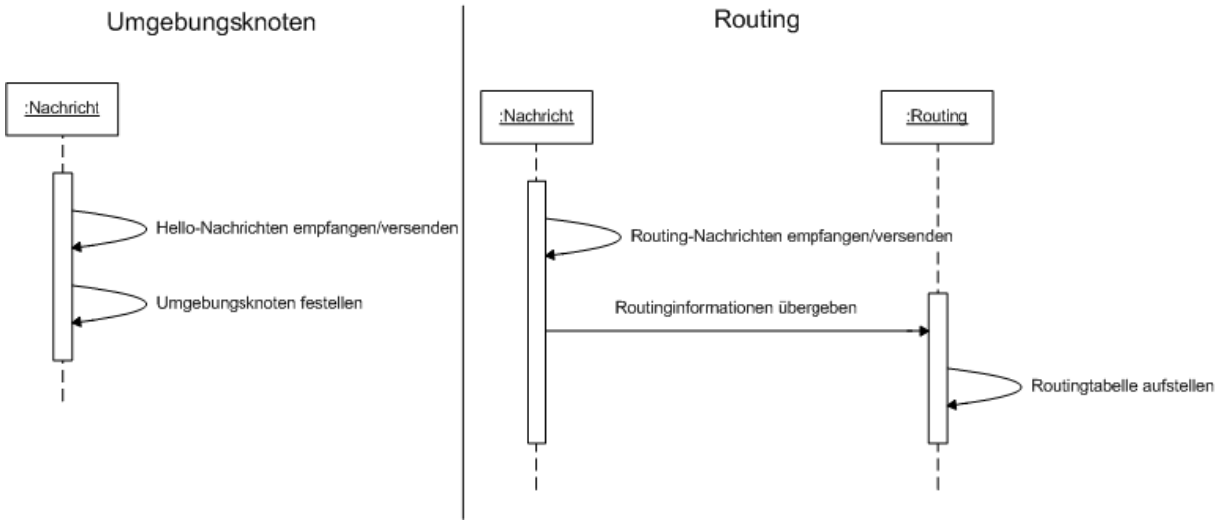

Abbildung 39: Sequenzdiagramm zur Aktualisierung der Netzstruktur

Die Diagramme zeigen die Interaktion der Objekte beim Aktualisieren der Netzstruktur. Die Hello-Nachrichten werden von dem Nachrichtenobjekt versendet und empfangen. Durch den Empfang der Hello-Nachrichten weiß man, welche anderen Benutzer in der Umgebung sind. Ebenfalls versendet und empfängt man Routing-Nachrichten. Nach dem Empfang einer Routing-Nachricht werden deren Informationen an das Routingobjekt weitergeleitet. Dadurch hat es genug Informationen um eine Routingtabelle aufzubauen. In den versendeten Routing-Nachrichten stehen Informationen darüber, welche Knoten man mit wie viel Hops erreicht.

2.19 Analyse von Funktionalität /F190/ : In den Infrastrukturmodus wechseln

Die Funktion ist dafür da, um in den Infrastrukturmodus zu wechseln.

2.19.1 Grobanalyse

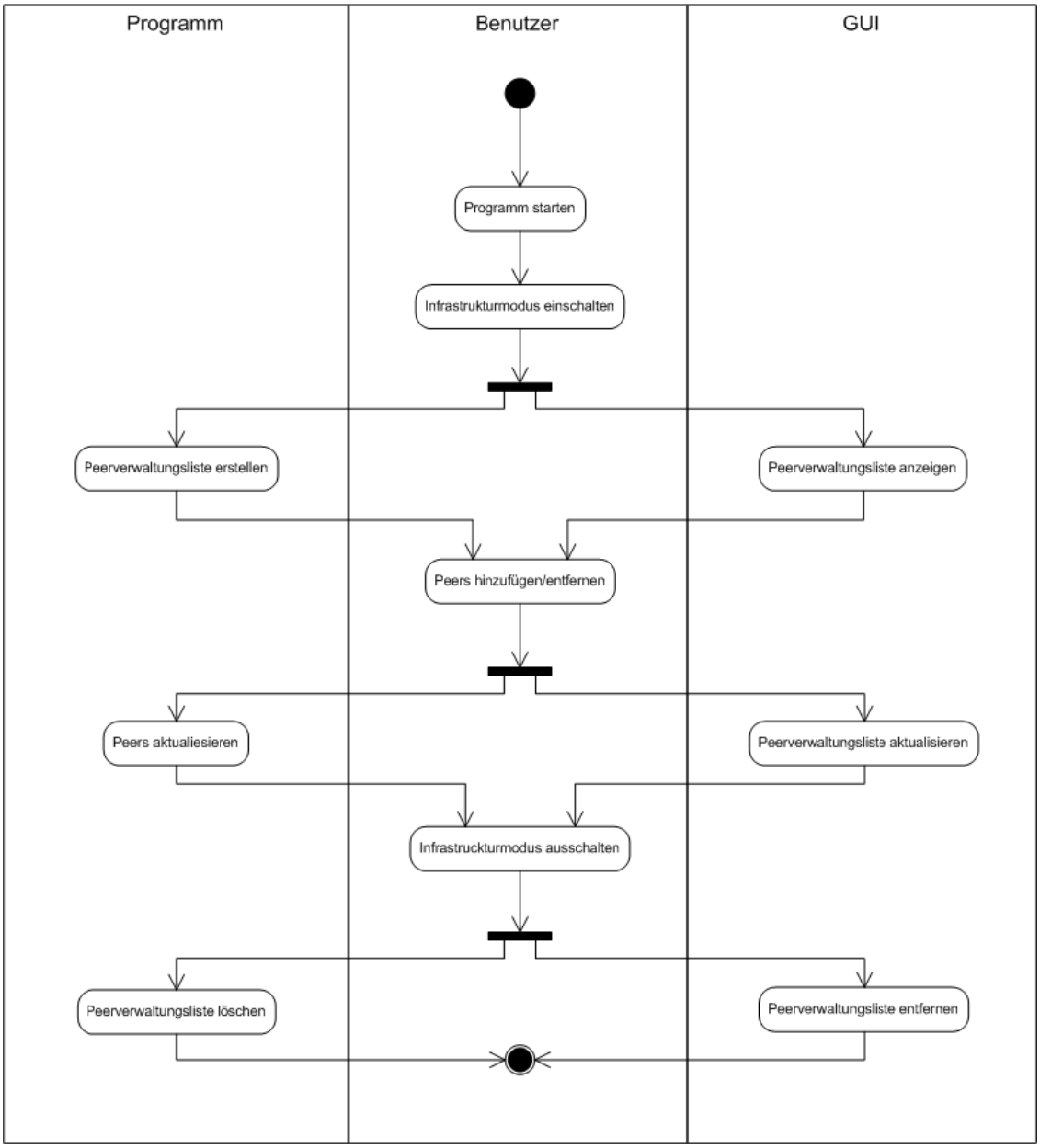

Abbildung 40: Aktivitätsdiagramm zum Wechseln in den Infrastrukturmodus

Das Diagramm zeigt den Ablauf beim Wechseln in den Infrastruktur Modus sowie den die Aktionen beim verlassen. Wurde die Peerverwaltungsliste erstellt, hat der Benutzer die Möglichkeit soviele Peers hinzuzufügen oder zu löschen wie er möchte. In diesem Modus kann der Benutzer alle Funktionen benutzen, die ihm auch im normalen Modus zur Verfügung stehen.

2.19.2 Feinanalyse

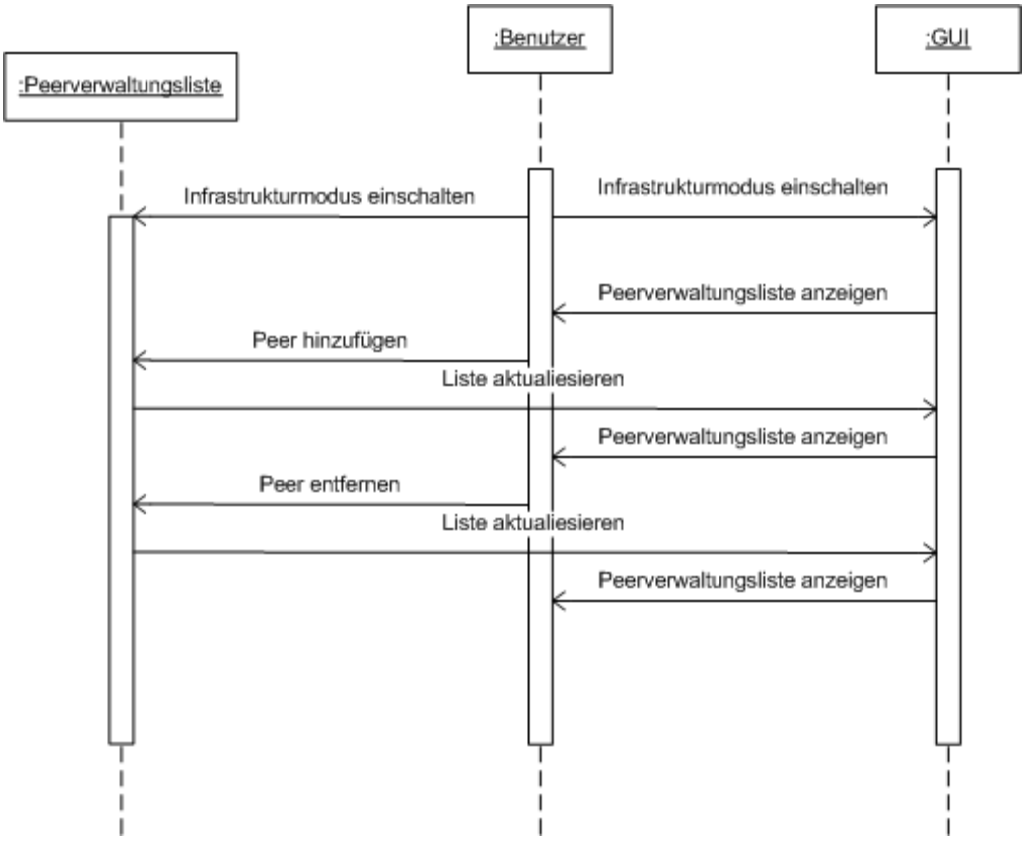

Abbildung 41: Sequenzdiagramm zum Wechseln in den Infrastrukturmodus

Das Diagramm zeigt wie der Benutzer in den Infrastrukturmodus wechselt, die Peerverwaltungsliste wird erstellt und dem Benutzer angezeigt. Der Benutzer fügt einen Peer hinzu und löscht diesen wieder, während der ganzen Zeit wird die Liste aktualisiert und dem Benutzer angezeigt

2.20 Analyse von Funktionalität /F200/ : Partitionierung und Verschmelzung von zwei Netzen

Die Funktion ist dafür zuständig um Konflikte die bei der Partitionierung und Verschmelzung von zwei Netzen zu behandeln.

2.20.1 Grobanalyse

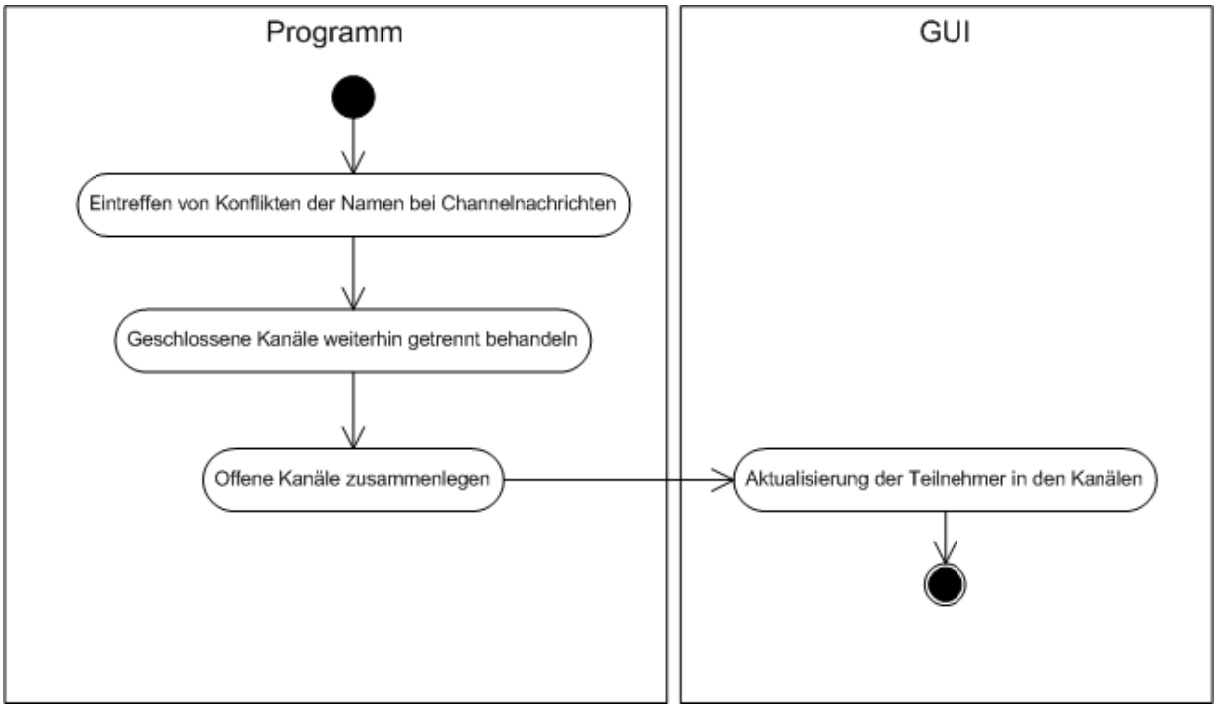

Abbildung 42: Aktivitätsdiagramm zur Behandlung von Kanalkonflikten

Das Diagramm beschreibt den Ablauf von der Verschmelzung zweier Netze. Probleme kann dies bereiten, weil Konflikte von Kanalnamen auftreten können und diese müssen behandelt werden. Geschlossene werden weiterhin getrennt behandelt. Offene werden zusammengelegt, so dass alle Benutzer aus beiden Kanälen dann in einem Kanal sind.

2.20.2 Feinanalyse

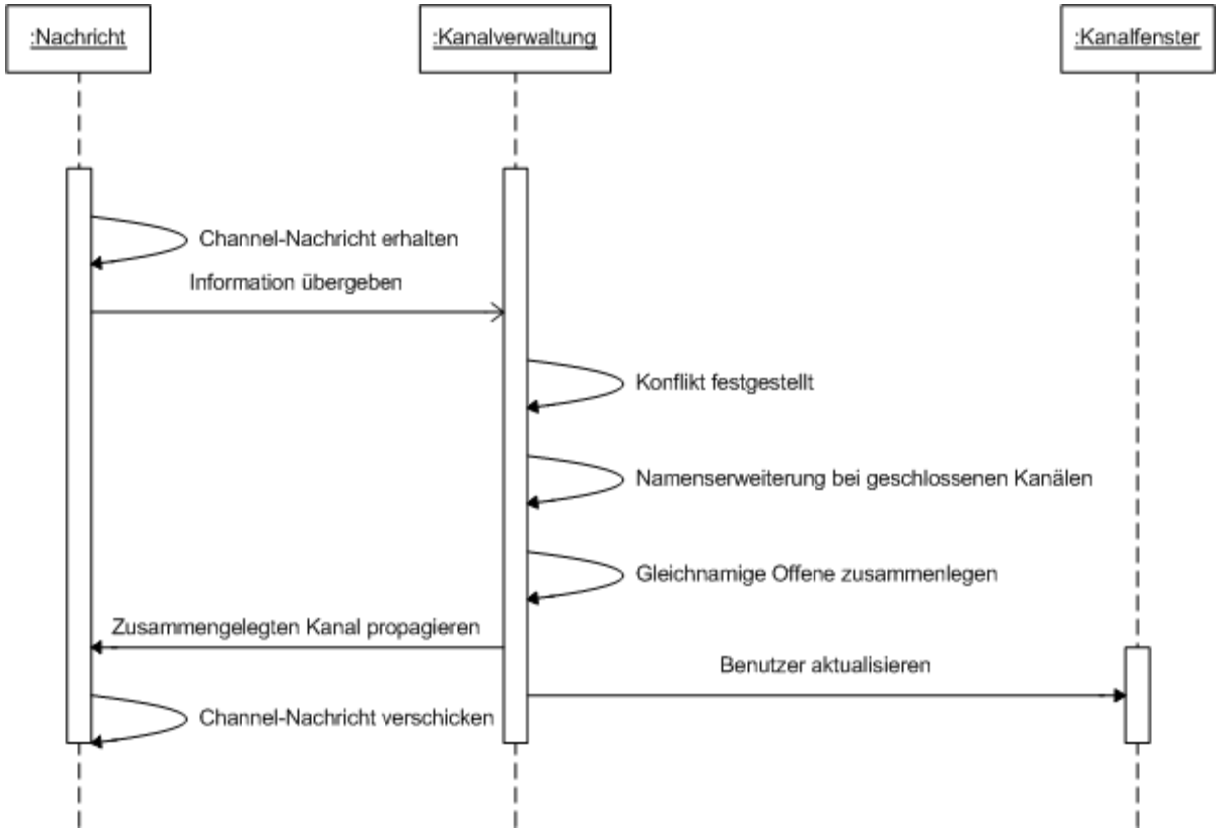

Abbildung 43: Sequenzdiagramm zu Behandlung von Kanalkonflikten

Das Diagramm zeigt die Interaktion von Objekten, um Kanalkonflikte zu beheben. Zunächst wird eine Channel-Nachricht erhalten und an das Kanalverwaltungsobjekt übergeben. Dies erkennt den Konflikt und führt bei geschlossenen Kanälen eine programminterne Namenserweiterung durch, damit die unterschieden werden können, falls ein Benutzer in Beiden sein möchte. Im Programm werden diese anhand der ID unterschiedenen. Dem Benutzer werden die Kanäle z.B. durch Kanal[1], Kanal[2] dargestellt. Bei offenen Kanälen werden gleichnamige zusammengelegt, indem die Kanalteilnehmerliste des Kanals mit der niedrigeren ID, um die neuen Benutzer erweitert wird und die andere Kanalteilnehmerliste entfernt wird. Anschließend wird der neue Kanal dem Nachrichtenobjekt übergeben, damit dies den anderen Netzteilnehmern mitgeteilt werden kann, indem eine Channel-Nachricht verschickt wird. Letztendlich werden die aktualisierten Benutzer noch im Kanalfenster angezeigt.

2.21 Analyse von Funktionalität /F210/ /220/ /230/ /240/ : Peerverwaltungsliste, Teilnehmerliste, Kanalliste, Kanalteilnehmerliste

Die Peerverwaltungsliste ist für den Infrastrukturmodus zuständig. Sie ist mit in dem Routingobjekt und wird beim Einschalten des Infrastrukturmodus aktiviert. Das Routingobjekt erstellt dann nach dieser Liste Routingtabellen und gibt diese Informationen, anstatt die Inhalte der Tabellen der Netzstruktur, zurück. Die Liste kann über die GUI vom Benutzer in Echtzeit angepasst werden.

Die Teilnehmerliste ist die Liste aller Benutzer die im aktuellen Netz vorhanden sind. Sie ist im Verwaltungsobjekt.

Die Kanalliste ist die Liste aller Kanäle die sich gerade im Netz befinden. Durch den Empfang von Channel-Nachrichten werden sie aktuell gehalten. Sie befindet sich ebenfalls im Verwaltungsobjekt.

Die Kanalteilnehmerliste ist eine Liste die für jeden Kanal separat angelegt wird. Sie enthält die Benutzer, die sich in dem jeweiligen Kanal befinden bzw. welche Zutritt zu einem geschlossenen Kanal haben. Sie wird auch durch Channel-Nachrichten aktuell gehalten und ist im Verwaltungsobjekt.

3 Resultierende Softwarearchitektur

3.1 Komponentenspezifikation

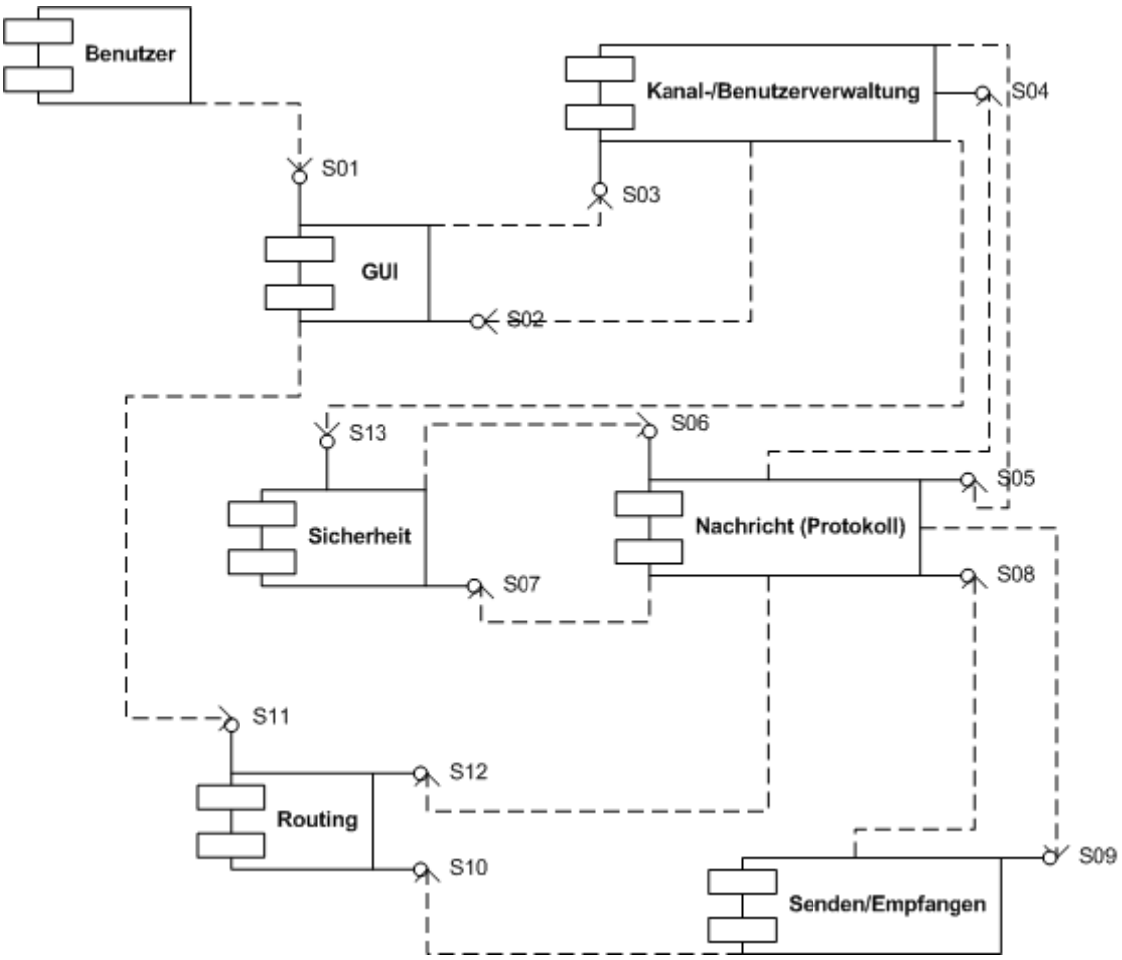

Abbildung 44: Komponentendiagramm

Das Diagramm zeigt die Komponenten die später im Programm miteinander kommunizieren. Die Benutzerkomponente steht für den menschlichen Teilnehmer, der mit dem Programm interagiert. Dieser verwendet das Programm nur über die GUI.

Die GUI leitet das weiter, was durch den Benutzer gefordert wird. Sie verwaltet die visuellen Objekte wie Kanalfenster, Eingabefeld, Teilnehmerfeld usw.

Die Kanal-/Benutzerverwaltung verwaltet die gesamten Informationen der Benutzer und Kanäle des Programms. Es interpretiert die Informationen, die von der Nachrichtenkomponente kommen und gibt sie in geeigneter Weise an die GUI weiter. Genauso nimmt sie Informationen von der GUI entgegen und gibt sie an die Nachrichtenkomponente weiter.

Die Nachrichtenkomponente kümmert sich um die Verwirklichung des Protokolls. Sie realisiert das Erstellen und parsen von XML-Dokumenten. Dementsprechend kann es auf Systemnachrichten reagieren, Nachrichten zum Senden erstellen und empfangene

Nachrichten parsen. Ebenfalls benutzt es die Sicherheitskomponente zum verschlüsseln, entschlüsseln, signieren und authentifizieren von Nachrichten. Ebenfalls übergibt es Informationen an die Routingkomponente bei eintreffenden Routingnachrichten, damit diese ihre Tabellen aktualisieren kann.

Die Sicherheitskomponente kümmert sich um die Sicherheit der Nachrichten und nutzt den Dienst der Nachrichtenkomponente, um Schlüssel und Zertifikate anzufordern.

Die Senden/Empfangen-Komponente ist die Schnittstelle zum Netz. Über sie kommen die UDP-Datagramme an und über sie werden die Nachrichten mittels UDP versenden. Es nutzt den Dienst der Routingkomponente, um an die richtigen Knoten zu senden.

Die Routingkomponente verwaltet die Routingtabellen und den Infrastrukturmodus. Die Nachrichten und Senden/Empfangen-Komponente wissen nicht, in welchem Modus sich die Komponente befindet, sondern nutzen nur deren Dienst, damit das Testen verwirklicht werden kann. Über die GUI kann der Infrastrukturmodus ein- und ausgeschaltet werden.

3.2 Schnittstellenspezifikation

Schnittstelle	**Aufgabenbeschreibung**	
	Operation	**Beschreibung**
S01: Benutzer-GUI-Interaktion	O10	Texteingabe
	O11	Senden
	O12	Datei anhängen
	O13	Kanäle erstellen
	O14	Kanäle verlassen
	O15	Kanäle beitreten
	O16	Infrastrukturmodus ein- / ausschalten
S02: Daten der GUI mitteilen	O20	Wenn eine Nachricht ankommt müssen der GUI die Ausgabeinformationen übergeben werden können. Ankommende Nachricht (Name, Text, Anhang)
S03: Daten von GUI der Kanal-/Benutzerverwaltung mitteilen und abrufen aus der Verwaltung	O30	Wenn ein Nachricht fertiggestellt worden ist müssen die nötigen Inhalte weitergegeben werden Nachricht losschicken(Text, Anhang)
	O31	Der GUI einen Kanalbeitritt mitteilen Kanal beitritt(Kanalnamen)
	O32	Um alle Kanäle anzeigen zu können muss die GUI die aktuellen Kanalnamen abrufen können Kanäle abrufen(Kanalnamen)
	O33	Im die Benutzer des Aktuellen Netzes anzeigen zu können muss dies aus der Verwaltung abgerufen werden können. Aktuelle Netzteilnehmer abrufen(Namen)
	O34	Es sollte möglich sein die Kanalteilnehmer abzurufen, um diese in der GUI anzeigen zu können Kanalteilnehmer abrufen(Namen)

	O35	Kanalerstellung der GUI mitteilen Kanal erstellen(Kanalnamen)
	O36	Aktualisieren wenn ein Teilnehmer ein Kanal verlässt. Kanal verlassen(Benutzernamen, Kanalnamen)
S04: Ankommende Informationen der Nachrichtenkomponente mitteilen	O40	Sendeinformationen übermitteln wenn eine Nachricht verschickt werden soll Nachricht senden(Kanalart, Kanal ID, Kanalname, Benutzername, Benutzer ID, Text, Anhang)
	O41	Aufruf eine Leave-Nachricht zu verschicken Leave(Benutzername, Benutzer ID, Kanalname, Kanal ID)
	O42	Aufruf eine Join-Nachricht zu verschickt. Join(Benutzername, Benutzer ID, Kanalname, Kanal ID)
	O43	Aufruf eine Channel-Nachricht zu verschicken Channel(Benutzernamen, Benutzer IDs, Channelname, Channel ID)
S05: Erhaltene Informationen der Verwaltung übergeben	O50	Empfangsinformationen übermitteln Nachricht (Benutzername, Benutzer ID, Kanal ID, Kanalname, Text, Anhang)
	O51	Um zu prüfen ob man Empfänger einer Nachricht ist. Benutzerinformationen abrufen(Benutzername, Benutzer ID)
	O52	Eintreffen einer Channel-Nachricht mitteilen Channel-Nachricht(Benutzernamen, Benutzer IDs, Kanalname, Kanal ID)
	O53	Eintreffen einer Join-Nachricht mitteilen Join-Nachricht(Benutzername, Benutzer ID, Kanalname, Kanal ID)
	O54	Eintreffen einer Leave-Nachricht mitteilen Leave-Nachricht(Benutzername, Benutzer ID, Kanalname, Kanal ID)
S06: Anforderungen mitteilen	O60	Wenn ein Zertifikat benötig wird muss ein angefordert werden. Zertifikat anfordern(Benutzer ID)
	O61	Wenn ein Schlüssel benötigt wird muss dieser angefordert werden. Gemeinsamen Schlüssel anfordern (Benutzer ID, Kanal ID)
S07: Informationen der Sicherheitskomponente übertragen	O70	Verschlüsseln für einen geschlossenen Kanal verschlüsselnClosed (Text, Anhang, Benutzer ID, Kanal ID)
	O71	Signieren für geschlossenen und offenen Kanal signieren (Text, Anhang, Benutzer ID)
	O72	Entschlüsselt für einen geschlossenen Kanal entschlüsselnClosed (Verschlüsselte Daten, Kanal ID, Benutzer ID)
	O73	Zum authentifizieren von Nachrichten authentifizieren (Text, Benutzer ID)
	O74	Verschlüsselt für anonymen Kanal verschlüsselnAnonym (Text, Anhang, Benutzer ID, Kanal ID)
	O75	Entschlüsseln einer Obscure-Nachricht EntschlüsselnObscure(verschlüsselte Daten)

S08: Erhaltene Nachricht zum Parsen übergeben	O80	Eintreffende XML Nachrichten übergeben empfangeXML(XML-Datei)
S09: Erstellte Nachrichten zum senden übergeben	O90	XML-Dokumente zum Senden übergeben. sendeXML(XML-Datei)
S10: Informationen zum Routing abrufen	O100	Nächsten Knoten für die Empfänger abrufen Knoten abrufen(Empfänger)
S11: Infrastrukturmodus wechseln	O110	Infrastrukturmodus ein- / ausschalten ChangeMode()
S12: Routinginformationen erhalten	O120	Beim Eintreffen von Routingnachrichten erhält die Komponente hier die Informationen ÜbergebeRoutinginformationen(Routing Daten)
	O121	Um anonyme Nachrichten zu erstellen kann abgerufen werden welche Benutzer sich im Netz befinden. Benutzer abrufen(Benutzer IDs)
S13: Vorsorgliches Anfordern von Zertifikaten und Schlüsseln	O130	Zertifikate Anfordern nach Beitritt in einen Kanal Zertifkate holen(Benutzer IDs)
	O131	Schlüssel anfordern nach Beitritt in einen geschlossenen Kanal Schlüssel holen(Kanal ID)

3.3 Protokolle für die Benutzung der Komponenten

3.3.1 Wiederverwendung im eigenen Programm

Die Komponenten eignen sich im Programm selbst nicht für die Wiederverwendung. Es besteht aber die Möglichkeit Komponenten auszutauschen. Beispielsweise kann man die Senden/Empfangen Komponente austauschen, wenn man nicht mehr über UDP versenden möchte oder die Sicherheitskomponente austauschen, um andere sicherheitsverfahren zu verwenden.

3.3.2 Wiederverwendung in Möglichen anderen Programmen

Es ist zwar nicht zwingend notwendig, aber bei manchen Komponenten könnte man sich vorstellen sie in anderen Programmen zu verwenden.
Die Nachrichtenkomponente könnte zum Beispiel in anderen Programmen integriert werden, um sich nicht Gedanken über das Protokoll machen zu müssen und so trotzdem eine Kommunikation zu ermöglichen.
Die Sicherheitskomponente könnte zum verschlüsseln oder signieren von anderen Inhalten außer Nachrichten benutzt werden.