TECHNISCHE UNIVERSITÄT
CHEMNITZ

Fakultät für Informatik

Professur Verteilte und Selbstorganisierende Rechnersysteme

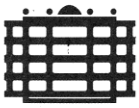
Masterarbeit

Serialisierung von Emotionen in der textuellen Kommunikation

Autor: Gerhard Fobe

Erstgutachter: Prof. Dr.-Ing. Martin Gaedke
Zweitgutachter: Dipl.-Inf. Stefan Wild

Abgabe: 10.09.2012



TECHNISCHE UNIVERSITÄT
CHEMNITZ

Aufgabenstellung

zur

Abschlussarbeit
im Studiengang Master Data and Web Engineering

für

Herrn Gerhard Fobe
geb. am 30. September 1986 in Görlitz

zum Thema

Serialisierung von Emotionen in der textuellen Kommunikation

Erstgutachter:	Prof. Dr. Martin Gaedke
Ausgabedatum:	03.04.2012
Abgabedatum:	10.09.2012
Tag der Abgabe:	

Unterschrift: .....

Prof. Dr. F. Hamker
Vorsitzender des Prüfungsausschusses

Aufgabenstellung

In der heutigen Zeit gewinnt die Kommunikation per Instant Messaging und sozialer Netzwerke immer mehr an Bedeutung. Der textuellen Nachrichtenübermittlung mangelt es dabei jedoch an eindeutiger Emotionalität. Es obliegt dem jeweiligen Empfänger die Nachricht im richtigen Kontext zu erfassen und zu interpretieren. Dies stellt vor allem in der interkulturellen Kommunikation ein erhebliches Problem dar, weil die unterschiedlichen Kulturen verschiedene Elemente in der Kommunikation abweichend deuten.

Ziel dieser Arbeit ist eine geeignete Lösung zur Serialisierung und Übertragung von Emotionen zu konzipieren. Ausgangslage sollen verschiedene Quellen zur Eingabe von Emotionen sein. Die so erfassten Emotionen müssen weiterverarbeitet werden und dabei ein geeignetes Serialisierungsformat annehmen, das sich zur Übertragung im Bereich des Instant Messaging und sozialer Netzwerke eignet. Eine exemplarische Implementierung anhand eines Prototyps soll die Umsetzbarkeit der zu erarbeitenden Lösung aufzeigen.

Abstract

Im Rahmen der vorliegenden Arbeit wird eine Lösung zur Serialisierung und Übertragung von aus mehreren verschiedenen Quellen stammenden Emotionen konzipiert. Anschließend wird mit Hilfe der exemplarischen Implementierung eines Chatclients die praktische Umsetzbarkeit der erarbeiteten Lösung aufgezeigt.

Hierfür wird eine erweiterbare Systemarchitektur darlegt, in der über definierte Schnittstellen angebundene Plugins die Verarbeitung in die Softwarelösung ein- und ausgehender Emotionen übernehmen. Die Kommunikation zwischen den Plugins, denen weitere Systeme zur Emotionserkennung und -verarbeitung vorgeschaltet sein können, und der eigentlichen Anwendung erfolgt dabei mittels EmotionML. Des Weiteren findet eine mittels individueller Benutzerkonfiguration beeinflussbare Bündelung der verschiedenen eingehenden Emotionen zu einer aggregierten in EmotionML abgebildeten Emotion statt.

Während der Nutzung des XMPP als Chatprotokoll für den Prototyp wird in eine XMPP-Nachricht EmotionML eingebettet, damit eine vom Chattext unabhängige Übertragung der Emotion möglich ist. Um die Bedeutung dieses Vorgehens und die damit verbundenen Möglichkeiten besser aufzeigen zu können, wird eine Transformation von westlichen zu östlichen Emoticons durch spezielle Plugins unternommen. Dabei werden Emoticon und Emotion semantisch innerhalb der durch EmotionML erweiterten XMPP-Nachricht in Form von RDF notiert. Dies erfolgt durch die Nutzung von Smiley Ontology und Emotion Ontology.

Schlüsselwörter: Emotion, Kommunikation, Instant Messaging, soziales Netzwerk, Emoticon, XMPP, EmotionML, Prototyp

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Quelltextverzeichnis

Abkürzungsverzeichnis

AIM	**A**OL **I**nstant **M**essenger
ASCII	**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange
CD	**C**ompact **D**isc
DLL	**D**ynamic **L**ink **L**ibrary
DTD	**D**ocument **T**ype **D**efinition
EMMA	**E**xtensible **M**ultiModal Annotation **M**arkup **L**anguage
EmotionML	**Emotion M**arkup **L**anguage
eRDF	**E**mbedded **RDF**
GRDDL	**G**leaning **R**esource **D**escriptions from **D**ialects of **L**anguages
GTalk	**G**oogle **Talk**
HTML	**H**yper**t**ext **M**arkup **L**anguage
HTML5	**H**yper**t**ext **M**arkup **L**anguage Version **5**
IBM	**I**nternational **B**usiness **M**achines Corporation
ID	eindeutiger **Id**entifikator
JID	**J**abber **Id**entifier
MSN	The **M**icro**s**oft **N**etwork
OSCAR	**O**pen **S**ystem for **C**ommunic**a**tion in **R**ealtime
PHP	**P**HP: **H**ypertext **P**reprocessor
RDF	**R**esource **D**escription **F**ramework
RDFa	**RDF** in **A**ttributes
RFC	**R**equest **f**or **C**omments
SDK	**S**oftware **D**evelopment **K**it
URI	**U**niform **R**esource **I**dentifier
URL	**U**niform **R**esource **L**ocator
URN	**U**niform **R**esource **N**ame
VoIP	**V**oice **o**ver **IP**
W3C	**W**orld **W**ide **W**eb **C**onsortium
XEP	**X**MPP **E**xtension **P**rotocol
XHTML	E**x**tensible **H**yper**t**ext **M**arkup **L**anguage
XML	E**x**tensible **M**arkup **L**anguage
XMPP	E**x**tensible **M**essaging and **P**resence **P**rotocol
XPath	**X**ML **Path** Language
XSD	**X**ML **S**chema **D**efinition
XSF	**X**MPP **S**tandards **F**oundation
XSL	E**x**tensible **S**tylesheet **L**anguage
XSLT	**XSL**-**T**ransformation

1 Einleitung

Das Internet verbindet Menschen – weltweit und darüber hinaus. Wie die wachsende Bedeutung der Kommunikation mittels Instant Messaging [1] und sozialer Netzwerke [2] zeigt, ist die textuelle Kommunikation ein probates Mittel der Verständigung. An einer für eine gute Kommunikation notwendiger eindeutiger Emotionalität mangelt es jedoch. Daher soll die vorliegende Arbeit dabei helfen, diese Emotionalität in der textuellen Nachrichtenübermittlung zu ermöglichen. Dafür ist es nötig eine geeignete Lösung zur Serialisierung und Übertragung von Emotionen zu konzipieren. Als Ausgangslage für dieses Vorhaben dienen verschiedene Quellen zur Eingabe von Emotionen. Die aus diesen Quellen erfassten Emotionen sollen während einer Weiterverarbeitung ein geeignetes Serialisierungsformat annehmen, das sich zur Übertragung im Bereich des Instant Messagings und sozialer Netzwerke eignet. Um die Umsetzbarkeit der erarbeiteten Lösung aufzuzeigen, wird nach erfolgreicher Konzeption des Lösungskonzeptes eine exemplarische Implementierung anhand eines Prototyps unternommen.

1.1 Vorherrschende Probleme im Kontext der Aufgabe

Wie die Kapitel 1.1 und 2.2 im weiteren Verlauf zeigen, spielt die Emotionalität bei der Kommunikation eine bedeutende Rolle. Zumeist wird diese Emotionalität jedoch nur indirekt und unbewusst übertragen. Ein großer Teil davon mittels Körpersprache. Fällt dieser Teil nun während der textuellen Nachrichtenübermittlung weg, fehlen wichtige Aspekte, die für eine erfolgreiche Erkennung der Emotionalität in der Kommunikation und der damit zusammenhängenden Interpretation der Worte nötig sind. Somit besteht die Möglichkeit, dass Missverständnisse entstehen. Die Folge ist ein Verlust an Effektivität und Güte der Kommunikation. Ist ein regelmäßiges persönliches Treffen der Kommunikationsteilnehmer möglich, so können die durch das Fehlen der Emotionalität verursachten Folgen verringert werden, da mehr über das Wesen des Kommunikationspartners bekannt ist, dieser und sein Verhalten also besser gekannt wird. Durch die oft großen räumlichen Distanzen der Kommunikationsteilnehmer ist dies jedoch nicht oft oder teilweise gar nicht möglich.

Diese räumliche Distanz ist nicht durch territoriale oder kulturelle Grenzen beschränkt. Aufgrund der globalen Verfügbarkeit des weltumspannenden Internets ist Kommunikation selbst voneinander territorial weit entfernten Personen mit geringem Aufwand möglich, so dass die räumliche Distanz an Bedeutung verliert und verschiedene regional stark verteilte Kulturgruppen zueinander gebracht werden. Doch gerade in dieser interkulturellen Kommunikation stellt die fehlende Emotionalität eine noch größere Hürde dar, da das Wesen und die Art der Kommunikation eine andere als die gewohnte sein kann oder wenigstens andere Facetten aufweist. Da es dem Empfänger einer Nachricht obliegt, diese im richtigen Kontext zu erfassen und zu interpretieren, ist die Wahrscheinlichkeit sehr hoch, dass dieser eine Interpretation gemäß den Maßstäben seines Kulturraumes vornehmen wird. Aufgrund dieser Tatsache kann es zu Informationsverlust oder Fehldeutungen kommen. Daraus entstehende Konflikte sind nicht auszuschließen.

Primär im Bereich der privaten textuellen Kommunikation (vgl. [3]) wird zur besseren und einfacheren Übermittlung von Emotionen auf eine Form der Emotionssymbolik zurückgegriffen – dem Emoticon (vgl. Kapitel 3.2.1). Dieses stellt zumeist eine dem menschlichen Gesicht und seiner Mimik nachempfundene Abbildung dar. Aufgrund eines globalen und verteilten Lernprozesses während der alltäglichen Kommunikation sind

Zuordnungen von Emotionen zu textuellen oder bildlichen Abbildungen entstanden. Ergänzt werden diese durch Elemente des Netzjargons (vgl. Kapitel 3.2.1.4). Jene sehr weit verbreitete und häufig genutzte Form der Emotionsübertragung weiset jedoch Probleme auf, die zu Nichteindeutigkeit und daraus resultierenden Missverständnissen führen können (vgl. Kapitel 3.2.1.6). Durch den verteilten Lernprozess, unter anderem kombiniert mit künstlerischer Gestaltungsvielfalt, ist keine einheitliche und klar spezifizierte Interpretation von Emoticons vorhanden. Dieses Problem verschärft sich bei größer werdenden Unterschieden zwischen einzelnen Kulturen. Hierbei spielen vor allem unterschiedliche Interpretationsweisen sowie unterschiedliche Emotionssymboliken eine tragende Rolle.

Eine große Bedeutung bei der Verbreitung von Emoticons ist in dem geringen Aufwand für die Auszeichnung der Emotion für die Nutzer zu suchen. Um diesen die Erfassung und Darstellung ihrer Emotionen weiter zu vereinfachen, ist es möglich die Emotionen automatisiert mittels verschiedener Algorithmen zu erkennen. Zwar sind in diesem Bereich schon vielfältige Projekte entstanden, die eingegrenzte Emotionen gut voneinander unterscheiden können (vgl. Kapitel 3.2.3), jedoch ist zu beobachten, dass keine einheitliche Auszeichnung der Emotion stattfindet. Dadurch ist eine einfache und flexible Weiterverarbeitung der Emotion nur schwer möglich. Dies wiederum hemmt die Nutzbarkeit vorhandener Prototypen für weiterführende Projekte.

Auf Grund dieser Ausgangslage soll mit Hilfe eines Lösungsansatzes und einem entwickelten Prototyp gezeigt werden, dass eine Verbesserung in der Kommunikation möglich ist.

1.2 Lösungsansatz und Prototyp

Im Rahmen dieser Ausarbeitung wird eine an vielen Stellen erweiterbare Systemarchitektur entworfen. Diese dient als Basis für einen Chatclient und dessen Implementierung als Prototyp. Für die programminterne Verarbeitung der Emotionen wird die Emotion Markup Language (EmotionML) [4] eingesetzt. Diese speziell für die Abbildung von Emotionen geschaffene Auszeichnungssprache ermöglicht eine sehr flexible, aber eindeutige Abbildung von Emotionen. Um der Tatsache gerecht zu werden, dass Emotionen aus mehreren Quellen kommen können, wird eine Input-Schnittstelle in EmotionML definiert, die eine Zuführung von Emotionen in die Anwendung ermöglicht. Aufgrund der vorherrschenden Heterogenität der Schnittstellen vorhandener Systeme zur Emotionserkennung werden der Inputschnittstelle so genannte Plugins vorgeschaltet. Dies hat das Ziel eine höhere Flexibilität der Lösung herzustellen**.** Die Plugins können dabei selbst einen Algorithmus zur Emotionserkennung beinhalten, der über eine festgelegte Schnittstelle mit der Anwendung EmotionML austauscht, oder sie stellen einen Adapter zu einer, dem Plugin vorgelagerten, emotionserkennenden Anwendung dar. Die verschiedenen eingehenden Emotionen werden nach ihrer Übergabe an die Anwendung gemäß einer nutzerspezifischen Konfiguration zueinander gewichtet und zu einer gemeinsamen Emotion aggregiert. Diese Emotion wird zusammen mit dem Chattext des Nutzers in ein um EmotionML erweitertes Nachrichtenpaket des Extensible Messaging and Presence Protocol (XMPP) eingebettet und an den Empfänger übertragen. Dort wird dieses Nachrichtenformat wieder aufgesplittet und somit die Emotion separiert. Diese kann im Anschluss über eine der Input-Schnittstelle ähnliche Output-Schnittstelle außerhalb der Anwendung verfügbar gemacht werden. Um Konflikte zwischen verschiedenen Plugins zu minimieren, werden diese zu so genannten

Extensions zusammengefasst, was eine gemeinsame Nutzung von Programmbestandteilen durch die innerhalb der Extension befindlichen Plugins ermöglicht.

Dem in dieser Arbeit entwickelten Prototyp liegen zwei solcher Extensions bei. Eine einfache Extension namens *Plainemotion* verdeutlicht den Aufbau einer Erweiterung für die Anwendung und stellt verschiedene Möglichkeiten der Interaktion mit externen Systemen vor. Die Extension *Emoticonhandling* hingegen ist komplexer gestaltet. Sie beschäftigt sich mit der Extraktion von Emoticons aus dem Chattext des Nutzers sowie deren Umwandlung in EmotionML. Weiterhin werden hiermit speziell die Vorteile der Lösung in der interkulturellen Kommunikation aufgezeigt, da sie mit den Emoticons verschiedener Kulturräume arbeitet und somit die Interpretation des Emoticons beim Empfänger vereinfachen kann. Durch dieses Plugin werden auch erweiterte Möglichkeiten mit Emoticons und Emotionen durch den Einsatz von Technologien aus dem Bereich des Semantic Web aufgezeigt.

1.3 Aufbau der Arbeit

Im weiteren Verlauf der Arbeit werden allgemeine Grundlagen im Bereich der Kommunikation, Emotion und des Semantic Web behandelt, die als Grundlage für die weitere Ausarbeitung notwendig sind. In Kapitel 3.1 werden Anforderungen an die Lösung definiert und im Anschluss Bewertungskriterien abgeleitet. Weiterhin wird ein Bewertungsschema aufgestellt, um Aussagen über die Gesamtgüte einer Lösungsvariante treffen zu können. Folgend werden im Verlauf des Kapitel 3 verschiedene Szenarien näher untersucht, die die Anforderungen erfüllen könnten. Im Detail sind dies die Übermittlung von Emotionen mittels Emoticons und Elementen des Netzjargons, eine Verwendung von Auszeichnungssprachen und Ontologien zur Abbildung einer Emotion sowie Möglichkeiten im Zusammenhang mit einer Emotionserkennung. Leider stellte sich bei der in Kapitel 3.3 getätigten Evaluierung heraus, dass keines der betrachteten Szenarien den Anforderungen im vollen Umfang entsprechen kann. Allerdings zeigte sich auch, dass eine Kombination der Szenarien Emoticons und Netzjargon mit Auszeichnungssprachen und Ontologien durchaus das Potenzial hat, gemeinsam mit einer flexiblen Architektur des Gesamtsystems eine den Anforderungen gerecht werdende Lösung darzustellen. Die Entwicklung eines entsprechenden Lösungskonzeptes erfolgt in Kapitel 4. In diesem wird durch Kapitel 4.1 eine Betrachtung durchgeführt, welche vorhandenen standardisierten Protokolle für die Entwicklung der Lösung genutzt werden können. Für den Bereich des Instant Messagings eignet sich XMPP im Vergleich zu allen anderen untersuchten Möglichkeiten am besten. Bei der Abbildung von Emotionen verspricht EmotionML den größten Erfolg. Im weiteren Verlauf von Kapitel 4.1 werden diese beiden Protokolle miteinander kombiniert und bilden das für die Anwendung genutzte Übertragungsformat. Folgend wird in Kapitel 4.2 eine flexible Systemarchitektur vorgestellt, die durch die in Kapitel 4.4 vorgestellten Schnittstellen einfach durch Plugins erweitert werden kann. Diese interagieren wiederum mit einem in Kapitel 4.3 definierten Chatclient, der den Prototyp der Lösung bildet. Die Beschreibung einer nutzerspezifischen Gewichtung, nach der die eingehenden Emotionen aus den Plugins zusammengefasst werden, wird in Kapitel 4.5 aufgezeigt. In Kapitel 5 wird die eigentliche Implementierung der Lösung und des Prototypen beschrieben. Dabei werden der softwaretechnische Aufbau der Anwendung sowie von ihr genutzte Programmbibliotheken definiert. Darunter befindet sich auch eine neu geschaffene Programmbibliothek für EmotionML. Da EmotionML einen noch in der Entwicklung befindlichen Webstandard darstellt, ist für diesen noch keine nutzbare Programmbibliothek vorhanden, so dass sie in Kapitel 5.1 spezifiziert werden muss. Es folgt eine Beschreibung der Implementierung des

Chatclients in Kapitel 5.35.3. Nach dessen Vollendung werden in Kapitel 5.4 Beispielplugins beschrieben, die dem Prototyp ermöglichen, die Umsetzbarkeit der zu erarbeitenden Lösung aufzuzeigen. Den Abschluss dieses Kapitels bildet eine Betrachtung, inwieweit sich die einzelnen Bestandteile der Implementierung auch für weitere Projekte nutzen lassen. Um zu überprüfen, ob die implementierte Lösung den Anforderungen gerecht wird, wird diese in Kapitel 6 analog zu den verschiedenen in Kapitel 3.3 betrachteten Szenarien bewertet.

2 Grundlagen

Die vorliegende Arbeit beschäftigt sich mit Emotionen in der textuellen Kommunikation. Daher ist es zunächst erforderlich die Begriffe Kommunikation und Emotion näher zu betrachten. Des Weiteren nutzt der als Prototyp implementierte Chatclient Techniken des Semantic Web, so dass auch hier eine einführende Betrachtung notwendig ist.

2.1 Kommunikation

Das primäre Ziel eines Chatclients, wie er in Form des Prototyps implementiert wurde, ist die Ermöglichung einer Kommunikation mehrerer Teilnehmer des Chats. Schon der Kybernetiker Norbert Wiener äußerte in den 1950er-Jahren „Kommunikation ist der Zement, der eine Organisation zusammenhält“. Die Kommunikation (abgeleitet von lateinisch communicare, „mitteilen“) ist der „Austausch einer Information zwischen zwei dynamischen Systemen bzw. Teilsystemen“. Dabei überträgt der Sender eine zu sendende Information aus einer Informationsquelle über einen Kommunikationskanal zu einem Empfänger, welcher die Information wieder einer Informationssenke übergibt. Um die Information über den Kommunikationskanal zu transferieren, wird aus der Information eine Nachricht erzeugt, welche zur Erkennung von Übertragungsfehlern mit einer Redundanz versehen wird [5].

Diese Redundanz äußert sich unter anderem in mehreren **Dimensionen einer Kommunikation**. So gibt es neben der verbalen Kommunikation (in Abbildung 1 grün dargestellt) eine nonverbale Kommunikation und paraverbale Bestandteile, welche nach Krämer/Quappe die größte Bedeutung für die Rückgewinnung der übertragenden Information inne haben [6]. Die **verbale Kommunikation** beinhaltet zumeist die Sprache bzw. den Inhalt. Verschiedene von der jeweiligen Kultur abhängige Kommunikationsstile, die sich in indirekter und direkter Kommunikation sowie in einem niedrigen oder hohen Kontextbezug wiederfinden, können das gegenseitige Verstehen behindern und somit für Missverständnisse verantwortlich sein. Eine **nonverbale Kommunikation** beinhaltet den nichtsprachlichen Bereich der Kommunikation. In diesen Bereich fallen die Elemente der eigentlichen Körpersprache wie Gestik und Mimik, aber auch äußere Erscheinung wie Kleidung oder Frisur. Ergänzt werden sie durch vegetative Symptome wie Erröten oder Schwitzen. Nonverbale Signale werden zumeist unbewusst übermittelt. Die Ebenen der Kommunikation werden vervollständigt durch die **paraverbale Kommunikation**. Hierbei wird die Art und Weise der Sprache bzw. des Sprechens betrachtet. Die daraus folgende

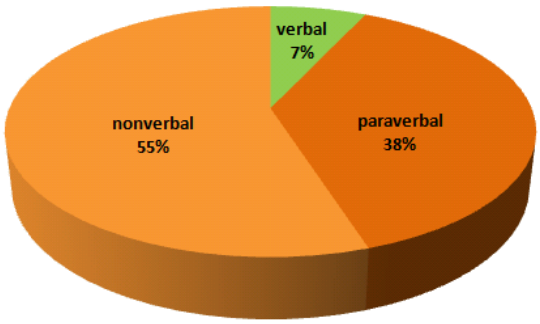


Abbildung 1: Anteile der Dimensionen in einer Kommunikation nach Krämer/Quappe

Interpretation von Elementen wie Tonfall, Stimmlage, Artikulation oder Lautstärke ist in einem hohen Maße kulturell geprägt [7].

In der Kommunikation spielt sowohl das Senden und Empfangen von Nachrichten als auch deren Übermittlung eine zentrale Rolle. An allen drei Stellen kann es zu einer Verwischung der eigentlichen Information und somit zu Missverständnissen kommen.

Für den Versand und Empfang von Nachrichten zwischen Personen hat der Psychologe Friedemann Schulz von Thun herausgefunden, dass die Interpretation jeder Nachricht auf **vier Ebenen** erfolgt. Daher leiten sich für dieses Modell die Begriffe *4-Ohren-Modell* (Betrachtung von Empfängerseite) oder auch *4-Schnäbel-Modell* (Betrachtung von Senderseite) ab [8].

Während auf einer Sachebene die eigentlichen Sachinformationen im Vordergrund stehen und Daten und Fakten vermittelt werden, rückt hingegen auf einer Beziehungsebene die eigentliche zwischenmenschliche Beziehung in den Vordergrund. Hier sind somit emotionale Aspekte der Kommunikation wie die Art der Formulierung, der Tonfall, die Körperhaltung oder Mimik und Gestik entscheidend. Ergänzt werden diese beiden Ebenen durch eine Selbstoffenbarungsebene, bei der der Empfänger nähere Informationen darüber ableiten kann, wie der Sender selbst zu der übertragenden Information steht. Das Modell wird komplettiert durch eine Apellebene, die Auskunft darüber geben kann, was der Sender mit der Übermittlung seiner Nachricht beim Empfänger bewirken will. Dies können unter anderem Handlungsempfehlungen, Wünsche, Ratschläge oder gar Apelle sein, die direkt oder unterschwellig in der Nachricht angesprochen werden. Das Ziel der Nachrichtenübermittlung kann also durch diese Ebene herausgefunden werden. Die Interpretation der Beziehungsebene bestimmt im großen Maße den Inhaltsaspekt der Kommunikation, so dass sie als Metakommunikation einzuordnen ist.

Durch die vorherrschende ständige Interpretation der vier Ebenen können schnell Missverständnisse zwischen den Kommunikationspartnern entstehen, da der Empfänger die Inforationen in der übertragenen Nachricht nicht nur objektiv auf der Sachebene interpretiert, sondern auch versucht weitere Informationen über den Sender und seine Ziele in den Text zu interpretieren. Zudem ist gerade die Interpretation emotionaler Aspekte kulturell sehr unterschiedlich. Wird kein Inhalt auf der Sachebene übermittelt (bspw. durch keine Übertragung von Text) findet nur eine Kommunikation über die restlichen drei Ebenen statt.

Nicht nur bei der Interpretation der übermittelten Nachricht anhand der Kommunikationsebenen treten Probleme auf. Eine **Störung des Kommunikationskanals** kann auf vielfältige Art und Weise auftreten, so dass eine erfolgreiche Kommunikation verhindert wird (vgl. Abbildung 2). Dies wurde mit dem von Claude Shannon und Warren Weaver in ihrem in den 1940er Jahren beschriebenen Kommunikationsmodell festgestellt. Es besagt unter anderem, dass ein Signal während der Übertragung im Kommunikationsmedium Störungen unterworfen sein kann, so dass die beim Empfänger eingehende und vom Sender geschickte Nachricht beim Empfänger verändert ankommen kann [9]. Dieses Modell lässt sich nicht nur auf die physische Übertragung von Nachrichten anwenden, bei der die Störquelle ein Leitungsrauschen oder störende Nebengeräusche zu einem Gespräch sind. Auch eine Anwendung auf logische Störquellen, wie es eine durch kulturelle Unterschiede geprägte differente Interpretation emotionaler Aspekte darstellt, ist möglich. Somit wird das Potential für Missverständnisse weiter verstärkt.

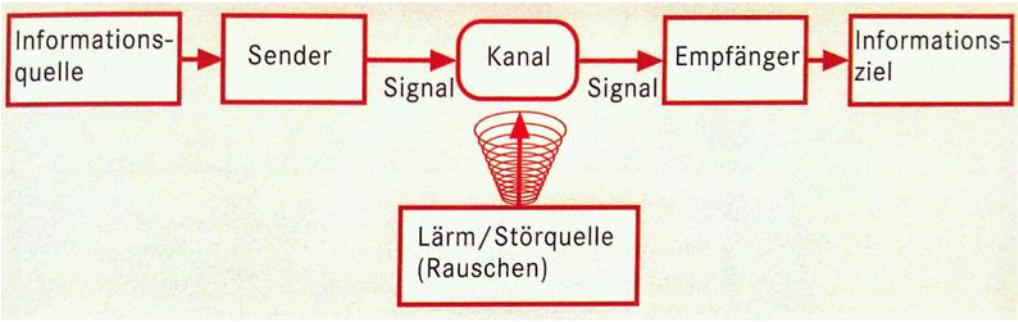


Abbildung 2: Ablauf einer Kommunikationskette mit Störung des Kommunikationskanals[1]

Bei der parallelen Übermittlung sprachlicher und nicht sprachlicher Aspekte auf den Kommunikationskanälen ist es möglich, dass diese sich ergänzen oder widersprechen. Schulz von Thun definiert die in sich stimmigen Nachrichten als kongruent und verweist darauf, dass inkongruente Nachrichten den Empfänger verwirren und somit Missverständnisse wahrscheinlich sind [8].

Eine weitere Störung der Kommunikation kann durch eine unterschiedliche Interpretation der über die verschiedenen Ebenen und Dimensionen der Kommunikation aufgenommenen Informationen erfolgen. Dies ist eng verbunden mit kulturellen Unterschieden zwischen Sender und Empfänger. Als Beispiel in der nonverbalen Kommunikation kann das Handreichen zur Begrüßung herangezogen werden. Dieses Vorgehen wird in Deutschland als obligatorische Begrüßungsgeste empfunden, ist jedoch in Japan unüblich und wird daher verwirrend empfunden, da die unterschwellig übermittelte Handlungsanweisung nicht bekannt ist [10].

2.2 Emotion

Eine Emotion (abgeleitet von lateinisch emovere, „herausbewegen“) ist Gefühl oder auch eine Gefühlsregung. Sie stellt einen „Teil der dem Menschen eigenen mannigfachen Gemütsbewegungen; mit spezif. Qualität, Intensität und Dauer“ [11] dar. Somit ist die Emotion eine „physiologische Zustandsform, bei der die Handlungsbereitschaft eines Individuums durch einen hohen Anteil vegetativer Vorgänge bestimmt wird“ [5]. Sie wird beeinflusst durch die Wahrnehmung eines äußeren oder inneren Ereignisses oder Anlasses sowie durch die eigene Motivation. Als Reaktion auf eine Emotion kann es zu abgeleiteten Handlungen kommen. Diese können sich in vegetativen Reaktionen (z.B. Erröten) oder Verhaltensänderungen (z.B. Mimik, Gestik) äußern. Zudem können mehrere Emotionen gleichzeitig auftreten. Ein Beispiel hierfür wäre der Besuch einer Geisterbahn, bei der sich Furcht und Freude überlagern. Eine genaue Definition, was eine Emotion ist, ist so schwierig, dass Schmidt-Atzert feststellen musste: „Bislang ist kein Konsens festzustellen, was man unter einer Emotion zu verstehen hat“ [12].

„Jeder weiß, was eine Emotion ist - bis man ihn darum bittet, sie zu definieren“[2]. Aufgrund der Komplexität von Emotionen ist eine genaue Einordnung und Abgrenzung von ihnen sehr schwierig. Um dies dennoch bewerkstelligen zu können, lassen sich für eine Emotion mehrere Einteilungen vornehmen. Zum einen ist eine grobe Einteilung emotionsbetreffender Zustände in Kategorien möglich. Diese, bspw. mit „glücklich“ oder „traurig“ benannten,

[1] Grafik entnommen aus [5]
[2] Fehr & Russell, 1984

Kategorien finden dabei auch im alltäglichen Sprachgebrauch oft Anwendung. Douglas-Cowie et al. gelang es eine Liste mit 48 unterschiedlichen Emotionskategorien zu definieren [13]. Eine weitere Möglichkeit der Unterteilung einer Emotion existiert durch ihre Einteilung in mehrere Dimensionen, die das Ausmaß im Kontext der Emotion verdeutlichen. Nach Fontaine et al. sind dabei primär vier verschiedene Dimensionen ausschlaggebend: Wertigkeit, Stärke, Erregung und Unvorhersehbarkeit [14]. Eine weitere Einteilung ist über die Bewertung der Emotion in dem Kontext, in dem sie auftritt, möglich. Somit kann eine Einteilung dahingehend vollzogen werden, ob jemand mit der Emotion vertraut ist oder die Person der Emotion positiv oder negativ gegenüber steht. Ein für diesen Bereich relevantes Modell ist in dem nach den Autoren benannten OCC-Modell von Ontony, Clore und Collins existent [15]. Eine weitere Möglichkeit der Beschreibung von Emotionen ist anhand der Handlungstendenzen, die sie beim Fühlenden auslösen, möglich. Somit kann unter anderem die Situation beschrieben werden, dass jemand einer angstbereitenden Situation aus dem Weg gehen möchte. Diese Bereitschaft zur Handlung aufgrund einer Emotion wird ausführlich von Frijda behandelt [16]. Die vorgestellten verschiedenen Möglichkeiten der Einordnung von Emotionen lassen sich dabei auch kombinieren, was eine konkretere und vielfältigere Definition der Emotion ermöglicht.

2.3 Semantic Web

Das Semantic Web ist eine Erweiterung des bestehenden Webs, in welcher Informationen mit wohl definierter Bedeutung vorliegen. Dies ermöglicht die bessere kooperative Arbeit zwischen Computer und Menschen [17]. Diese neue Form der Wissensrepräsentation hat das Ziel „die Bedeutung von Informationen für Computer verwertbar zu machen, damit Lösungsansätze für die Probleme, welche sich aus der Fülle der im Web präsenten Informationen ergeben […], geschaffen werden können“ [18]. Somit ist es unter anderem leichter möglich mehrere Informationen verschiedener Quellen auf eine einfache Art und Weise miteinander zu kombinieren, indem diese miteinander vernetzt werden.

Durch den Einsatz festgelegter Ontologien und Vokabularien ist es möglich, die für das Semantic Web notwendigen Metadaten auszuzeichnen. Durch die Entwicklung einer Reihe von Standards durch das World Wide Web Consortium (W3C) wurde die Entwicklung des semantischen Webs stark beschleunigt, so dass es nunmehr teilweise in produktiven Systemen eingesetzt werden kann. Hierzu gehört beispielsweise eine Familie von Standards rund um das Resource Description Framework (RDF) [19]. Dies ist eine Empfehlung des W3C, die die Erstellung von Metadaten-Strukturen für im Web definierte Daten ermöglichen soll. Dadurch wird eine formale Beschreibung von Informationen über Objekte ermöglicht, welche im Kontext des Semantic Web Ressourcen genannt werden. Jene Ressourcen sind jeweils durch einen eindeutigen Bezeichner in Form eines Uniform Resource Identifier (URI) [20] identifizierbar. Diese ja schon durch die bisherige Verwendung im Internet bekannten URIs, haben im Bereich des Semantic Web die Besonderheit, dass ein URI nicht nur die Adresse einer Internetseite repräsentiert, sondern jede mögliche Ressource. Dabei ist es explizit nicht erforderlich, jedoch wünschenswert, dass dieser URI durch ein Programm wie einen Internet Browser erfolgreich aufgerufen werden kann. Die benötigte Information steckt dann nicht in der abrufbaren Ressource, sondern ist der URI selbst.

In RDF werden Aussagen über Ressourcen so definiert, dass dadurch ein Tripel entsteht. Dieses Tripel ist eine Sequenz aus den Elementen *Subjekt*, *Prädikat* und *Objekt*. Ein Objekt kann dabei wieder das Subjekt eines weiteren Tripels sein. Weiterhin besteht die Möglichkeit

eine Menge von Tripeln in einem Graphen zusammenzufassen. Das Subjekt beschreibt, über welche Ressource eine Aussage getroffen wird. Die konkrete Eigenschaft des Subjekts stellt dabei das Prädikat dar. Ein Objekt wiederum bildet das Argument für die durch das Prädikat beschriebene Eigenschaft. Während Subjekt und Prädikat immer ein URI sind, kann ein Objekt auch reine Daten beinhalten. In diesem Fall stellt es keine Verknüpfung zu einem anderen Subjekt dar und wird daher Literal genannt. Erfolgt eine grafische Darstellung, so ist es üblich, dass die URIs von Subjekt und Objekt in elliptischer Form dargestellt werden, während bei Literalen zur besseren Unterscheidung eine rechteckige Darstellung genutzt wird. Subjekt und Objekt bzw. Literal werden durch eine gerichtete Linie vom Subjekt weg verbunden, wie es das Beispiel in Abbildung 3 zeigt. Im Verlauf dieser Arbeit wird die Schreibweise in Form der Extensible Markup Language (XML) [21] für Beispiele verwendet.

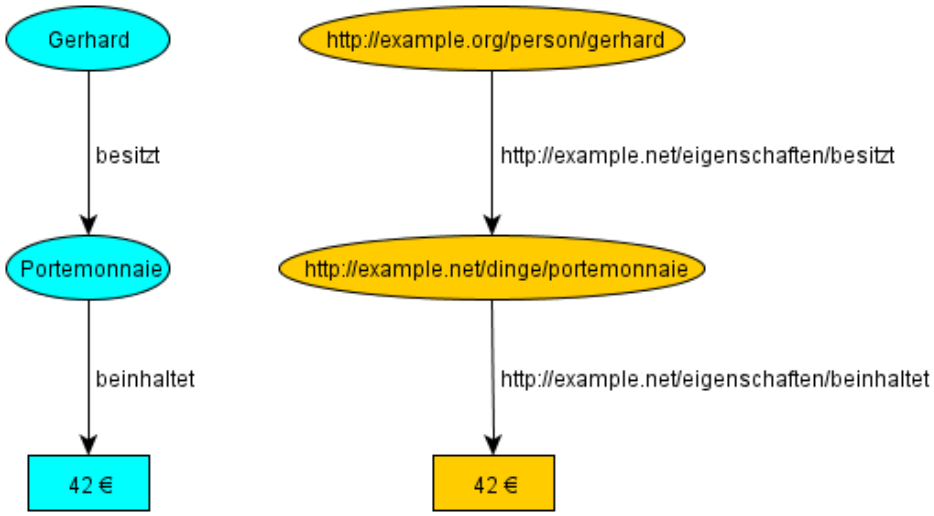


Abbildung 3: Beispiel RDF

Damit Rechner möglichst einfach nicht nur die Struktur, sondern auch die semantische Bedeutung der Daten zueinander verstehen zu können, um damit unter anderem implizites Wissen ermitteln zu können, werden Ontologien eingesetzt. Diese Ontologien beschreiben den Aufbau einer bestimmten Sache. Sie bestimmen also die mit der Ontologie verwendbaren Prädikate auch auf semantische Art und Weise oder erstellen Typen, so genannte Klassen, für die Subjekte bzw. Objekte. Weiterhin lässt sich eine Vielzahl von Beschränkungen definieren. So ist es möglich, dass einem Subjekt über ein bestimmtes Prädikat ein Objekt eines bestimmten Typs zugeordnet ist.

Da die stetige Verwendung von URIs oft eine erhöhte Redundanz aufgrund gleicher Bestandteile aufweist, ist es üblich XML-Präfixe als Kurzschreibweise einzusetzen. Dies folgt dem Vorgehen von XML, bei dem mit Hilfe eines kurzen Präfixes ein Namensraum definiert werden kann. Die in dieser Arbeit verwendeten Präfixe sind in Tabelle 1 notiert.

Präfix	**XML-Namensraum**	**Beschreibung**
dcterms	http://purl.org/dc/terms/	Konventionen für Metadaten der Dublin Core Metadata Initiative [22]
emo	http://www.w3.org/2009/10/emotionml	EmotionML [4]
emochat	urn:emotionchat:config:	Namensraum für Konfigurationen im Prototypen (Chatclient)

foaf	http://xmlns.com/foaf/0.1/	Beschreibungsvokabular Friend of a Friend (FOAF) [23]
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#	RDF [19]
smiley	http://www.smileyontology.com/ns#	Smiley Ontology [24]
xsd	http://www.w3.org/2001/XMLSchema#	XML-Schema [25]

Tabelle 1: Übersicht verwendeter Namensräume und Präfixe

3 Stand der Technik

Der weitere Verlauf der Arbeit beschäftigt sich mit der Betrachtung möglicher Kandidaten, die eine Lösung für die Problemstellung darstellen können. Es werden zuerst Anforderungen an die Problemlösung herausgearbeitet und aus diesen Bewertungskriterien abgeleitet. Im Anschluss werden mehrere verschiedene Szenarien näher untersucht. In einer sich anschließenden Evaluation wird analysiert, ob die gewählten Szenarien eine Lösung für die gegebene Problemstellung darstellen.

3.1 Anforderungen an die Problemlösung

Für die Lösung der Problemstellung ergeben sich vielfältige Anforderungen. Die Basis dafür bilden Anforderungen, die aus der Aufgabenstellung abgeleitet werden können. Diese werden ergänzt durch Anforderungen, die im Rahmen dieser Arbeit erkannte Probleme beim momentanen Umgang mit Emotionen in der textuellen Kommunikation lösen sollen. Aus den Anforderungen werden verschiedene Kriterien abgeleitet, die zur Bewertung möglicher Lösungen genutzt werden. Bei diesem Prozess kann es dazu kommen, dass von einer Anforderung mehrere Kriterien abgeleitet werden können, die sich ganz oder teilweise mit den Kriterien einer anderen Anforderung überschneiden. Zum Zweck einer besseren Bewertungsmöglichkeit werden die Kriterien in die Kategorien „sehr wichtig“, „wichtig“ und „empfehlenswert“ absteigend eingeteilt und priorisiert. Der Übersicht halber werden die Kriterien im Anschluss an die Aufstellung der Anforderungen gruppenweise zusammengefasst. Weiterhin wird für den Bewertungsprozess ein Bewertungsschema definiert.

3.1.1 Aufstellung von Anforderungen

Anforderung 1: Instant Messaging unterstützen
Die Aufgabenstellung spricht an, dass die Kommunikation via Instant Messaging an Bedeutung gewinnt. Daher ist es nötig, dass für eine erfolgreiche Lösung das Instant Messaging an sich unterstützt werden muss. Somit entsteht das erste Bewertungskriterium, dass die Lösung ein Protokoll zum Instant Messaging verwendet.

Anforderung 2: Soziale Netzwerke unterstützen
Weiterhin wird in der Aufgabenstellung auf die immer größer werdende Bedeutung sozialer Netzwerke verwiesen. Es ist somit wünschenswert, dass eine Anbindung an soziale Netzwerke durch die Lösung möglich ist. Daraus ergibt sich das Kriterium, dass die im Text versendeten Daten und speziell die Emotionen auch im Rahmen eines sozialen Netzwerkes verarbeitbar sein müssen.

Anforderung 3: Eindeutige Emotionalität herstellen
Der heutigen textuellen Nachrichtenübermittlung fehlt es an eindeutiger Emotionalität. Diesem Zustand muss entgegengewirkt werden. Daraus ergeben sich die Kriterien, dass

eine Emotion abgebildet werden können muss und dass diese Abbildung der Emotion eindeutig ist.

Anforderung 4: Erfassung und Interpretation der Emotion stärken
Bei der textuellen Kommunikation obliegt es dem Empfänger die Emotionalität einer Nachricht zu erfassen und zu bewerten. Kapitel 2.1 zeigt, wie das im Detail von statten geht. Es ist empfehlenswert, den Empfänger bei diesen Aufgaben zu unterstützen. Dies kann durch die schon in Anforderung 3 definierten Kriterien der eindeutigen Abbildung einer Emotion realisiert werden.

Anforderung 5: Übertragung emotionaler Aspekte in interkultureller Kommunikation verbessern
Verschiedene Kulturen deuten Elemente in der Kommunikation abweichend. Wie Kapitel 3.2.1.1 zeigt, ist dies auch bei emotionalen Bestandteilen der Kommunikation der Fall. Die mit diesem Aspekt zusammenhängenden Probleme verschärfen sich durch die immer globaler agierende Welt, da hierdurch vermehrt interkulturelle Kommunikation stattfindet. Beim Empfang emotionaler Aspekte in einer interkulturellen Kommunikation muss eine Lösung gefunden werden, die eigene Emotion dem Gegenüber verständlich zu machen. Hieraus leitet sich das Kriterium für die Lösung ab, in der die Emotion einer Nachricht in die Deutungsweise bzw. eine nach bekannten Mitteln erfassbare Darstellung einer anderen Kultur umgewandelt werden können muss.

Anforderung 6: Serialisierung und Übertragung von Emotionen ermöglichen
Als Ziel der Arbeit ist definiert, dass eine Lösung zur Serialisierung und Übertragung der Emotion gefunden werden soll. Abgeleitet daraus muss die Lösung das Kriterium erfüllen, Emotionen serialisieren zu können. Ein weiteres Kriterium besteht darin, dass es möglich ist diese serialisierten Emotionen zu übertragen.

Anforderung 7: Emotionen mehrerer Quellen im Kontext Instant Messaging anwenden
Als Ausgangslage werden verschiedene Quellen zur Eingabe der Emotionen definiert. Da diese im Rahmen bspw. des Instant Messaging übertragen werden sollen, müssen sie zusammengefasst werden, da sich eine Person während des Chats in einem konkreten emotionalen Zustand befindet. Daraus muss das Kriterium abgeleitet werden, dass die Problemlösung mehrere unterschiedliche Quellen für den Emotionsinput unterstützen muss. Ein weiteres Kriterium stellt die Zusammenfassung mehrerer Emotionen bspw. aus mehreren Quellen der Eingabe zu einer Gesamtemotion dar.

Anforderung 8: Darstellung der Emotion beim Empfänger unterstützen
Weitere Anforderungen an die Lösung ergeben sich aus der näheren Betrachtung des momentanen Umgangs mit Emotionen in der textuellen Kommunikation. Ein im Bereich des Instant Messagings und sozialer Netzwerke verbreitetes Phänomen ist die Darstellung von Emotionen durch Emoticons (Kapitel 3.2.1). Bei der Darstellung erfolgt dabei oft eine Transformation der Emotionsrepräsentation, also meist die automatische Umwandlung eines textbasierten Emoticons in eine grafische Darstellung. Den hierbei entstehenden fehlerhaften Umwandlungen muss entgegengewirkt werden. Daher ist das Kriterium der Trennung von Inhalt und Emotion zu berücksichtigen, welches für dieses Szenario eine mögliche Abhilfe darstellt.

Anforderung 9: Verwendung der Lösung in verschiedenen Programmen ermöglichen
Menschen verwenden heutzutage für die digitale Kommunikation unterschiedlichste Plattformen und Programme. Dies kann unter anderem anhand der Verbreitung von

verschiedenen Typen des Instant Messaging (vgl Kapitel 4.1.1) oder anhand von verschiedenen sozialen Netzwerken (siehe Kapitel 4.1.1) dargestellt werden. Eine breite und einfache Verwendung sowie eine optionale Weiterentwicklung der beschriebenen Lösung sind daher für die Gewinnung einer möglichst hohen Interoperabilität nötig. Hierbei spielen die Kriterien der Verwendung offener und standardisierter Protokolle sowie einer einhergehenden möglichen Veröffentlichung als Open-Source-Software eine Rolle.

Anforderung 10: Verwendung mit Software zur Emotionserkennung ermöglichen
Die Emotionserkennung wird bei der Verarbeitung von Emotionen in der Zukunft eine immer größere Rolle spielen. Kapitel 3.2.3 zeigt verschiedene Möglichkeiten für den Erkennungsprozess auf. Dabei nutzen viele emotionserkennende Applikationen keine global standardisierten Ausgabe- oder Übertragungsformate für die von ihnen erkannten Emotionen. Aufgrund dieser Bedeutung ist es empfehlenswert, die Interaktion mit heutiger und zukünftiger Software zur Emotionserkennung zu ermöglichen. Als Kriterien lassen sich hier eine dokumentierte Schnittstelle für den Emotionsinput sowie eine einfache Erweiterbarkeit der an dieser Schnittstelle angebundenen Inputvektoren definieren. Weiterhin ist bei der Nutzung der implementierten Lösung zu Beachten, dass eine hohe Akzeptanz des Nutzers für die Anwendung vorliegen soll. Dies ist nötig, da Software zur Emotionserkennung teilweise die aktive Mithilfe des Nutzers benötigt. Der Nutzer könnte jedoch bei der automatischen Verbreitung seiner Emotionen zurückhaltend sein, da, wie Kapitel 2.2 zeigt, die Emotion viel über seine Handlungsweise aussagen kann. Somit könnte ihn ein Nachteil z.B. bei Verhandlungen mit Geschäftspartnern entstehen. Das Vertrauen des Nutzers ist jedoch stark abhängig vom Kontext der eingesetzten Lösung, so dass hieraus kein Bewertungskriterium abgeleitet werden kann.

Anforderung 11: Zukünftige Emotionen unterstützen
Wie die Betrachtungen in den Kapiteln 3.2.1.4, 3.2.2.2 und 3.2.3 zeigen, sind Emotionen sehr komplex und vielfältig, so dass heutige Systeme zur Standardisierung von Emotionen nicht ausreichen. Weiterhin werden immer wieder neue emotionale Zustände durch gesellschaftliche Strömungen definiert. Dies muss bei der Lösung Beachtung finden, so dass es ein Kriterium darstellt, dass ein für die Emotionen verwendetes Vokabular erweiterbar sein muss.

Anforderung 12: Eigenarten in Emotionsverarbeitung des Nutzers berücksichtigen
Eine weitere Tatsache, die die Analyse in Kapitel 3.2.3 hervorgebracht hat, besagt, dass nicht nur kulturelle Unterschiede, sondern auch Einschränkungen des Menschen verantwortlich für seine bevorzugte Ausdrucksweise der Emotionen sind. Demzufolge wäre ein Kriterium der Lösung, dass sie die Emotionserfassung anhand einer nutzerspezifischen Konfiguration beeinflussbar macht.

3.1.2 Aufstellung eines Bewertungsschemas

Zur besseren Übersicht im weiteren Verlauf der Arbeit werden die von den Anforderungen abgeleiteten Bewertungskriterien in drei Gruppen eingeteilt und zusammengefasst. Weiterhin erfolgt eine Gewichtung nach Wichtigkeit. Kriterien, die unbedingt für eine erfolgreiche Implementierung eines Prototyps notwendig sind, werden dabei als sehr wichtig bewertet. Elemente die einem wirtschaftlichen Einsatz eines Prototyps dienen könnten und nicht unmittelbar für die erfolgreiche erarbeitete Lösung notwendig sind, werden allerdings nur mit empfehlenswert bewertet. Allen anderen Kriterien wird eine wichtige Gewichtung zugesprochen.

Zusammenspiel mit externen Systemen

- für Instant Messaging nutzbar (sehr wichtig)
- Möglichkeit der Anbindung an soziale Netzwerke (wichtig)
- Emotionserfassung unterstützen (empfehlenswert)
- Verwendung offener Protokolle (wichtig)
- Erweiterbarkeit der Inputvektoren (wichtig)

Implementierung

- mehrere Quellen für Emotionsinput (sehr wichtig)
- Trennung von Emotion und Inhalt (sehr wichtig)
- Nutzung von Standards (wichtig)
- Open Source frei verfüg- und nutzbar (wichtig)
- dokumentierte Schnittstellen (empfehlenswert)
- Abbildung einer Emotion (sehr wichtig)

Emotionen

- Eindeutigkeit der abgebildeten Emotion (sehr wichtig)
- Emotion in Deutungsweise einer anderen Kultur umwandeln (empfehlenswert)
- Serialisierung von Emotionen (sehr wichtig)
- Übertragung von Emotionen (sehr wichtig)
- Zusammenfassung mehrerer Emotionsrepräsentationen (wichtig)
- Vokabular für Emotionen erweiterbar (wichtig)

Um mögliche Lösungen anhand dieser aufgestellten Kriterien miteinander vergleichen zu können, ist die Definition eines geeigneten Bewertungsschemas notwendig. Die Basis bildet dabei eine Einordnung dahingehend, wie gut die entsprechende Lösung das Bewertungskriterium umgesetzt hat. Hierfür wird ein Bewertungsrahmen von „sehr gut“ über „gut“, „neutral“, „schlecht“ bis zu „sehr schlecht“ gemäß Tabelle 2 definiert.

Abkürzung	Bedeutung	Multiplikand
++	sehr gut	2
+	Gut	1
o	Neutral	0
-	Schlecht	-1
--	sehr schlecht	-2

Tabelle 2: Bewertungstabelle mit Zuordnung einer Berechnungsbasis

Aufgrund der unterschiedlichen Wichtigkeit der einzelnen Bewertungskriterien muss für einen einfachen Vergleich mehrerer Lösungsvarianten eine Berechnung durchgeführt werden, welche die Gewichtung und die entsprechende Güte der Entsprechung einbezieht. Hierfür wird eine einfache Multiplikation von Güte und Wichtigkeit vorgenommen. Dabei wird der Güte gemäß Tabelle 2 ein Multiplikand für die weitere Berechnung zugeordnet. Die Wichtigkeit des Kriteriums wird durch einen Multiplikator gemäß Tabelle 3 berücksichtigt.

Wichtigkeit	Multiplikator
sehr wichtig	3
wichtig	2
empfehlenswert	1

Tabelle 3: Zuweisung von Multiplikatoren zur Wichtigkeit der Kriterien

Damit auf eine einfache Art und Weise die Gesamtgüte verschiedener Lösungsvarianten miteinander verglichen werden kann, werden die Produkte der Bewertung der einzelnen Kriterien zu einer Summe addiert:

$$Gesamtgüte = \sum Güte_{Kriterium} * Wichtigkeit_{Kriterium}$$

3.2 Prüfung vorhandener Möglichkeiten der Emotionsübertragung

Im Kontext des Instant Messaging und sozialer Netzwerke steht die textuelle Kommunikation im Vordergrund. Für die Übertragung von Emotionen in dieser Form der Kommunikation ist eine Kombination aus Emoticons, eine meist zu gesichtsähnlichen Darstellungen kombinierte Zeichenfolge, ergänzt durch Elemente des Netzjargons, einer zumeist aus Akronymen bestehenden Sprachkultur, in der praktischen Anwendung beobachtbar. Allerdings kann die Emotion einer textuellen Kommunikation nicht nur durch Emoticons und ähnlichem repräsentiert werden, sondern auch durch Auszeichnungssprachen oder Ontologien, die für eine maschinelle Verarbeitung optimiert sind. Ein dritter Bereich, der die genaue Erfassung von Emotionen für eine Übertragung möglich macht, sind Technologien zur Erkennung und Klassifikation von Emotionen. Alle drei Varianten haben das Potenzial eine Lösung für das Problem der fehlenden eindeutigen Emotionalität in der textuellen Kommunikation darzustellen. Obwohl es weitere Möglichkeiten der Übertragung von Emotionen gibt, sollen diese drei vielversprechenden Varianten als mögliche Lösung überprüft werden.

3.2.1 Nutzung von Emoticons und Netzjargon

Wird in der virtuellen Welt kommuniziert, so sind meist die aus der Realwelt bekannten Mittel der nebenläufigen Emotionsübertragung bei Gesprächen nicht oder nur eingeschränkt verfügbar (vgl. Kapitel 1.1). Oft wird nur in Textform über Kommentare, während Chats oder mittels E-Mails kommuniziert. Die aus dem persönlichen Alltagsgespräch bekannten und automatisch angewandten Interpretationen emotionaler Gesprächsbestandteile über Elemente wie Mimik, Gestik oder Körpersprache ist so meist nicht anwendbar. Um diese wichtigen und gewohnten emotionalen Bestandteile gerade in der textbasierten Kommunikation wieder hinzuzufügen, haben sich mehrere Möglichkeiten entwickelt eigene Emotionen in einer Kurzform darzustellen.

Da Emotionen in ihrem Wesen sehr komplex und deren Ausdruck dadurch sehr schwierig ist, hat sich im Laufe der Jahre das Phänomen der Emoticons entwickelt. Diese meist gesichtsähnlichen Darstellungen sind ein gängiger Repräsentant von Emotionen in der textuellen Kommunikation. Durch eine getrennte historische Entwicklung, werden diese in westliche Emoticons (Smileys) und östliche Emoticons (Emojis) unterteilt (vgl. Kapitel 3.2.1.1). Ergänzt werden die Emoticons als Übermittler von Emotionen durch spezielle Elemente des Netzjargons, einer zumeist aus Akronymen bestehenden Sprachkultur, die vorwiegend in der chatbasierten digitalen und meist privaten Kommunikation anzufinden ist.

Ein **Emoticon** ist ein „Symbol, das in der elektron. Korrespondenz (E-Mail u.a.) einen Gefühlszustand bzw. eine Mimik auszudrücken vermag“ [11]. Sie werden daher in der textbasierten Kommunikation oft mit Emotionen gleichgesetzt oder zumindest eng mit einer realen Emotion verknüpft. Die Bezeichnung Emoticon stellt dabei eine Wortkreuzung aus den englischsprachigen Begriffen **emot**ion (Emotion) und **icon** (Bild) dar. Emoticons können

dabei als einzelne Zeichen, Zeichenfolgen oder auch mittels Bildern dargestellt werden. Den Kern der Emoticons bilden gesichtsähnliche Darstellungen, die die menschliche emotionale Mimik widerspiegeln sollen und somit bspw. Freude oder Traurigkeit darstellen [26]. Die Darstellung und Interpretation fällt damit zum großen Teil auf die gewohnte alltägliche emotionale Interpretation von Gestik und Mimik zurück und unterliegt somit kulturellen Unterschieden. Weiterhin kann die Interpretation je nach Kontext, in dem das Emoticon verwendet wird, unterschiedlich sein. Erweitert werden die an Gesichtern angelehnten Emoticons durch andere Symbole aus der Realwelt wie bspw. eine Rose, ein Cocktailglas oder Pizza. Diese sind in ihrer emotionalen Bedeutung meist abgeschwächt, stellen aber oft eine Handlungstendenz dar. Weiterhin ist zu beobachten, dass Abkürzungen aus dem Netzjargon teilweise auch emotionale Aspekte widerspiegeln. Primär bei der zeichenkettenbasierten Darstellung von Emoticons, aber auch bei der davon abgeleiteten Darstellung als Bilderzeugnis, kann eine Unterteilung in westliche und östliche Emoticons vorgenommen werden. Diese unterscheiden sich in Aussehen und Interpretationsmethodik. Dass der Emotionsdarstellung durch Emoticons kommerzielle Wichtigkeit zugerechnet wird, können die Markenanträge der Firma Despair Inc. für das traurige Smiley [27] und des russischen Geschäftsmannes Oleg Teterin für das Zwinkersmiley belegen [28].

Das Wesen eines Emoticons wird von drei aufeinander aufbauenden Ebenen beeinflusst (vgl. [29]). Die Basis bildet dabei die eigentliche Emotion oder der Gefühlszustand, welcher durch das Emoticon ausgedrückt werden soll. Auf diesem aufbauend wird die Struktur des Emoticons bestimmt. In diesem Bereich fallen also der Gesichtsausdruck des Emoticons, aber auch dessen Aktionen und Objekte, mit denen das Emoticon interagiert. Schlussendlich wird daraus die eigentliche Repräsentation des Emoticons durch Farben, grafische Abbildungen oder Animationen generiert. Findet eine Interpretation des Emoticons statt, um dessen dargestellten emotionalen Zustand erkennen zu können, erfolgt eine Interpretation in umgekehrt chronologischer Reihenfolge. Zuerst wird die visuelle Erscheinung des Emoticons wahrgenommen. Anschließend werden seine Bestandteile strukturell separiert und zusammen mit dem visuellen Erscheinungsbild interpretiert, damit schlussendlich die eigentlich übertragene Emotion erkannt werden kann.

Die Darstellung von Emoticons in der textuellen Kommunikation auf verschiede Art und Weise realisiert werden. Die historische Basis stellen dabei zeichenkettenbasierte Emoticons dar. Durch eine Darstellung als Schriftzeichen erhielten zeichenbasierte Repräsentationen Einzug in gängige Zeichensätze. Der Kreativität des Menschen ist es zu verdanken, dass diese dann wiederum in grafischen Repräsentationen erneut eine Wandlung erfahren.

3.2.1.1 Emoticons Zeichenkettenbasiert

Zeichenkettenbasierte Emoticons sind aus mehreren einzelnen Zeichen, zumeist Sonderzeichen, zusammengesetzt und bilden somit als eine Einheit eine textbasierte Repräsentation eines Emoticons. Der Übergang zwischen Emoticons und künstlerischen Gebilden aus verschiedenen Zeichen, dem so genannten ASCII-Art[3], geschieht fließend. Im Groben können bei dieser Art der Emoticons zwei relativ parallel entstandene Gruppen voneinander unterschieden werden. Auf der einen Seite sind die im amerikanischen Raum vorherrschenden und entstandenen Smileys, die als Emoticons der westlichen Welt sich rasch von ihren Ursprüngen in Nordamerika u.a. auf den europäischen Kontinent ausbreiten

[3] ASCII-Art ist eine zumeist im digitalen Kontext anzutreffende Kunstrichtung, bei der mit Hilfe einer Festbreitenschrift bildliche Abbildungen dargestellt werden.

konnten. Demgegenüber steht die Entwicklung der Emojis im asiatischen Raum als Emoticons der östlichen Welt, welche sich in Aussehen und Interpretationsweise von den Smileys unterscheiden.

Westliche Emoticons, welche auch **Smileys** genannt werden, sind eine ursprünglich im amerikanischen Bereich entstandene textbasierte Form der Emotionsdarstellung, welche sich später weiter vor allem mit dem Internet verbreitet hat. Eine einfache Erkennung und Interpretation der Smileys ist realisierbar, wenn der Kopf um 90° zumeist nach links geneigt wird. Somit kann dann ein gesichtsähnliches Abbild mit entsprechender Mimik erkannt werden. Eine weniger häufige Verwendung finden so genannte linkshändische Smileys, bei denen der Kopf 90° nach rechts geneigt werden muss, um die Mimik des gesichtsähnlichen Gebildes einfach erkennen zu können [30].

Die Interpretation von Smileys erfolgt unter Aspekten der emotionalen Interpretation einer Mimik aus der Realwelt zumeist primär durch die Mundregion des gesichtsähnlichen Gebildes, gefolgt von der Augenregion. Die Nase sowie andere möglicherweise vorkommende Elemente spielen eine untergeordnete, zumeist sogar eine rein dekorative Rolle und werden bei Kurzschreibweisen oft weggelassen [31].

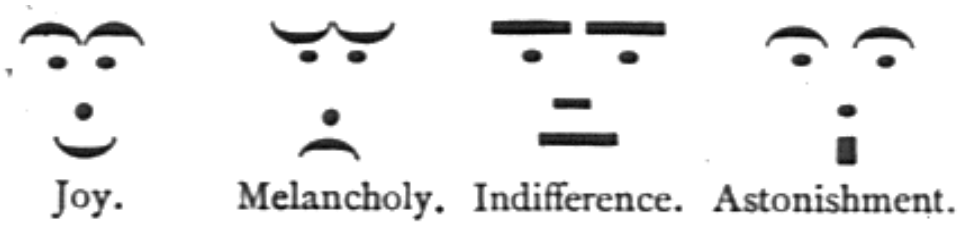
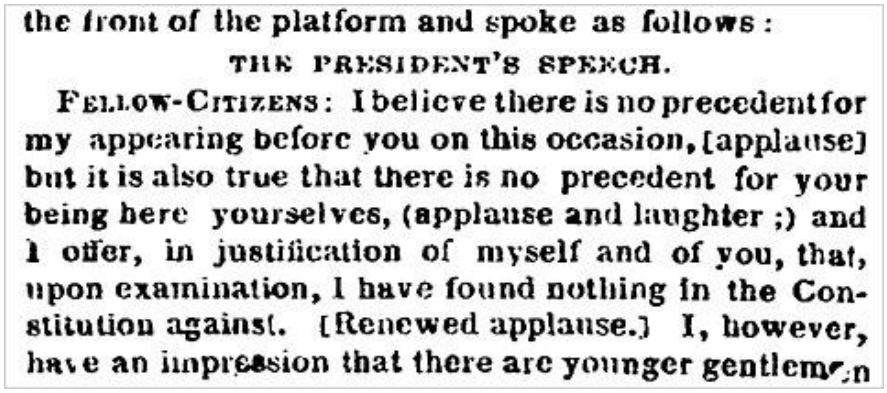


Abbildung 4: Erste Emoticons aus dem Satiremagazin Puck[4]

Die erste Veröffentlichung einer Abwandlung von Smileys, wie sie in Abbildung 2 ersichtlich ist, ist im US-Satiremagazin „Puck“ vom 30. März 1881 zu finden. Ob eine als Zwinkersmiley interpretierbare Zeichenkette in einer in Abbildung 5 dargestellten Rede von Abraham Lincoln aus dem Jahre 1862 einen Smiley darstellt oder nicht, ergibt in der Wissenschaft keine Einigkeit [32].


the front of the platform and spoke as follows:

THE PRESIDENT'S SPEECH.

FELLOW-CITIZENS: I believe there is no precedent for my appearing before you on this occasion, [applause] but it is also true that there is no precedent for your being here yourselves, (applause and laughter;) and I offer, in justification of myself and of you, that, upon examination, I have found nothing in the Constitution against. [Renewed applause.] I, however, have an impression that there are younger gentlemen

Abbildung 5: Emoticon in einer Rede von Abraham Lincoln[5]

Die Entstehung von auf der Seite liegenden Smileys während einer digitalen Konversation ist auf Scott E. Fahlman von der Carnegie Mellon University zurückzuführen. Der Definition von

[4] Entnommen entsprechender Zeitschriftenseite in den Wikimedia Commons [109]
[5] Foto aus Artikel der New York Times [32]

Smileys ging voraus, dass ironisch gemeinte Beiträge in Newsgroups[6] manchmal schwer von nicht ironischen Beiträgen unterscheidbar waren. Primär wurde in der Gruppe überlegt, ob über spezielle Symbole im Titel des Beitrages eine etwaige Klassifizierung vorgenommen werden kann. So wurde postuliert, dass ein * im Titel einen guten Witz repräsentieren solle, während % einen schlechten Witz darstelle. Die Kombination *% sollte demnach einen Witz darstellen, der so schlecht ist, dass er schon wieder als lustig erscheint. Auch für den Betrachter lustig aussehende Symbole wie & und # waren als Indikator im Gespräch. Für eine Frage wurde ein einfaches Fragezeichen definiert. Am 19. September des Jahres 1982 um 11:44 amerikanischer Zeit schlug Fahlman vor, dass die Phrase „:-)“ Nachrichten mit lustigem Inhalte kennzeichnen sollten, während Dinge, die nicht lustig gemeint waren, durch ein „:-(„ dargestellt werden sollten. Somit war die initiale Idee hinter diesen Emoticons nicht die heutige Übertragung der eigenen Emotion, sondern eher eine Möglichkeit für die Unterscheidung ironisch gemeinter Phrasen vom Rest. Als Antwort darauf schlug Jeff Shrager am 19. September 1982 18:56 eine Taxonomie für humorvolle Nachrichten vor. Sein komplizierter Vorschlag konnte sich jedoch nicht gegen das einfache und intuitive System von Fahlman durchsetzen. Letzteres schaffte es auch, sich gegen andere bis dahin geltende Praktiken, wie die Phrase „__/“ des Benutzers „Gandalf vax“ für ein universelles Lachen, durchsetzen. In der folgenden Zeit zeigte sich ein exploratives Verhalten der Teilnehmer der Newsgroup mit dem neuen System. So wurden schnell die so genannten Linkshändersmileys „)-:“ genauso genutzt wie das von Fahlman definierte Rechtshändersmiley „:-)''. Bald darauf folgten andere Smileys wie „:-|“ für neutrale Informationen und „:-O“ für überraschende Tatsachen. Schon einen Monat später wurden außerhalb der Carnegie Mellon University Emoticons beobachtet, welche sich nicht mehr nur auf Emotionen beschränkten, sondern ein Themenbezug zum Inhalt darstellten (bspw. Fahrrad, NuclearWar, nur für Frauen). Somit fand schon in den frühen Anfängen der Nutzung von Smileys eine Fortbewegung von der emotionalen Bedeutung der Smileys hin zu einer Beschreibung von Inhalt statt, die die Aktion des Empfängers beeinflussen sollte [30].

Die neben den Smileys existierenden **Östliche Emoticons**, auch **Emojis** genannt, haben ihren Ursprung im asiatischen Raum, wobei hier der Japanische als Vorreiter dieser Art von Emoticons dient. Aufgrund der auf dem asiatischen Kontinent vorherrschenden Schriftzeichen (Kanji) und deren Position im Unicode[7] werden Emojis zumeist aus Multibytezeichen generiert. Die Bezeichnung Emoji ist abgeleitet vom japanischen **e** (Bild) und **moji** (Brief). Emojis werden normal im Text notiert und sind somit in Leserichtung bei normaler Kopfhaltung als Emoticon erkennbar. Bei der Interpretation der Emojis ist vor allem die japanische Kultur zu beachten. Während die Japaner an sich eine emotionale Zurückhaltung in der direkten, persönlichen Kommunikation pflegen, werden in der digitalen Kommunikation besonders viele Emoticons zu deren Ausdruck verwendet. Somit entwickelt sich auch schnell ein weitreichendes Repertoire an möglichen Emojis, welche teilweise sogar genderspezifisch unterschieden werden [31].

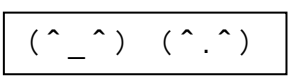


Abbildung 6: Glückliche Emojis (links männlich, rechts weiblich)

[6] Eine Newsgroup ist eine virtuelle Nachrichtengruppe, in der Nachrichten zu einem bestimmten Thema ausgetauscht werden.

[7] Bereich 4E00-9FCF [108]

Emojis bekommen ihre Bedeutung primär durch die Interpretation der Augen des entstehenden Gesichtes. Dieses wird unterstützt durch optional dargestellte Hände und Nasen. Evtl. vorkommende Münder dienen nicht primär der Interpretation und werden nur zur Verstärkung herangezogen, indem bspw. der Mund stärk verlängert wird. Runde Klammern werden zumeist als Abgrenzung des Gesichts genutzt. Da diesen jedoch bei der Interpretation des Emojis keine große Bedeutung zukommt, werden sie bei Kurzschreibweisen weggelassen [33].

Gerade im Bereich mobiler Kommunikation ist der Einsatz von Emojis vor allem im japanischen Territorium enorm[8]. Daher schlossen sich hier die größten Handyhersteller unter Führung von Apple zusammen mit dem Softwarekonzern Google und versuchen gemeinsam eine Integration von Emojis im Unicode zu erwirken [34].

Die Entstehungsgeschichte von Emojis ähnelt der von Smileys und ist auch durch Mitschnitte aus Newsgroups belegt. Das wohl erste Vorkommen des fröhlichen Emojis „(^_^)“ wurde im Juni 1986 beobachtet, als der Nutzer Wakabayashi, der auch unter den Namen Wakan I bekannt ist, diese Zeichenkonstruktion nutzte. Damit beginnt die Entwicklung der Emojis später als die der Smileys. Die erste Exploration war danach in Briefen von High-School-Studentinnen beobachtbar, die die Zeichenkette „(^O^)“ mit dem Lachen eines Koalas beschrieben. Zwischen Mai und Juli 1988 wurde es durch Tatsuo Kondo zum ersten Mal im digitalen Kontext genutzt. Schon damals wurde erkannt, dass die Bedeutung der Emojis nicht immer komplett klar ist. Die grobe Interpretationsrichtung war jedoch aus dem Realleben bekannt, so dass die Menschen damit schnell umgehen lernten. Somit gab es schon 1992 eine ganze Fülle verschiedener Emojis, die die Anzahl der Smileys sicher übertraf [35].

Im direkten **Vergleich von Smileys und Emojis** sind viele Unterschiede nicht nur in der Darstellung, sondern gerade auch in der Interpretation zu erkennen. Die Unterschiede der Darstellung belaufen sich darauf, dass zum Lesen der Smileys der Kopf um 90° gedreht werden muss, während bei Emojis eine normale Kopfhaltung beibehalten werden kann. Weiterhin kommen aufgrund der im jeweiligen Kulturkreis vorherrschenden Sprache unterschiedliche Schriftzeichen und somit in der digitalen Welt unterschiedliche Zeichenkodierungen zum Einsatz. Daraus folgt, dass im jeweils ursprünglichen Kulturkreis Smileys mit 1byte pro Zeichen und Emojis mit 2byte pro Zeichen gespeichert und übertragen werden. Bei der Interpretation sind die Unterschiede weitreichender. Während bei Smileys primär der Mund ergänzt durch die Augen zur Interpretation der Emotion zu Rate gezogen wird und die Nase nur einen dekorativen Charakter aufweist, tragen bei Emojis die Augen unterstützt durch Nase und evtl. vorkommende Arme hauptsächlich die Emotion [31]. Die Münder und Gesichtsbegrenzungen haben bei Emojis eher dekorativen Charakter und werden bei Kurzschreibweisen oft weggelassen. Wie Tabelle 4 zeigt, ist bei Smileys mitunter eine Mehrfachbelegung („:-o“ für erschrocken, schreiend) oder eine Verwechselungsgefahr („:-o“ vs. „:-0“) zu beobachten. Dies wird bei Emojis durch die wesentlich höhere Vielfältigkeit vermieden [36]. Somit ist es nicht verwunderlich, dass die Bedeutungen von Smileys je nach kulturellem Kreis viel weiter voneinander abweichen, als dies bei Emojis der Fall ist.

Weiterhin ist zu beobachten, dass es heutzutage zu einer Vermischung von Smileys und Emojis in der Anwendung kommt. Dies kann dem globalen Vorhandensein des Mediums

[8] Siehe Statistiken in http://whatjapanthinks.com/2009/02/09/why-use-japanese-emoticons/ (Abruf 23.07.2012)

Internet und der damit geförderte interkulturellen Kommunikation zugeschrieben werden. Durch die unterschiedliche Darstellung kann es gerade bei fremdsprachigen Texten der Fall sein, dass ein Emoticon des nicht eigenen Kulturraumes nicht als solches erkannt wird. Auch ist es der Fall, dass bei der Interpretation der Bedeutung des Emoticons die aus dem eigenen Kulturkreis bekannte Interpretationsweise von Smileys bzw. Emojis zur Anwendung kommt. Somit kann es jedoch bei Emoticons des anderen Kulturraumes zu einer Informationsverwischung kommen, da der Informationsgehalt einzelner Bestandteile des Emoticons falsch gewichtet wird oder gar eine abweichende Bedeutung interpretiert wird. Daher kann es hier zu Verlust der Klarheit der Information oder gar zu Missverständnissen und Fehlinterpretationen kommen.

Westliche Emoticons (Smileys)		**Östliche Emoticons (Emojis)**	
Emoticon	**Bedeutung**	**Emoticon**	**Bedeutung**
:-)	freundlich, fröhlich, lachend	(^ _ ^) (^ . ^)	lächeln männlich lächeln weiblich
:-(	Traurig, schlecht befindlich	(>_<)	traurig
;-)	ironisch, nicht erst gemeint, Augenzwinkern	(x_x ;)	jm. meiden
:-o	erschrocken, schreiend, erstaunt	(¯ ^ ¯)	grinsen
:-0	schreiend, sprechaktiv	＼ (^ ^) ／	ich bin erfreut
:-D	Lachen, sehr fröhlich sein	(´ ∀ `)	lachen
:-P	jm. Zunge zeigen / Zunge rausstrecken	Σ (0_0 ;)	überrascht sein

Tabelle 4: Gegenüberstellung Smileys und Emojis[9]

3.2.1.2 Emoticons Zeichenbasiert

Im Laufe der Entwicklung der elektronischen Datenverarbeitung, erfuhren die Smileys eine Metamorphose. Schon im Jahre 1984, also kurz nachdem die Nutzung von Smileys in den Newsgroups begann, definierte IBM auf der Codepage 437 [37] in den ersten beiden Positionen zwei lachende Emoticons, eines in ungedeckter Farbe an Position 1 (☺) und eines in gedeckter Farbe an Position 2 (☻).

Durch die Fortentwicklung der Technik und der damit verbundenen Durchsetzung von Unicode als Standardzeichensatz[10] spielt die Codepage 437 heute eine wesentlich weniger bedeutende Rolle. Die Emoticons wurden aber in den Unicode übernommen [38] und erweitert, so dass zu dem ungedeckten glücklichen Emoticon an Position 263A und seiner Entsprechung als gedecktes Emoticon an Position 263B nun erstmals auch ein ungedecktes trauriges Emoticon an Position 2639 zu finden ist. Mit Unicode-Version 6.0 wurde im Oktober 2010 der Bereich 1F600-1F64F veröffentlicht, welcher speziell für Emoticons reserviert ist [39]. Somit kamen zu diesem Zeitpunkt 63 farblich ungedeckte Emoticons, von denen einige in Abbildung 7 dargestellt sind, als standardisierte Zeichen im Unicode hinzu. Die dazugehörige Beschreibung ordnet den Bildnissen allerdings keine Emotionen zu, sondern beschreibt bspw. mit „GRINNING FACE WITH SMILING EYES“ wie üblich, was mit dem Zeichen dargestellt wird.

[9] Inhalte entstanden durch Vergleich in der Nutzung privater Kommunikation sowie Befragungen in Alltagsgesprächen, ergänzt durch http://office.microsoft.com/ja-jp/support/HA010103000.aspx (Abruf 08.09.2012)

[10] Vgl. Statistiken im Google Blog unter http://googleblog.blogspot.de/2010/01/unicode-nearing-50-of-web.html (Abruf 23.07.2012)

Abbildung 7: Emoticons im Unicode

3.2.1.3 Grafische Gestaltung von Emoticons

Zur Unterstützung der Interpretation bekannter und unbekannter Emoticons sowie um die Qualität der Darstellung von Emoticons zu verbessern, werden vor allem in Foren und Chats zeichenkettenbasierte Emoticons in eine grafische Repräsentation analog Abbildung 8 umgewandelt. Die Entwicklung grafischer Darstellungen von Emoticons hat ihren Ursprung in der künstlerischen Darstellung gesichtsähnlicher Emoticons außerhalb des digitalen Kontextes. Durch die gewonnenen Freiheiten aufgrund der grafischen Darstellung kommt es zu vielfältigen Anpassungen des Aussehens der Emoticons. Diese können somit optimiert zum grafischen Gesamtbild der Umgebung, in der sie auftreten, angepasst werden. Auch animierte Darstellungen erfreuen sich gerade in der chatbasierten Kommunikation einer großen Beliebtheit. Durch die einfachere Interpretation können mehr verschiedene Emoticons genutzt werden, die Gefühle oder Handlungstendenzen darstellen. Gleichzeitig können sie auch bei Unbekanntheit vom Nutzer leichter interpretiert werden, als dies bei zeichen- und zeichenkettenbasierten Emoticons der Fall ist. Somit gewinnt vor allem der nicht auf Gesichtern basierende Bereich der Emoticons an Bedeutung.

Während bei der Nutzung von zeichenkettenbasierten Emoticons in Chat-Umgebung zumeist zu beobachten ist, dass sie bei der Darstellung in eine grafische Repräsentation umgewandelt werden, erfolgt die Übertragung in der Nachricht vom Sender zum Empfänger jedoch meist in ihrer ursprünglichen Textform.

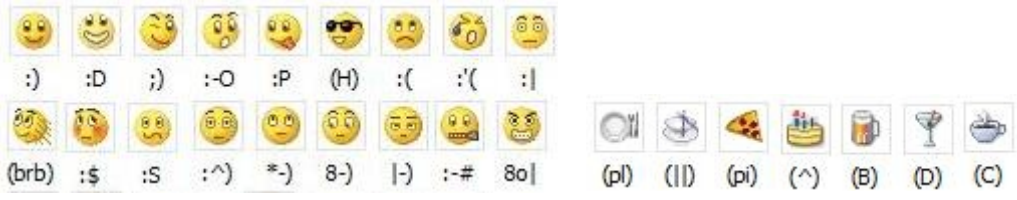


Abbildung 8: Umwandlung Emoticons MSN-Messager – Bereich Gesichter und Nahrung

Vor allem in Foren ist beobachtbar, dass gesichtsähnliche Emoticons ihre Bedeutung als reine Repräsentation der Emotion oft einbüßen und mehr Handlungstendenzen oder gar Meinungen repräsentieren wie dies in Abbildung 9 ersichtlich ist.

Abbildung 9: Emoticon mit Schild[11]

3.2.1.4 Netzjargon

Auch in Foren verbreitet ist der Netzjargon, eine von Abkürzungen geprägte Sprachkultur, die auch Chats vorzufinden ist. Mit Hilfe dieser Sprachweise können Textinhalten in knapper Form Zustands- und Gefühlsäußerungen hinzugefügt werden. Die Abkürzungen beziehen sich nicht nur auf Wörter, sondern auf ganze Sätze und können daher auch für sich eine Bedeutung aufweisen. Zur Verstärkung der Bedeutung können die Akronyme in Großbuchstaben notiert werden. Emotionale Äußerungen werden häufig mit Asterisken umschlossen, um ihre Trennung vom eigentlichen Inhalt darzustellen [40]. Teilweise werden solche umschlossenen Zeichenketten in Instant Messagern wieder in Emoticons umgewandelt.

Akronym	Bedeutung	Erklärung	Art
lol	Laughing Out Loud	Lautes Lachen	Handlungstendenz, Emotion
rofl	Rolling on (the) floor laughing	Lachend auf dem Boden wälzen	Handlungstendenz, Emotion
ily	I love you	Ausdruck der Liebe	Emotion
g	Grinsen	Ausdruck der Heiterkeit	Handlungstendenz
gg	Großes Grinsen	Gesteigerter Ausdruck der Heiterkeit	Handlungstendenz
hdl	Hab‘ dich lieb	Ausdruck der Zuneigung	Emotion

Tabelle 5: Beispiele für Akronyme im Netzjargon[12]

Wie in Tabelle 5 ersichtlich ist, sind die emotionalen Wendungen im Netzjargon feingranular, so dass mehrere Steigerungen der Emotion erreicht werden können. Dies ist im deutschen Sprachraum am Akronym „hdl“ erkennbar. Eine Diskussion im Bekanntenkreis ergab dabei die mögliche Steigerung hdl (Hab‘ dich lieb) → hddl (Hab‘ dich doll lieb) → hdgdl (Hab‘ dich ganz doll lieb) → ld (Lieb‘ dich) → ild (Ich liebe dich). Bei dieser Phrase ist auch häufiger beobachtbar, dass die g für „ganz“ und d für „doll“ vermehrt werden, um diesen Ausdruck weiter zu steigern. Bei diesem Akronym ist zudem durch eigenen Erfahrungsschatz erkennbar, dass dies primär von weiblichen Personen genutzt wird. Dies zeigt, dass bei der Nutzung von Netzjargon teilweise eine genderspezifische Nutzung erfolgt.

Noch mehr als bei den Emoticons kann bei den Akronymen in Netzjargon festgestellt werden, dass diese oft national verschieden sind. Die Basis bilden hier zwar einige weitläufig bekannte englischsprachige Abkürzungen, jedoch werden diese, wie bei dem Beispiel „lol“, durch Elemente des eigenen Sprachraumes oder onomatopoetische Entsprechungen ersetzt. So ist „*g*“ im dänischen eine Abkürzung für das Wort „Griner“, welches in der

[11] Bild entnommen aus http://www.allmystery.de/blogs/the.smoger/viele_neue_smileys (Abruf 23.07.2012)

[12] Ausgewählte Beispiele entnommen aus http://de.wikipedia.org/wiki/Liste_von_Abkürzungen_(Netzjargon) (Abruf 08.09.2012)

Landessprache mit „Lachen“ übersetzt wird [41]. Die Lautmalerei kann am koreanischen 하하 nachvollzogen werden, welches "haha" ausgesprochen wird.

Bei der grafischen Gestaltung von Emoticons können Kombinationen aus Emoticon und Akronymen des Netzjargons auftreten, so dass die Intensität der Bedeutung gesteigert wird. Das Problem hierbei ist, dass sowohl die korrekte Interpretation des Emoticons als auch des Netzjargons vom Betrachter erlernt sein muss.

Abbildung 10: Kombination Emoticon mit Netzjargon[13]

3.2.1.5 Evolution

Wie die Kombination von Emoticon und Netzjargon zeigt, unterliegen die Darstellungen von Emoticons einem ständigen Wandel. Dabei gilt der Grundsatz: „Die Innovationskraft und Kreativität des Menschen ist unerschöpflich“[14]. Das kreative Schaffen des Menschen führt dazu, dass immer wieder neue Emoticons und Elemente des Netzjargons entwickelt werden. Durch die Fülle an definierten Zeichen in modernen Zeichensätzen wie im Unicode wird die Zweckentfremdung von Zeichen für Emoticons weiter unterstützt. Diese folgen, wie in Abbildung 11 zu sehen ist, weniger dem Vorbild von Smileys, wie das z.B. beim Smiley „#:-)“ der Fall ist, sondern lehnen sich oft an das Aussehen der Emojis an. So entstehen größtenteils sowohl gesichtsähnliche Darstellungen, die aus nicht ASCII-konformen Zeichen zusammengesetzt sind, wie auch komplexe Gebilde die auch kombinatorische Zeichen im Unicode ausnutzen. Durch Unbekanntheit der Emoticons beim Empfänger und dessen ungewissem Erfahrungsschatz, ist die Interpretation des neuen Emoticons oft ungenau. Hier kommt es in diesem Fall einfach zu Missverständnissen oder einer Informationsverwischung, bis dem Emoticon in dem kulturellen Kreis durch einen gemeinsamen Lernprozess eine gemeinsame Bedeutung und Interpretation zugeordnet wird. Bei zeichenbasierten Emoticons findet durch den technisch eingeschränkten Zeichenvorrat in der Regel keine Evolution statt.

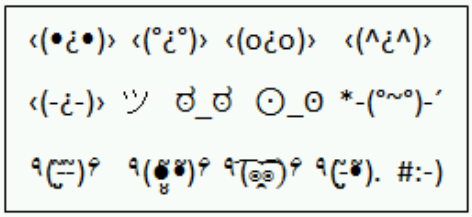


Abbildung 11: Verschiedene Evolutionen zeichenkettenbasierter Emoticons

3.2.1.6 Probleme bei der Verwendung

Aufgrund der Tatsache, dass Emoticons nicht durch eine Institution in Aussehen, Verwendung und Bedeutung standardisiert sind, ist eine stets subjektive Interpretation des Emoticons nötig. Diese Interpretation ist abhängig von den kulturellen Erfahrungen des Senders und des Empfängers sowie dem Kontext, in welchem das Emoticon angewandt wird. Wie in den Kapiteln 3.2.1.1 und 3.2.1.4 beschrieben ist, sind sowohl die Interpretationsweisen von Smileys und Emojis als auch die des Netzjargons verschiedener

[13] Bild entnommen aus http://radioaktivchat.com:5160/radioaktiv/help/smileys.html?Page=2 (Abruf 24.07.2012)
[14] MANAGERSEMINARE NR. 15, April 1994, managerSeminare Verlags GmbH, Bonn

Kulturkreise unterschiedlich. Kommunizieren nun zwei Personen unterschiedlicher Kulturkreise unter Zuhilfenahme von Emoticons oder Netzjargon miteinander, so ist die Wahrscheinlichkeit hoch, dass es zu Missverständnissen aufgrund nicht funktionierender Metakommunikation kommt. Diese rühren daher, dass die Emoticons oder Elemente des Netzjargons des Gegenübers unbekannt sein können, so dass diese ohne eigenen Erfahrungsschatz in dem jeweiligen Kontext interpretiert werden müssen. Die Interpretation erfolgt dabei zumeist nach den gewohnten Interpretationsregeln des eigenen Kulturkreises. Gleiches Problem tritt auf, wenn Kontakt zu völlig neuartigen Emoticons oder Akronymen hergestellt wird, was gerade bei der grafischen Darstellung von Emoticons auf Webseiten oft der Fall ist. Durch viele Missverständnisse ist daher die Anzahl verwendeter Emoticons, die in ihrer Bedeutung unterschiedlich sind, sehr gering[15]. Eine Abhilfe hierfür können auch nicht die zeichenbasierten Emoticons darstellen, da die menschlichen Emotionen wesentlich facettenreicher und vielfältiger sind, als es die momentanen Standards bspw. im Unicode abbilden. Zudem ist der Übergang zwischen verschiedenen Emotionen fließend und stark subjektiv geprägt, was eine korrekte Einordnung der momentanen Emotion sehr schwierig macht.

Ein weiteres Problem stellt die unterschiedliche Verwendung derselben Emoticons in verschiedenen gesellschaftlichen Gruppen dar, da sich gerade bei Smileys mitunter die Bedeutung der einzelnen Emoticons überschneidet. Um dieses Problem in Foren zu lösen, gibt es in einigen Foren Einträge, die die im Forum genutzten Emoticons mit ihrer Bedeutung definieren[16]. Die den Emoticons somit zugewiesende Bedeutung ist jedoch zwischen verschiedenen Plattformen nicht kongruent[17]. Durch die Überlappung verschiedener Erfahrungen mit Emoticons, kann es wie beim Emoticon „:-o“ vorkommen, dass dies mit mehreren weit voneinander unterschiedlichen Bedeutungen bekannt ist. Dieses wird je nach Quelle und/oder Kontext als erschrocken, schreiend oder erstaunt interpretiert. Weiterhin kann es selbst bei in der Bedeutung gleich definierten Emoticons zu einer unterschiedlichen Darstellung der Zeichen kommen. Dies ist beispielsweise bei neuen Erweiterungen des Unicodes der Fall, die die Systeme anfangs unterschiedlich gut oder unvollständig interpretieren können.

Im Bereich der nicht-gesichtsähnlichen Emoticons, welche meist eine schwache emotionale Komponente besitzen, dafür aber häufig Handlungstendenzen aufzeigen, kann ein weiteres Problem mit der Kontextabhängigkeit von Emoticons nachvollzogen werden. Wird das Bild einer Pizza als Emoticon übertragen, so kann dies vom Empfänger auf mehreren Wegen interpretiert werden:

- Der Sender ist hungrig
- Der Sender hat Appetit auf Pizza
- Der Sender möchte dem Empfänger symbolisieren, dass er etwas zu Essen schickt, damit es dem Empfänger besser geht
- Der Sender möchte einen Appell an den Empfänger übermitteln oder ihn fragen, ob gemeinsam (Pizza) gegessen werden soll
- Der Sender kann ganz allgemein die wirkliche Bedeutung von Pizza gemeint haben

[15] Vgl. Statistik verwendeter Emoticons auf http://www.allmystery.de/blogs/dns/allmystery_smiley_statistik (Abruf 23.07.2012). Bei 5.450.618 Beiträgen 493.287mal „:-D“, 398.809mal „;-)“, 271.477mal „:-)“, 44.029mal „:-(„, 20.565mal „:-|“ sowie stetig abnehmende Anzahl anderer Emoticons.

[16] Beispielseite für die Arcor-Foren unter http://www.arcor.de/tp/foren/smileys.php (Abruf 23.07.2012)

[17] Vgl. Arcor-Foren und http://www.vtxnet.ch/community/chat/smilies.asp (Abruf 24.07.2012)

Sehr wenige nicht gesichtsähnliche Emoticons, wie Herz und Rose, haben eine starke emotionale Aussage, die meist mit der sehr starken Emotion Liebe verknüpft ist.

Wie im Kapitel 3.2.1.3 gezeigt, wandeln die meisten der heutigen Chatclients textuell eingegebene Emoticons in grafische Repräsentationen um. Nutzen Sender und Empfänger unterschiedliche Programme zum Versenden und Empfangen der Nachrichten, so ist zu beobachten, dass die Darstellung des grafischen Emoticons nicht übereinstimmt und teilweise sehr stark voneinander abweicht, wie dies Abbildung 12 ersichtlich ist. Ein Standard für die Aussage von Emoticons und somit auch zu einer notwendigen Aussage der grafischen Repräsentation gibt es nicht. Selbst ein Pseudostandard ist nur bei sehr häufig genutzten Emoticons aufzufinden.

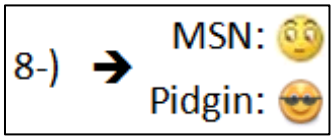


Abbildung 12: Gegenüberstellung Emoticonumwandlung in MSN-Messager und Pidgin

In diesem Kontext kann es weiterhin vorkommen, dass das Programm des Senders Emoticons unterstützt, die im Programm des Empfängers nicht als grafische Repräsentation dargestellt werden können, weil für die übermittelte Zeichenkette keine Ersatzgrafik definiert ist. Somit ist hier eine große Quelle für Missverständnisse. Viele Mitarbeiter in der Softwareentwicklung kennen ein weiteres Beispiel für den Verlust der Eindeutigkeit einer Nachricht aus ihrem Alltag. In Quelltexten sind des Öfteren aufeinanderfolgende Sonderzeichen wie Klammern, Bindestriche, Doppelpunkte oder Semikola zu finden. Werden diese via Instant Messaging übertragen, erfolgt beim Empfänger die automatische Umwandlung in Emoticons. Nur wenige Programme wie bspw. der Messager Pidgin[18] unterstützen die Abschaltung der Umwandlung von Emoticons in grafische Repräsentationen für einen selektierten Text. Durch die Nichteindeutigkeit der Umwandlungsfunktion („;-)“ und „;)“ werden in die identische Grafik überführt) müssen in diesem Szenario, wo es mitunter auf die korrekte und vollständige Reihenfolge der Zeichen ankommt, Informationsverluste hingenommen werden (vgl. Abbildung 13). Dies passiert, da keine Trennung von Nutztext und Emotionen bei der Übertragung erfolgt. Diese fehlende Trennung in Kombination mit der emotionalen Ungenauigkeit der Emoticons sorgt auch dafür, dass die während des Chats übertragenden Emotionen nur schwer maschinell weiterverarbeitet und ausgewertet werden können.

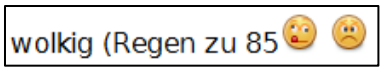


Abbildung 13: Vermischung Inhalt und Emotion bei "wolkig (Regen zu 85%) :-("

3.2.2 Nutzung von Auszeichnungssprachen und Ontologien

Neben der Darstellung als Emoticons oder ähnlichem können Emotionen in der textuellen Kommunikation auch mit der Hilfe von Auszeichnungssprachen abgebildet werden. Diese in der Regel auf der Extensible Markup Language (XML) [21] basierenden Sprachen sind für Maschinen genauer zu verarbeiten, als dies bei Emoticons der Fall ist. Während die Bedeutung von Emoticons erfahrungsbasiert und durch die Kommunikation selbst weitergegeben wird, findet bei den Auszeichnungssprachen ein mehr oder weniger

[18] Ein Open-Source-Messager, der mehrere Protokolle unterstützt.

komplexes Standardisierungsverfahren statt. Somit ist die Aussage hinter den Emotionsrepräsentationen deutlich klarer für Außenstehende nachvollziehbar, als dies bei der Verwendung von Emoticons der Fall ist. Durch eine hohe Nähe zur Technik werden Auszeichnungssprachen in der Regel nicht von Personen zur Auszeichnung von Emotionen während der Kommunikation genutzt, sondern sind für die Übertragung zwischen und Speicherung von Emotionen durch Programme vorgesehen.

3.2.2.1 Versenden der Gemütslage mit XMPP

Eine XMLbasierte Übertragung von Chattexten kann das freie Extensible Messaging and Presence Protocol (XMPP) leisten. Interessant wird dieses Protokoll durch seine Erweiterbarkeit in Form des XMPP Extension Protocol (XEP) [42]. Als Besonderheit ist festzustellen, dass es mit XEP-0107: User Mood [43] eine Erweiterung für das XMPP gibt, welche sich speziell mit der Übertragung der Gemütslage des Nutzers beschäftigt. Somit ist es möglich, dass diese Erweiterung als Basis für die Übermittlung von Emotionen der Lösung dienen kann.

Die Entwicklung des in Request for Comments (RFC) 6120 bis 6022 [44] [45] [46] spezifizierten Standards **XMPP** wird durch die XMPP Standards Foundation (XSF) verwaltet. Das Protokoll ist modular aufgebaut, so dass es durch XEPs erweitert werden kann. Die Verwaltung dieser XEPs obliegt dabei gänzlich der XSF. Durch diese Vorgehensweise ist XMPP ein offen dokumentierter Standard, welcher während seiner Weiterentwicklung für alle ersichtlich Prozesse einer globalen Standardisierung offen verfolgt.

Ein bedeutendes Einsatzgebiet für XMPP ist das Instant Messaging. Durch „Jabber“, dem ersten Instant-Messaging-Client, welcher XMPP unterstützte, ist das Protokoll daher landläufig auch unter diesem Namen bekannt. Die Grundstruktur von XMPP ist dezentral, d.h. es gibt mehrere XMPP-Server, die untereinander Nachrichten austauschen können. Es gibt also kein zentralisiertes System, wie das bei anderen Messagingprotokollen oft der Fall ist (vgl. Kapitel 4.1.1). Jedem XMPP-Server können mehrere Nutzer zugeordnet sein. Diese Nutzer übergeben mit Hilfe eines XMPP-Clients ihre Nachrichten an den XMPP-Server. Sollte der Empfänger der Nachricht Mandant des gleichen XMPP-Servers sein, so kann die Nachricht direkt zugestellt werden. Ist dies nicht der Fall, so wird eine Verbindung mit dem XMPP-Server, der den Empfänger verwaltet, hergestellt und diesem die Nachricht übergeben. Daher sind Verbindungen über die Grenzen des eigenen Anbieters hinweg möglich, wenn dies nicht zum Zwecke der Bildung eines Intranets am Server unterbunden wird. Eine Besonderheit des Protokolls sind so genannte Transports [47]. Diese sorgen serverseitig dafür, dass eine Verbindung zu Servern und damit zu Nutzern anderer Protokolle jenseits von XMPP aufgebaut werden kann, so dass diesen Nutzern Nachrichten weitergegeben werden können. Dadurch ist eine deutliche Steigerung der erreichbaren Nutzerbasis möglich. Die eindeutige Identifizierung der Nutzer erfolgt im XMPP über einen „Jabber Identifier“ (JID) [46]. Dieser in Form einer E-Mail-Adresse aufgebaute Bezeichner setzt sich aus einem Nutzernamen gefolgt von einem @ und dem Servernamen des XMPP-Servers zusammen (z.B. gerhard@example.org). Obwohl technisch gesehen E-Mail und XMPP unabhängig voneinander sind, ist es beliebt seine E-Mail-Adresse gleichzeitig auch als XMPP-Identifizierung zu nutzen. Dieses Verfahren ist bspw. mit einer E-Mail-Adresse von Google Mail nutzbar. Somit ist es auch möglich, Konversationen im Chat von Google+ via XMPP außerhalb der Plattform zu empfangen. Ähnliches Verhalten ist auch bei den Chats

des internationalen sozialen Netzwerks Facebook[19] sowie bei eher in Deutschland national agierenden sozialen Netzwerken der VZ-Gruppe (StudiVZ, MeinVZ, SchülerVZ) der Fall[20]. Neben der eindeutigen Nutzerkennung gibt es für jeden angemeldeten Nutzer noch das Prinzip der „Ressourcen“. Dieses ermöglicht dem Nutzer sich mehrfach gleichzeitig mit einer Identität von verschiedenen Endpunkten aus am XMPP-Server anzumelden. Der Bezeichner für die Ressource wird dabei der Nutzerkennung getrennt durch einen Schrägstrich angefügt (bspw. gerhard@example.org/home).

Neben grundlegenden Elementen einer Kommunikation via Instant Messaging, wie dies bspw. der Online-Status eines Nutzers ist, sind durch die Offenheit und Erweiterbarkeit des Protokolls viele Möglichkeiten hinzugekommen. Diese beschränken sich nicht nur auf Dateiübertragungen oder Konferenzen mit mehreren Benutzern, sondern gehen mit der von Google hinzugefügten XEP-0166: Jingle [48] selbst bis in den Bereich des Voice over IP (VoIP).

Im Fokus der weiteren Betrachtung soll die XMPP-Erweiterung **XEP-0107: User Mood** [43] stehen. Diese Protokollerweiterung soll die Möglichkeit geben, dass Informationen über die Gefühlslage des Nutzers ausgetauscht werden können, also ob eine Person gerade glücklich, traurig oder verärgert ist. Die Erweiterung liegt als „Draft Standard“ vor, d.h. das Implementierungen willkommen sind und eine Eignung des Protokolls in Produktivumgebungen möglich ist, es jedoch jederzeit zu Änderungen am Protokoll kommen kann. Die Erweiterung wurde von Peter Saint-Andre und Ralph Meijer entworfen und ist seit 29.10.2008 in der Version 1.2 verfügbar. Die initiale Version 0.1 wurde am 22.07.2003 veröffentlicht.

Die Informationen zu der Gefühlslage des Nutzers, welcher einer Emotion sehr nahe kommen, werden durch den Nutzer selbst bereitgestellt und mit Hilfe seines Clients im Netzwerk propagiert. Zur Abbildung der Gefühlslage wurde eine zusätzliche XML-Struktur im Namensraum http://jabber.org/protocol/mood um das XML-Element <mood> definiert ([43] Abschnitt 2.1). Die Emotion an sich wird in separaten XML-Elementen wie <happy /> oder <sad /> als Kindelemente von <mood> definiert. Weiterhin kann der Nutzer im XML-Element <text>, welches ebenfalls ein Kindelement von <mood> ist, eine natürlichsprachliche Beschreibung zu der übertragenen Stimmung hinzufügen. Die Gemütslagen sind nicht nur auf die im Anhang des Standards angegebenen 84 Verschiedenen beschränkt, sondern auch durch XML-Elemente in eigenen Namensräumen als Kindelemente der definierten Stimmung erweiterbar und präzisieren diese dadurch, wie Quelltext 1 zeigt.

```
<mood xmlns='http://jabber.org/protocol/mood'>
  <happy>
    <ecstatic xmlns='http://ik.nu/ralphm'/>
  </happy>
  <text>Yay, the mood spec has been approved!</text>
</mood>
```

Quelltext 1: Erweiterte Stimmung im XEP-0107[21]

Ein Anwendungsfall dieser XEP ist die Einbindung des momentanen Gemütszustands in eine Chatnachricht ([43] Abschnitt 2.3). Dabei wird der die Gemütslage darstellende Bereich als Schwesternelement zum <body>-Element, welches den eigentlichen Text überträgt, und

[19] siehe Dokumentation http://developers.facebook.com/docs/chat/ (Abruf: 08.09.2012)
[20] siehe Entwicklerblog http://developer.vz.net/2010/06/30/xmpp-chat-beta/ (Abruf 08.09.2012)
[21] Beispiel übernommen aus [43]

somit als Kindelement des alles umhüllenden <message>-Elementes in das XMPP-Paket eingebunden. Ein Beispiel ist in Quelltext 2 ersichtlich.

```
<message from='romeo@montague.lit/pda'
         to='juliet@capulet.lit'
         type='chat'>
  <body>A thousand times good night!</body>
  <mood xmlns='http://jabber.org/protocol/mood'>
    <sad/>
  </mood>
</message>
```

Quelltext 2: Zu einer Nachricht übertragener Gefühlszustand[22]

Neben der Einbettung von Gemütszuständen in die versendete Chatnachricht, ist ein Abonnement der Stimmungslage vorgesehen ([43] Abschnitt 2.2). Dabei veröffentlicht der Benutzer eine Stimmung optional mit beschreibendem Text und allen Nutzern, die die Gemütslage des Nutzers abonniert haben, wird diese zugestellt. Soll das Veröffentlichen der Gemütslage gestoppt werden, so wird ein leeres <mood>-Element an die Abonnenten versandt. Somit kann neben dem Gemütszustand in einzelnen Chatnachrichten auch die generelle Gemütslage abgebildet werden.

3.2.2.2 EmotionML

Neben dem XEP-0107 gibt es weitere Auszeichnungssprachen, die sich dem Kontext der Emotionen widmen. Ein weiterer Vertreter ist die Emotion Markup Language (EmotionML), eine beim W3C in der Standardisierung befindliche Auszeichnungssprache für Emotionen. Sie verfolgt das Ziel als „plug-in“-Sprache für die Darstellung von Emotionen nicht nur allein für sich zu stehen, sondern in andere Sprachen eingebettet zu werden. Zum Zeitpunkt der Ausarbeitung liegt EmotionML in Stadium einer W3C Candidate Recommandation vom 10. Mai 2012 [4] vor. Die Auszeichnungssprache soll primär in drei Bereichen Anwendung finden:

1. Manuelle Annotation von Daten
2. Automatische Erkennung von emotionsbezogener Zustände des Nutzerverhaltens
3. Generation von emotionsbezogenen Systemverhalten

Sie hat das Ziel, wissenschaftliche Erkenntnisse und praktischen Nutzen zu verbinden. Die mit EmotionML mögliche automatische maschinelle Verarbeitung von Emotionen in der Mensch-Maschine-Kommunikation schafft neue Anwendungsbereiche und kann helfen, die Frustrationsrate und Hilflosigkeit zu senken, wenn etwas bei dieser Interaktion nicht wie erwartet funktioniert. Auch ist die Verarbeitung von Emotionen in einer beobachteten Mensch-Mensch-Kommunikation möglich, so dass als Anwendungsszenario eine Frustrationsüberwachung beim Kundendienst denkbar ist. Dies alles ist möglich, da durch die EmotionML die Interoperabilität zwischen verschiedenen technologischen Komponenten bei der Datenverarbeitung und -übermittlung vereinfacht wird. Verschiedene weitere Anwendungsmöglichkeiten können im Standardisierungsdokument sowie damit im Zusammenhang stehenden Dokumenten erschlossen werden.

Nach einer ausführlichen Analysephase [49] ist die W3C Multimodal Interaction Working Group, welche EmotionML betreut, zu dem Entschluss gekommen eine Emotion in vier Dimensionen aufzuteilen (category = Kategorie, dimension = Ausmaß, appraisal =

[22] Beispiel übernommen aus [43]

Bewertung, action tendency = Handlungstendenz). Die Kategorie der Emotion ergibt eine Einteilung in das emotionale Gefühl, beschreibt also, ob die Emotion bspw. glücklich oder traurig ist. Das Ausmaß der Emotion beschreibt ihre Wertigkeit und wie dadurch eine Erregung stattfindet. Die Bewertung der Emotion bewertet den Kontext, in der die Emotion aufgetreten ist. Somit lässt sich abbilden, ob eine Vertrautheit mit der Emotion vorliegt, sie plötzlich auftrat oder kontrollierbar ist. Schlussendlich wird über die Handlungstendenz eine aus der Emotion folgende Handlung abgeleitet. Diese kann z.B. zur Ablehnung einer Sache führen.

Die genaue Benennung dieser verschiedenen Dimensionen erfolgt dabei über erweiterbare Vokabularien. Beispielvokabularien für verschiedene wissenschaftliche Einteilungen von Emotionen sind in einem separaten Dokument für Beispielvokabularien [50] aufzufinden. Diese sind auf den Seiten des W3C unter http://www.w3.org/TR/emotion-voc/xml maschinenlesbar abgelegt.

Als Namensraum für EmotionML selbst ist http://www.w3.org/2009/10/emotionml definiert. Weiterhin wird bei der Übertragung von EmotionML die Verwendung des Multipurpose Internet Mail Extensions type (MIME) application/emotionml+xml oder bei einer Speicherung in einer Datei die Dateiendung .emotionml empfohlen ([4] Anhang B).

Wird ein EmotionML-Dokument nicht in eine andere Auszeichnungssprache integriert, so bildet das <emotionml>-Element die Basis für alle weiteren im Standardisierungsdokument definierten Elemente. Dieses so genannte XML-Root-Element ist jedoch optionaler Natur, da es durch das das EmotionML umschließende Element bei einer Integration in eine andere Auszeichnungssprache weggelassen wird. Als Kindeelemente kann <emotionml> entweder ein <info>-Element für allgemeine Informationen, beliebig viele <vocabulary>-Elemente zur Abbildung von Emotionsvokabularien und/oder beliebig viele <emotion>-Elemente, welche die eigentliche Emotion abbilden, beherbergen. Zudem können die für die Definition der verwendeten Vokabularien zuständigen Attribute category-set, dimension-set, appraisal-set und action-tendency-set direkt am <emotionml>-Element angegeben werden. Eine Verwendung dessen im <emotion>-Element entfällt dann. Gleiches gilt für die Verwendung eines Attributes version, welches die Version von EmotionML (also momentan 1.0) angibt.

Das <emotion>-Element kann wie sein Elternelement ein <info>-Element sowie die Attribute für die Verweise zu den Vokabularien und der Versionsnummer beinhalten. Ergänzt wird dies durch einen optionalen eindeutigen Identifikator im Attribut id. Weiterhin ist es möglich mit Hilfe einer durch Leerzeichen getrennten Liste von Schlüsselwörtern im Attribut expressed-throught anzugeben, durch welches Medium die Emotion ausgedrückt wurde. Komplettiert werden die Attribute dieses Elements mit mehreren Attributen, die herangezogen werden können um der Emotion einen zeitlichen Kontext hinzuzufügen. Als Kindelemente von <emotion> sind die Elemente <category>, <dimension>, <appraisal> und <action-tendency> definiert, die die konkreten Eigenschaften des im Vokabular benannten Emotionsbestandteils definieren.

Aufgrund ihrer nahen Verwandtschaft sind diese Elemente auch gleich aufgebaut. Die Basis bildet das Attribut name, dessen Wert ein Begriff aus dem entsprechend gewählten Vokabular ist. Dieses kann ergänzt werden durch das Attribut confidence, welches die Güte der Korrektheit des angegebenen Emotionsbestandteiles im Wertebereich von 0 bis 1 als Gleitkommazahl angibt. Weiterhin ist eine Konkretisierung des Emotionsbestandteiles durch das Attribut value, dass die Intensität des Emotionsbestandteiles im gleichen Schema angibt,

möglich, wie das Beispiel in Quelltext 3 zeigt. Dieses Attribut wiederum kann durch ein Kindelement <trace> ersetzt werden. Jenes Element beschreibt dann anhand einer im Attribut freq angegebenen Frequenz, unter Zuhilfenahme der im Attribut samples angegebenen und durch Leerzeichen separierten Gleitkommawerte im Bereich von 0 bis 1, die zeitliche Änderung der Intensität.

```
<emotion category-set="http://www.w3.org/TR/emotion-voc/xml#big6">
    <category name="happyness" value="0.7" confidence="0.95"/>
</emotion>
```

Quelltext 3: Notation einer Emotionskategorie mit glücklicher Emotion

Ein weiteres optionales Kindelement von <emotion> ist <reference>. Hierdurch lässt sich die dargestellte Emotion mit einer beliebigen anderen durch einen URI referenzierbaren Ressource in Relation setzen. Dieser URI wird als Referenzziel im Attribut uri notiert. Die Relation zu dieser Referenz kann durch die Attribute role und media-type näher beschrieben werden. Während das Attribut role mit Hilfe seiner Werte „expressedBy“, „experiencedBy“, „triggeredBy“ und „targetedAt“ näher beschreibt, in welcher Art von Relation die Emotion zu der referenzierten Ressource steht, kann im Attribut media-type der MIME type der referenzierten Ressource angegeben werden. Wie Abschnitt 2.4.2.4 des EmotionML-Standardisierungsdokuments entnommen werden kann, ist es möglich dabei bspw. bei einer Referenz zu Videos durch ein URI-Fragment bzw. einen Query-Parameter t gemäß der Media Fragments Recommendation [51] genau auf die entsprechende Stelle im Video zu referenzieren.

Um die Definition von EmotionML zu vervollständigen, wird im Abschnitt 3 von EmotionML auf die Definition der Vokabularien eingegangen. Genau wie bei <emotion>-Elementen ist es auch für die <vocabulary>-Elemente, welche die Vokabularien repräsentieren, möglich, dass diese innerhalb eines <emotionml>-Wurzelelementes vorkommen oder in eine andere Auszeichnungssprache ohne <emotionml>-Element eingebettet werden. Einem Vokabular sind immer durch das Attribut type ein Typ (also „category“, „dimension“, „appraisal“ oder „action-tendency“) sowie eine eindeutige ID im Attribut id zugeordnet. Jedes <vocabulary>-Element hat dabei wiederum Kindelemente <item>, welche die einzelnen Einträge des Vokabulars darstellen. Deren Attribut name benennt den Eintrag so, dass dieser im name-Attribut bspw. des <category>-Elements angesprochen werden kann. Ein Beispiel für ein komplettes Emotionsvokabular ist in Quelltext 4 aufgeführt. Sowohl <vocabulary> als auch <item> kann jeweils wieder ein <info>-Bereich für weitere Informationen oder zusätzliche XML-Elemente wie RDF-XML hinzugefügt werden. Leider sind zurzeit die Vokabularien selbst nicht in RDF definiert, was eine bessere Weiterverarbeitung durch Semantic-Web-Technologien ermöglichen würde. Dies ist durch fehlendes Expertenwissen über RDF in der Working Group zu begründen. [52]

```
<emotionml xmlns="http://www.w3.org/2009/10/emotionml">
    <vocabulary type="category" id="big6">
        <item name="anger"/>
        <item name="disgust"/>
        <item name="fear"/>
        <item name="happiness"/>
        <item name="sadness"/>
        <item name="surprise"/>
    </vocabulary>
</emotionml>
```

Quelltext 4: Definition der EmotionML-Kategorie big6

3.2.2.3 Smiley Ontology

Neben der Auszeichnungsmöglichkeit von Emotionen oder vergleichbaren Elementen mit der Hilfe von speziell angepassten Auszeichnungssprachen ist eine Abbildung der Informationen durch Ontologien möglich. In diesen Begriffswelten lassen sich gerade in Kombination mit Technologien des Semantic Web mit der Repräsentation von Emotionen in Zusammenhang stehende Elemente maschinenverarbeitbar abbilden. Eine Ontologie, die für die Beschreibung von Emoticons vorgehsehen ist, ist die Smiley Ontology, welche von Mitgliedern der Universität Belgrad ins Leben gerufen wurde [24]. Diese Ontologie tritt unter anderem der vielfältigen Abwandlung von Emoticons und den Inkompatibilitäten verschiedener Systeme in diesem Bereich, die in Kapitel 3.2.1.6 beschrieben wurden, entgegen. Nicht nur die Interoperabilität verschiedener Systeme soll durch die Ontologie gestärkt werden. Es soll auch unterstützt werden, dass der Ausdruck einer Emotion durch ein Emoticon anstelle von beschreibendem Text zu einem leichteren, schnelleren und sogar semantisch korrekteren Abbild der Emotion führt. Um diese Ziele zu erreichen, müssen die Grenzen der in verschieden Instant Messagern unterschiedlich vordefinierten Emoticons zugunsten ihrer eigentlichen Beschreibung aufgebrochen werden. Es wird also die Struktur und Bedeutung eines Emoticons beschrieben, anstatt eine Zeichenkette oder Grafik des Emoticons als Repräsentant für die Emotion zu übertragen. Wenn eine die Emotion bzw. das Emoticon verarbeitende Anwendung nun ein Emoticon empfängt, welches nicht in der Anwendung implementiert ist, so ist kann es der Anwendung doch möglich sein, anhand des Kontexts des Emoticons ein geeignetes Substitut mit möglichst gleicher Bedeutung zu finden. Durch die detaillierte und gute maschinenverarbeitbare Abbildung von Emoticons, werden Algorithmen, die Datamining mit Hilfe dieser Informationen durchführen, in die Lage versetzt, den Kontext einer Emotion in das Suchergebnis einzubeziehen. Somit ist auch eine Filterung der Informationen nach emotionalem Kontext oder Eigenschaften des Emoticons möglich, wodurch sich wieder neue Anwendungsfelder ergeben. Da Emotionen ein hochkomplexes Gut sind, ist es in manchen Situationen einfacher, die eigene Emotion mit Hilfe eines kurzen, gut beschriebenen Emoticons auszudrücken, als durch einen Menge von Wörtern, die den eigenen emotionalen Zustand beschreibt. Durch die semantische Verknüpfung eines Emoticons mit einem emotionalen Zustand kann die Einführung der Nutzung konkreter Emotionen in der textuellen Kommunikation über die momentan gewohnte Nutzung von Emoticons erleichtert werden. Die bestehenden Nutzergewohnheiten können also für eine bessere maschinennahe Abbildung aufgegriffen werden.

Die nach Stankovic definierten drei Ebenen eines Emoticons (zugrundeliegende Emotion, Struktur und Erscheinungsbild) [29], werden auch im Aufbau der Smiley Ontology [53] deutlich. Den Kern bildet dort eine Klasse Emoticon, der durch die Property representsEmotion eine Klasse Emotion zugeordnet ist. Diese Emotion möchte der Absender vermitteln und der Empfänger erfassen. Weithin werden über die Property hasFraction durch die Objekt-Klasse EmoticonFraction weitere Eigenschaften über die Struktur des Emoticons abgebildet. Dies können beispielsweise ein Objekt wie eine Sonnenbrille oder ein Gesichtsausdruck sein. Die Zuordnung einer visuellen Repräsentation des Emoticons erfolgt dabei über die Klasse VisualRepresentation, die über die Property representsBy an das Emoticon gebunden ist. Hierbei handelt es sich um eine zeichenbasierte Repräsentation (Klasse CharacterRepresentation) oder ein Bild (Klasse Picture). Letzteres kann wiederum Bestandteil eines Systems von Emoticons, also einer Sammlung der Emoticons einer bestimmten Plattform wie bspw. von einem sozialen Netz, sein (Klasse EmoticonSystem). Über die Property isAnimated kann ausgedrückt werden, ob

die Repräsentation animiert ist oder nicht. Ein komplexes Beispiel für die Notation in Smiley Ontology ist in Quelltext 5 dargestellt. Der Name Smiley Ontology ist an sich ungenau, da sich mit dieser Ontologie auch Emojis oder emotionale Komponenten des Netzjargons abbilden lassen (vgl. Kapitel 3.2.1.1).

```
<!DOCTYPE emotionml [
    <!ENTITY smiley "http://www.smileyontology.com/ns#" >
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
]>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:foaf="http://xmlns.com/foaf/0.1/"
         xmlns:smiley="&smiley;">
  <smiley:Emoticon>
    <smiley:representsEmotion><smiley:Emotion/></smiley:representsEmotion>
    <smiley:representedBy><smiley:CharacterRepresentation>
      <smiley:characters><![CDATA[:-D]]></smiley:characters>
    </smiley:CharacterRepresentation></smiley:representedBy>

    <smiley:representedBy><smiley:Picture>
      <smiley:belongsToSystem rdf:resource="&smiley;SkypeEmoticonSystem"/>
      <smiley:isAnimated rdf:datatype="&xsd;boolean">
        false
      </smiley:isAnimated>
      <foaf:depiction rdf:resource="http://example.com/smiley122.gif"/>
    </smiley:Picture></smiley:representedBy>
  </smiley:Emoticon>
</rdf:RDF>
```

Quelltext 5: Beispiel eines Emoticons in RDF-XML (Zeichen und Bild)

3.2.2.4 Emotion Ontology

Neben der Möglichkeit ein Emoticon als Repräsentation der Emotion mit Hilfe einer Ontologie darzustellen, ist es auch möglich dies für die Emotion direkt zu tun. Zu diesem Zweck kann die Emotion Ontology von Hastings et al. [54]. genutzt werden. Diese Ontologie hat das Ziel, gebrauchsfähige Beschreibungen und Kategorisierungen emotionaler Erfahrungen unter Berücksichtigung von Resultaten wissenschaftlicher Untersuchungen zu schaffen. Hierfür wurde eine Ontologie mit semantischen Technologien entworfen [55]. In dieser sind neben einer Reihe hierarchisch verbundener Emotionen auch Elemente zum Bewertungsprozess einer Emotion sowie aus dieser Bewertung abgeleiteten Handlungsweisen definiert. Dadurch lassen sich die in dieser Ontologie definierten Emotionen bspw. im Zusammenspiel mit der Property representsEmotion der Smiley Ontology [53] nutzen, ersichtlich in Quelltext 6.

```
<smiley:representsEmotion
rdf:resource="http://purl.obolibrary.org/obo/MFOEM_000034"/>
```

Quelltext 6: Nutzung Emotion Ontology mit Smiley Ontology

3.2.3 Nutzung von Emotionserkennung

Eine Emotion ist nicht nur ein gedanklicher Prozess, sie äußert sich auch konkret im Verhalten des Menschen und seines Körpers. Auf sehr heterogene Art und Weise können eigene Emotionen meist subtil und ohne eigenes bewusstes Zutun an die Außenwelt abgegeben werden, so dass sie von einem Außenstehenden aufgenommen und in den Kontext der Kommunikation integriert werden können. Aufgrund der hohen Heterogenität der

emotional bedingten Handlungen des Menschen und seines Körpers, ist auch die Erkennung von Emotionen sehr breit gefächert.

Nach Krämer/Quappe [6] unterteilt sich die Kommunikation in einen verbalen, einen paraverbalen und einen non-verbalen Bestandteil. In all diesen Bereichen kann eine Erkennung von Emotionen stattfinden. Bei der verbalen Kommunikation steht das eigentliche Wort im Fokus der Betrachtungen. Der gesprochene oder geschriebene Inhalt, der die in der Kommunikation von Sender zu Empfänger übertragene Äußerung bildet, kann dabei auf vielfältige Art und Weise einer Analyse mit dem Ziel der Extraktion von Emotionen unterzogen werden (vgl. Schuller [56] Abschnitt 4). Dabei wird u.a. beachtet, dass Personen, getrieben durch ihren emotionalen Zustand und erlernten Verhaltensweisen, bestimmte Ausdrücke oder grammatikalische Veränderungen in bestimmten emotionalen Zuständen verwenden. Durch die Betrachtung einzelner Schlüsselwörter oder auch komplexer Wortfolgen unter Zuhilfenahme von Sprachmodellen, kann eine Vergleichsbasis für die Emotionsbetrachtung gebildet werden. Der Vergleich dieser Betrachtung mit Referenzwerten mündet schließlich in der Bestimmung einer Emotion. Der für die Betrachtung herangezogene Inhalt kann dabei sowohl akustisch, als auch schriftlich übermittelt sein. Daher können diesem Prozess der Emotionserfassung auch paraverbale Bestandteile zur Analyse hinzugefügt werden. Diese Art und Weise der Übertragung einer Emotion in einer Nachricht könnte sich u.a. in einer zittrigen Handschrift äußern, die folgend einer Bildanalyse zur Betrachtung der Klarheit der Schriftlinien unterzogen werden kann. Bei gesprochenem Wort existiert unter Zuhilfenahme von Mikrophonen und Algorithmen die Möglichkeit einer akustischen Emotionserkennung bspw. in einer angsterfüllten Stimme. Der menschlichen Sprache können in diesem Fall für eine weitergehende Analyse prosodische, stimmqualitative und artikulatorische Charakteristika für die Analyse entnommen werden (vgl. Schuller [56] Abschnitt 3). Im Betrachtungskontext der vorliegenden Ausarbeitung ist zumeist das geschriebene Wort im Instant Messaging relevant. Erfolgt die Übertragung z.B. mittels Nachrichten im Format der Hypertext Markup Language (HTML), so ist auch hier durch unterschiedliche Formatierung oder Färbung von Bestandteilen des Textes eine paraverbale Übertragung eines emotionalen Zustandes möglich. Eine Verfassung des Textes in Großbuchstaben wird bspw. als ein schreiender Ausruf wahrgenommen [57]. In der chatbasierten Kommunikation müssen auch Emoticons als paraverbale Bestandteile der Kommunikation der Analyse hinzugefügt werden [58]. Dabei sind kulturelle Unterschiede zu berücksichtigen [31]. Schuller zeigt, dass eine Erkennung von emotionalen Zuständen auch durch die Auswertung der Interaktion mit Eingabegeräten wie mit Maus oder Touchscreen Erkennungsraten bis zu 90% möglich machen ([56] Abschnitt 5). Durch die steigende Verbreitung meist mit Touchscreens verbreiteter Smartphones [59] und der stetig zunehmenden Nutzung von Instant Messaging im mobilen Kontext [60] gewinnt diese Erkenntnis weiter an Bedeutung. Neben verbalen und paraverbalen Bestandteilen der Kommunikation, gibt es auch non verbale Bestandteile, die für eine Emotionserkennung dienen können. So kann die mit Hilfe von Kameratechnik aufgenommene Mimik durch Algorithmen des Medien Retrieval erkannt und mit Vergleichswerten interpretiert werden, um einen emotionalen Zustand bestimmen zu können [61]. Auch hier werden bspw. bei ausreichendem Training neuronaler Netze gute Erkennungsraten erreicht. Aus dem Bereich der non-verbalen Kommunikation kann unter Nutzung der Motion-Capture-Technologie eine Emotion auch anhand des Ganges erschlossen werden [62].

Emotionen können aber nicht nur über Elemente erfasst werden, die für einen Teilnehmer der persönlichen Kommunikation hörbar oder sichtbar sind. Auch subtile oder innere

körperliche Veränderungen können für die Erkennung einer Emotion zu Rate gezogen werden. Eine Möglichkeit dies zu tun stellt die Messung der Herzaktivitäten mittels Elektrokardiogramms dar, also der Messung der Herzfrequenz, Zeiten zwischen zwei Herzschlägen, der Herzfrequenz-Variabilität sowie der Respiratorischen Sinusarrhythmie [63]. Sinkt die Herzfrequenz, so ist dies ein Anzeichen für eine Entspannung der Person. Sie befindet sich in einem glücklichen Zustand. Steigt jedoch die Variabilität der Herzfrequenz, so ist dies ein Indiz für Stress oder Frustration. Abhängig von der Herzaktivität ist somit auch der über ein Blutdruckmessgerät bestimmbare Blutdruck [64]. Steigt der Blutdruck, so ist dies ein Anzeichen für Wut oder Stress. Sinkt er hingegen, kann dies von Entspannung oder gar Traurigkeit herrühren. Mit Hilfe der Elektroenzephalografie kann die Aktivität des Gehirnes gemessen werden [65]. Auch hier können unter Zuhilfenahme von Neuronalen Netzen bei speziellem Training brauchbare Ergebnisse generiert werden. Ein weiteres Merkmal stellt die Muskelaktivität dar, die mit Hilfe der Elektromyografie gemessen wird [66]. Sie hat Bezug zu der Emotionserkennung anhand der Mimik, Gestik und Zeichensprache. Analog der Bestimmung anhand der Muskelaktivität lassen sich auch Emotionen anhand der Atmung und ihrer Frequenz bestimmen [66]. Eine ansteigende Atemfrequenz deutet auf Ärger oder Freude hin, ihr abklingen auf Entspannung und Glück. Auch eine Emotion durch Auswertung des Hautwiderstandes ist möglich [67]. Zu dessen Ermittlung sowie der Ermittlung der Hauttemperatur kann der Q-Sensor der Firma Affectiva, Inc. eingesetzt werden. Dieses am Handgelenk ähnlich einer Uhr tragbare Gerät dient als Sensor und kann zusätzlich noch Bewegungsinformationen bestimmen. Die Messdaten werden dann drahtlos für die weitere Emotionserkennung an ein spezielles Programm übertragen [68].

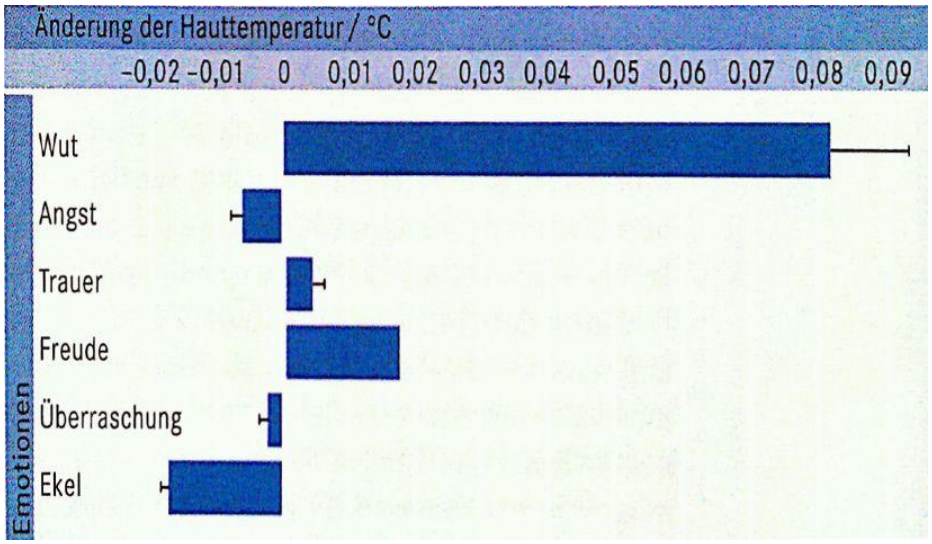


Abbildung 14: Zusammenhang zwischen Hauttemperatur und Emotion[23]

Wie anhand von Abbildung 14 und Abbildung 15 ersichtlich ist, können allein durch die Messung zweier Werte, nämlich Hauttemperatur und die Änderung der Herzfrequenz, die Emotionen Wut, Angst, Trauer, Freude, Überraschung und Ekel erkannt werden. Gleiches ist durch die Kombination anderer Methoden zur Emotionserkennung möglich. Die hier behandelten Möglichkeiten stellen bei weitem keine vollständige Auflistung dar, verdeutlichen aber, dass durch die heterogenen Möglichkeiten, die die wissenschaftliche Disziplin der Emotionserkennung prägt, anhand der Nutzung mehrerer verschiedener Messsysteme eine Emotion mit hoher Qualität erkannt werden kann.

[23] Abbildung entnommen aus [11]

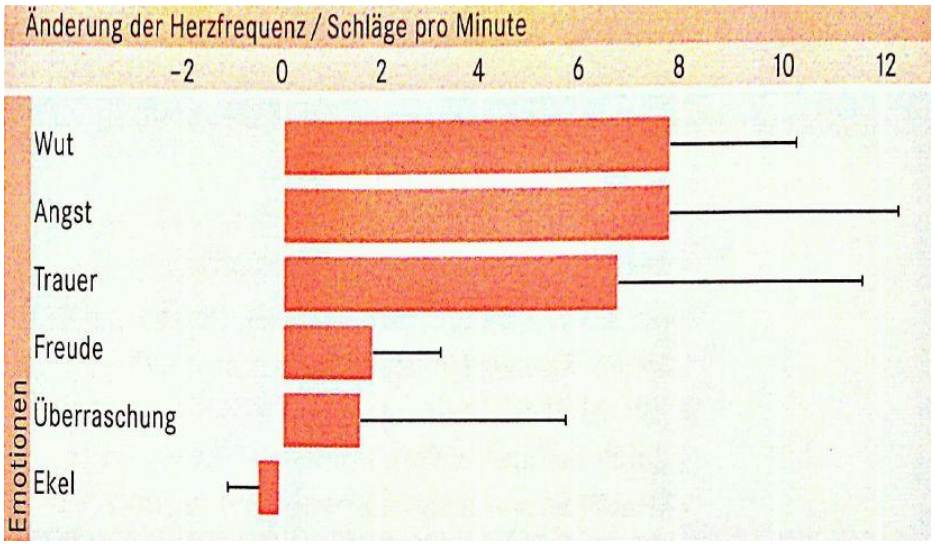


Abbildung 15: Zusammenhang zwischen Herzfrequenz und Emotion[24]

Alle emotionserkennenden Systeme, haben die Gemeinsamkeit, dass sie darauf angewiesen sind, dass das beobachtete Individuum zu einer Emotionsäußerung anhand der für die Emotionserkennung herangezogenen Messszenarien in der Lage sein muss. Dies muss nicht immer der Fall sein. Bei Menschen, die einer Gesichtslähmung unterliegen, ist die Emotionserkennung via Mimik nicht möglich. Ähnlich verhält es sich bei von Geburt an gehörlosen Menschen, die ihre Emotion nicht gut anhand der Stimme übertragen können, da sie sich selbst noch nie hörten. Kompensierend kann bei diesen jedoch der emotionale Ausdruck durch die Mimik vielfältiger und präziser sein, als es bei einem durchschnittlichen Menschen der Fall ist. Wird dies bei der Bewertung der Messdaten nicht beachtet, so kann es hier zu einer fehlerhaften Erkennung kommen. Findet eine Beachtung statt, können die Erkennungsraten nochmals gesteigert werden. Weiterhin ist beobachtbar, dass viele Prototypen zur Emotionserkennung über keine standardisierte Ausgabe von Emotionen verfügen und diese nur in unterschiedlichsten proprietären Darstellungen oder Texten ausgegeben werden (vgl. [69]).

3.3 Evaluierung der Nutzungsmöglichkeiten

Nachdem nun die drei Szenarien näher betrachtet wurden, soll eine Evaluierung der Szenarien anhand der in Kapitel 3.1.1 aufgestellten Bewertungskriterien gemäß des in Kapitel 3.1.2 aufgestellten Bewertungsschemas erfolgen.

Bei der Nutzung von Emoticons und Netzjargon ist als sehr gut zu bewerten, dass diese bereits im Bereich des Instant Messaging eingesetzt werden. Gleiches trifft für den Bereich der sozialen Netzwerke zu. Eine gute Bewertung ist bei der unterstützenden Emotionserfassung möglich, da Emoticons im alltäglichen Gebrauch als Emotionen wahrgenommen werden. Hierbei ist die vollkommene Klarheit über die ausgesagte Emotion jedoch nicht immer vorhanden, wie die Ausführungen in Kapitel 3.2.1.6 zeigen. Ebenfalls eine gute Bewertung ist im Punkt der Verwendung offener Protokolle möglich, da Emoticons und Elemente des Netzjargons oft, wie in den Kapiteln 3.2.1.1 und 3.2.1.2 beschrieben, aus Zeichen eines bekannten und offen dokumentierten Zeichensatzes gebildet werden. Hierbei ist jedoch zu beachten, dass diese Protokolle nicht direkt für die Übertragung von Emotionsrepräsentationen geschaffen wurden, sondern die Arbeit mit Text zum Ziel haben.

[24] Abbildung entnommen aus [11]

Auch eine Erweiterung der Inputfaktoren kann mit gut bewertet werden, da durch Abwandlungen von Emoticons oder neuen Elementen im Netzjargon jederzeit eine Erweiterung der Emotionsdarstellung erfolgen kann (vgl. Kapitel 3.2.1.5). Problematisch ist hier jedoch, dass sich neue Inhalte unter den Nutzern herumsprechen müssen. Im Bereich „Implementierung“ ist das Kriterium des Vorhandenseins mehrerer Quellen für den Emotionsinput auch gegeben, was zu einer guten Bewertung führt. Aus dem Text, der im gleichen Kontext wie die Emoticons verwendet wird, sowie durch eine unterschiedliche Formatierung, kann auf die Emotion geschlossen werden. Eine Trennung von Inhalt und Emotionsdarstellung findet dabei jedoch nicht statt (siehe Abbildung 13: Vermischung Inhalt und Emotion bei "wolkig (Regen zu 85%) :-("), weshalb dieses Kriterium als sehr schlecht erfüllt gewertet werden muss. Eine Nutzung von Standards muss als neutral bewertet werden, da zwar manche Emoticons in Codepage 437 [37] und im Unicode [38] [39] standardisiert sind, jedoch in der Masse der Korrespondenz genau diese Emoticons nicht genutzt werden. Dadurch dass diese Emoticons aber zumeist mit frei verfügbaren Zeichensätzen erstellt werden, kann das nächste Kriterium der freien Verfüg- und Nutzbarkeit mit sehr gut bewertet werden. Die Übergabe von Text von einem System in ein anderes ist dabei vielfach angewandt und dementsprechend dokumentiert. Da diese Dokumentation nur für die Übergabe von Text geschaffen wurde, nicht jedoch für die Übergabe von Emotionen, kann dieses Kriterium nur mit gut bewertet werden. Primär mit Hilfe von Emoticons ist eine Abbildung einer Emotion möglich, allerdings sind diese sehr heterogen, so dass ein Emoticon nicht als solches und damit auch nicht als Emotion erkannt werden muss (vgl. Kapitel 3.2.1.1). Somit kann dieses Kriterium nur mit gut bewertet werden. Im Bereich „Emotionen“ muss die Eindeutigkeit der abgebildeten Emotion mit schlecht bewertet werden, da dies bei Emoticons nicht immer der Fall ist, wie Tabelle 4: Gegenüberstellung Smileys und Emojis zeigt. Hieran kann auch sehr gut nachvollzogen werden, dass ein großer interkultureller Unterschied im Bereich der Emoticons vorliegt. Die Auflösung dessen ist nicht immer einfach und nur mühsam möglich, weshalb das Kriterium der Umwandlungsmöglichkeit in die Deutungsweise einer anderen Kultur mit sehr schlecht bewertet werden muss. Da Emoticons jedoch mit einer Emotion in der textuellen Kommunikation gleichgesetzt werden, erfüllen sie das Kriterium der Möglichkeit zur Serialisierung von Emotionen gut. Abzüge müssen hier aufgrund der manchmal ungenauen maschinellen Weiterverarbeitung gemacht werden. Die Übertragung von Zeichen ist dabei zwar als gut für die Übertragung von Emotionen zu bezeichnen, ein sehr gut kann für dieses Kriterium allerdings nicht vergeben werden, da die Übertragung von Zeichen nicht auf Emotionen optimiert ist. Die Möglichkeit der Zusammenfassung mehrerer Emotionen zu einer Repräsentation kann als gut bewertet werden, da über neue Emoticons oder Akronyme solche Objekte eingeführt werden können. Hier kommt allerdings wieder die Unbekanntheit neuer Emoticons beim Empfänger zum Tragen (vgl. Kapitel 3.2.1.5). Aufgrund der geringen Standardisierungstiefe von Emoticons und der geringen Nutzung dieser Standardisierungen ist eine sehr gute und einfache Erweiterung des Vokabulars für Emotionen möglich.

Für die Nutzung von Auszeichnungssprachen und Ontologien erfolgt die Bewertung analog zum vorherigen Abschnitt. Durch die Existenz der XMPP-Erweiterung XEP-0107 [43] ist die Nutzung für Instant Messaging als sehr gut einzuschätzen. Das XMPP [45] findet dabei nicht nur als Chatprotokoll Anwendung, sondern erfährt vielerlei andere Nutzungsszenarien, so dass die Möglichkeit der Anbindung an soziale Netzwerke als gut definiert werden kann. Durch die Fokussierung dieser Auszeichnungssprachen auf die neutrale Auszeichnung von Emotionen ist jedoch eine mögliche Unterstützung der Emotionserfassung als sehr schlecht zu beurteilen. Die Protokolle XMPP-0107 und EmotionML [4] haben einen offenen

Standardisierungsprozess durchlaufen, sodass die Verwendung offener Protokolle als sehr gut bewertet werden muss. Inputvektoren sind aufgrund der Ausrichtung als Übertragungs- und Auszeichnungsprotokolle allerdings nicht vorhanden, was zu einer neutralen Behandlung dieses Kriteriums führt. Allerdings ist es mit dem <reference>-Element von EmotionML möglich, mehrere Referenzen zu verschiedenen Emotionsinputs anzugeben, weshalb dieses Kriterium als gut gewichtet wird. In den Protokollen findet eine sehr gute Trennung von Inhalt und Emotion statt. Aufgrund der offenen Standardisierung werden Standards sehr gut genutzt, ist eine sehr gute Verfüg- und Nutzbarkeit vorhanden und die Schnittstelle entsprechend dokumentiert, was in der mit der gleichen Wertigkeit hierfür resultiert. Durch die Nutzung von Emotionsvokabularien oder gar -ontologien, ist die klare und eindeutige Abbildung der Emotion jeweils sehr gut gegeben. Wie der weitere Verlauf dieser Ausarbeitung in Kapitel 5.4.2 zeigen wird, ist auch durch eine nachgelagerte Lösung eine gute Umwandlung in die Deutungsweise einer anderen Kultur möglich. Aufgrund der Tatsache, dass XEP-0107 extra für die Nutzung im Instant Messaging geschaffen wurde, ist jeweils eine sehr gute Serialisierung und Übertragung von Emotionen möglich. Allerdings kann die Zusammenfassung mehrerer Emotionsrepräsentationen zu einer nur durch Überlagerungen geschehen, weshalb die Erfüllung dieses Kriteriums als schlecht beurteilt wird. Im Gegensatz dazu ist in allen betrachteten Protokollen eine sehr gute Erweiterung des Emotionsvokabulars möglich.

Bei der Verarbeitung von Ergebnissen einer Emotionsmessung muss leider festgestellt werden, dass diese nur schlecht für Instant Messaging und soziale Netzwerke direkt einsetzbar ist. Alleinig die Übermittlung der Ausgaben über Trägermedien ist möglich. Ein mehr als sehr gutes Ergebnis erreicht dieser Bereich jedoch logischerweise bei der Unterstützung einer Emotionserfassung. Zwar werden von momentan gängigen Anwendungen nur zumeist proprietäre Ausgaben getätigt (vgl. [69]), was als sehr schlecht einzuordnen ist, jedoch ist, wie Kapitel 3.2.3 zeigt, die Erweiterbarkeit der Inputvektoren sehr gut. Ein Erkennungssystem von Emotionen kann dabei auch mehrere Sensoren ansprechen, weshalb das Kriterium der mehreren benötigten Quellen für einen Emotionsinput auch als gut erfüllt bewertet werden kann. Abzüge gibt es hier aufgrund des zumeist geschlossenen Systems. Auch die Trennung von Emotion und Inhalt kann als gut bewertet werden, da z.B. eine Analyse von Text zum Erkennen einer Emotion führen kann. Leider lässt sich hier nicht immer die Emotion dann auch aus dem Text heraus extrahieren. Ein bereits dargelegtes Problem sind die proprietären Ausgabeformate, weshalb eine Nutzung von Standards in diesem Bereich als sehr schlecht bewertet werden muss. Dafür gibt es Open-Source-Software[25], die frei verfüg- und nutzbar ist und zu der meist eine Dokumentation existiert. Somit sind diese beiden Kriterien als gut einzustufen. Weiterhin als gut einzuordnen ist die Abbildungsmöglichkeit einer Emotion. Durch die proprietäre Ausgabe erfolgt dies oft mittels einfachen Textes. Die Eindeutigkeit der Emotion ist dabei, wie Versuche zeigen [56], als sehr gut einzustufen. Durch die Begrenzung auf die Emotionserkennung ist die Umwandlung in eine Deutungsweise einer anderen Kultur als schlecht zu beurteilen. Die Ausgaben können nur von nachgelagerten Programmen dementsprechend angepasst werden. Eine Serialisierung der Emotion findet jedoch gut statt, da die reale Emotion in eine virtuelle Repräsentation umgewandelt wird. Leider geschieht dies oft in kein offizielles und standardisiertes Format wie EmotionML [4]. Eine Übertragung der Emotion ist auch als schlecht zu beurteilen, da durch die Ausgabe von reinem Text die Emotion mitunter nicht

[25] Ein gutes Beispiel ist „Human Emotion Detection from Image“ http://www.codeproject.com/Articles/110805/Human-Emotion-Detection-from-Image (Abruf 11.08.2012)

vom restlichen Inhalt des Transportmediums automatisch getrennt werden kann. Da mehrere Sensorquellen für die Erkennung der Emotionen hinzugezogen werden können, ist auch eine Zusammenfassung mehrerer Emotionsrepräsentationen sehr gut möglich. Die Erweiterbarkeit des Emotionsvokabulars ist dabei jedoch nur je Anwendung erweiterbar und somit nur als gut zu bewerten.

Legende:

++	sehr gut
+	gut
o	neutral
-	schlecht
--	sehr schlecht

	Emoticons & Netzjargon	**Auszeichnungssprachen und Ontologien**	**Ergebnisse von Emotionsmessungen**
Zusammenspiel mit externen Systemen			
für Instant Messaging nutzbar	++	++	-
Möglichkeit der Anbindung an soziale Netzwerke	++	+	-
Emotionserfassung unterstützen	+	--	++
Verwendung offener Protokolle	+	++	--
Erweiterbarkeit der Inputvektoren	+	o	++
Implementierung			
mehrere Quellen für Emotionsinput	+	+	+
Trennung von Emotion und Inhalt	--	++	+
Nutzung von Standards	o	++	--
Open Source/frei verfüg- und nutzbar	++	++	+
dokumentierte Schnittstellen	+	++	+
Abbildung einer Emotion	+	++	+
Emotionen			
Eindeutigkeit der abgebildeten Emotion	-	++	++
Emotion in Deutungsweise einer anderen Kultur umwandeln	--	+	--
Serialisierung von Emotionen	+	++	+
Übertragung von Emotionen	+	++	-
Zusammenfassung mehrerer Emotionsrepräsentationen	+	-	++
Vokabular für Emotionen erweiterbar	++	++	+
Gesamtgüte	27	56	15

Tabelle 6: Evaluierung der aufgestellten Szenarien anhand der Bewertungskriterien

Nachdem die Kriterien der Lösungen bewertet wurden, kann gemäß dem in Kapitel 3.1.2 festgelegten Bewertungsschema eine Gesamtgüte der jeweiligen Lösungen festgestellt werden. Die Verwendung von automatischer Emotionserkennung ist mit 15 von möglichen 76 Punkten in der Gesamtgüte klar abgeschlagen. Klarer Favorit für eine mögliche Lösung mit einer Gesamtgüte mit 56 von möglichen 76 Punkten ist die Verwendung von Auszeichnungssprachen und Ontologien. Hierbei ist jedoch auffällig, dass diese Variante im Bereich der Unterstützung der Emotionserfassung sowie bei der Zusammenfassung mehrerer Emotionsrepräsentationen Mängel aufweist. Diesem ließe sich durch die gleichzeitige Verwendung einer der anderen beiden Szenarien begegnen. Weiterhin darf nicht vergessen werden, dass diese Gruppe ein Zusammenschluss mehrerer separater aber gleichartiger Technologien ist, die allerdings alle auf dem gemeinsamen Standard XML basieren und daher gut kombinierbar sind.

3.4 Zusammenfassung Stand der Technik

Die Betrachtung des Stands der Technik hat gezeigt, dass viele verschiedene Herangehensweisen einen Umgang mit Emotionen in der textuellen Kommunikation zu ermöglichen. Die Verwendung von Emoticons und Elementen aus dem Netzjargon ist dabei bei den Internetnutzern weit verbreitet und anerkannt. Allerdings sind diese aufgrund vieler Möglichkeiten für Missverständnisse, sei es durch die unterschiedliche Interpretation von westlichen und östlichen Emoticons (vgl. Kapitel 3.2.1.1) oder durch unbekannte Abkürzungen im Netzjargon (vgl. Kapitel 3.2.1.4), nicht in allen Kontexten und Gesellschaftsschichten einsetzbar. Durch weitere Probleme, die bei der Verwendung von Emoticons und angelehnten Elementen entstehen können (vgl. Kapitel 3.2.1.6), sind sie jedoch keine Lösung für das in dieser Arbeit behandelte Problem.

Anders sieht es da im Bereich der Auszeichnungssprachen und Ontologien aus. Bis auf die Kriterien der Emotionserfassung, der Erweiterung der Inputfaktoren und der Zusammenfassung mehrerer Emotionsrepräsentationen zeigten sich diese durchaus als brauchbar. Dabei ist jedoch zu beachten, dass es die Kombination der Protokolle ist, die dieses Ergebnis erreicht hat. Während das XMPP [45] und speziell die XMPP-Erweiterung für Übertragung von Gefühlszuständen XEP-0107 [43] im Bereich des Instant Messagings und der sozialen Netzwerke ihre Kernkompetenz einbringen, konnte EmotionML [4] mit einer flexiblen Abbildung der Emotion zum Erfolg beitragen. Die Emotion Ontology [54] ist dabei dafür verantwortlich, dass die Emotion in kompletter Klarheit abgebildet werden kann.

Die Emotionsmessung allein hat sich im Test als nicht geeignet herausgestellt. Bemerkenswert ist jedoch, dass sie bei den Kriterien gut punkten konnte, wo sich Auszeichnungssprachen und Ontologien eher schwach präsentierten. Weiterhin könnte diese Technologieklasse sehr viele verschiedene Inputvektoren für die Lösung bereitstellen (vgl. Kapitel 3.2.3).

Da die Auszeichnungssprachen und Ontologien die Kriterien am besten erfüllen, können sie als Basis für eine Lösung genommen werden. Bei der Arbeit mit Emotionen ist der Nutzer sehr im Fokus, so dass dessen Akzeptanz neuen technologischen Gegebenheiten gegenüber beachtet werden muss. Hier zeigt sich, dass die Verwendung von Emoticons eine sehr hohe Akzeptanz und auch Verbreitung inne hat. Durch die Kombination von Emoticons, Auszeichnungssprachen und Ontologien können die Kriterien komplett erfüllt werden. Weiterhin ist somit ein gutes Bootstrapping für die Einführung einer kombinierten Technologie möglich.

4 Lösungskonzept

Gemäß der Zielsetzung soll ein Prototyp entwickelt werden, der Emotionen aus verschiedenen Quellen aggregieren und in ein geeignetes Serialisierungsformat überführen kann, das sich zur Übertragung im Bereich des Instant Messaging und sozialer Netzwerke eignet. Dafür wird nach den Anforderungen in Kapitel 3.1 ein offenes und erweiterbares Übertragungsprotokoll benötigt. Bevor für die Lösung ein geeigneter Prototyp implementiert werden kann, muss eine Systemarchitektur definiert werden, mit der sich diese Anforderungen umsetzen lassen. Diese muss verschiedene Quellen zur Eingabe von Emotionen in das System genauso beachten wie die mögliche Übertragung einer Chatnachricht inklusive Emotion zu einem Gesprächspartner oder einem sozialen Netzwerk.

Da der Absender einer Nachricht im Moment ihres Verfassens eine konkrete Emotion aufweist, müssen die verschiedenen Emotionen über einen flexiblen und anpassbaren Gewichtungsalgorithmus zu einer oder mehreren auf die Chatnachricht oder deren konkreten Teile zugeschnittenen Emotion zusammengefasst werden. Wie in Kapitel 3.2.3 festgestellt wurde, ist dabei auf die Eigenarten bei einer möglichen Emotionsäußerung des Nutzers einzugehen und die Emotion anhand dieser entsprechend zu gewichten.

Die Basis der Lösung stellt das geeignete Übertragungsprotokoll dar, da dies und seine Fähigkeiten wiederum Auswirkungen auf die globale Systemarchitektur der Lösung haben können. Ist dies gefunden, kann im weiteren Verlauf eine Systemarchitektur definiert werden. Anschließend muss ein sich in diese Architektur integrierender Prototyp definiert werden. Dies stellt aufgrund der Nähe zum Instant Messaging hier ein Chatclient dar. Ist dies geschehen, kann der Umgang mit mehreren Quellen für Emotionen in diesem näher abgebildet werden. Dies mündet dann schließlich in einer Definition der vom Nutzer abhängigen Gewichtung der dem Chatclient zugeführten Emotionen.

4.1 Definition eines geeigneten Übertragungsprotokolls

Bei der Wahl der zugrundeliegenden Protokolle und im speziellen einem Übertragungsprotokoll im Bereich des Instant Messagings gibt es mehrere Kriterien an die Lösung, denen durch die gewählten Protokolle für die Abbildung der Emotion und die Übertragung im Instant Messaging entsprochen werden muss. Diese sind konkret:

- für Instant Messaging nutzbar
- Möglichkeit der Anbindung an soziale Netzwerke
- Verwendung offener Protokolle
- Trennung von Emotion und Inhalt
- Nutzung von Standards
- Open Source/frei verfüg- und nutzbar
- Abbildung einer Emotion
- Eindeutigkeit der abgebildeten Emotion
- Serialisierung von Emotionen
- Übertragung von Emotionen
- Vokabular für Emotionen erweiterbar

4.1.1 Wahl zugrundeliegender Protokolle

Diese Kriterien müssen bei der Definition eines geeigneten Übertragungsprotokolls Beachtung finden. Dabei ist auch zu beachten, dass Instant Messaging und soziale Netzwerke von Masseneffekten geprägte Medien sind. Ihre Attraktivität und der resultierende Nutzen steigen mit zunehmender Teilnehmerzahl. Es existieren bereits viele etablierte Plattformen sowohl für Instant Messaging als auch für soziale Netzwerke, die jeweils eine hohe Nutzerschaft aufweisen. Daher ist es empfehlenswert, für die Abbildung des Instant Messagings kein neues Übertragungsprotokoll einzuführen, sondern stattdessen ein in diesen Bereich etabliertes Protokoll zu verwenden oder zu erweitern, um sofort eine hohe Nutzerschaft erreichen zu können. Ähnlich verhält es sich im Kontext der sozialen Netzwerke. Es ist empfehlenswert, dass die resultierende Lösung mit etablierten sozialen Netzwerken interagieren kann, anstatt an dieser Stelle neue Netzwerke zu schaffen.

Im Bereich des Instant-Messagings gibt es nur wenige Anbieter, die eine globale Bedeutung haben. Wie statistische Quellen [70] belegen, hat deutschlandweit der Anbieter ICQ die Marktführerschaft, gefolgt von The Microsoft Network (MSN), Yahoo, AOL Instant Messenger (AIM) und Google Talk (GTalk). Die Chatclients Jabber und QQ sind dabei abgeschlagen. Dies ist jedoch nicht weltweit identisch, wie andere Quellen zeigen [71]. Diese besagen, dass ICQ nur in Deutschland und Russland eine bedeutende Rolle in Form einer Marktführerschaft zukommt. Nur im chinesischen Raum ist QQ mit ca. 60% vorherrschend, während MSN dort 25% Marktanteil belegt. MSN kann dafür die Marktführerschaft in Amerika (außer USA), Australien, großen Teilen Europas und Japan für sich verbuchen. Dieser Dienst hat teilweise sehr hohe Marktanteile, sogar bis zu 84% in Mexiko. Auch in Ländern Südasiens, wo der Yahoo-Messager eine marktbeherrschende Stellung einnimmt, kann MSN mittelmäßige Marktanteile vorweisen. GTalk kann zwar keine Marktführerschaft erreichen, ist jedoch in manchen Ländern wie Südafrika, Indien und Japan gut vertreten. AIM hat mit 35% Marktanteil die Marktführerschaft in den USA, wird aber dicht gefolgt von MSN und Yahoo. Ansonsten spielt dieses Protokoll wie auch Jabber weltweit keine Rolle. Technisch gesehen nutzen AIM und ICQ beide das Protokoll OSCAR (Open System for Communication in Realtime). Sowohl Jabber als auch GTalk greifen auf XMPP zurück.

Im Bereich der sozialen Netzwerke hat das Netzwerk Facebook klar die weltweite Marktführerschaft inne [72]. Jedoch kann eine regionale Marktführerschaft auf Staatenebene nicht überall erreicht werden. Hier ist bspw. In China das Netzwerk Qzone führend, so dass es von den Mitgliederzahlen her auch im weltweiten Vergleich auf Facebook folgt. In Brasilien hat das von Google betriebene Netzwerk Orkut die Marktführerschaft inne. Das Netzwerk VKontakte ist im russischen Raum die erste Wahl. Auch der Microblogging-Dienst Twitter weist eine weltweit starke Verbreitung auf, kann jedoch in keinem Land eine Marktführerschaft erreichen. Auch dem im Jahre 2011 gestarteten Google+ kann eine immer größere Bedeutung zugeschrieben werden [73]. Alle anderen Netzwerke werden aufgrund ihrer geringeren Nutzeranzahl oder Regionalität nicht betrachtet.

Die aus den Anforderungen abgeleiteten Bewertungskriterien fordern eine Verwendung offener und standardisierter Protokolle für die Übertragung via Instant Messaging. Dies ist bei einigen der betrachteten Instant-Messaging-Diensten leider nicht der Fall. Das Protokoll OSCAR, welches ICQ und AIM nutzen, ist proprietärer Natur. Analog verhält es sich mit dem in China genutzten QQ. Auch für die Protokolle von Yahoo und Microsoft sind keine offiziellen Standardisierungen auffindbar, die eine freie Weiterentwicklung des Protokolls erlauben. Anders ist dies bei GTalk und Jabber der Fall, welche beide XMPP als Protokoll einsetzen, das durch RFCs öffentlich standardisiert ist. Weiterhin hat dieses Protokoll den Vorteil, dass bei eine Erweiterung unterstützt und gefördert wird. Das XMPP ist auch ein guter Vertreter für die Verwendung mit sozialen Netzwerken. So lassen sich die Chats von Google+, Facebook, den VZ-Netzwerken und vKontakte via XMPP abonnieren. Somit wird die Möglichkeit geschaffen, dass auch mit einem alternativen XMPP-Client eine Konversation mit einem anderen Mitglied der Plattform geführt werden kann. Wie anhand von E-Mail-Adressen bei Google, die gleichzeitig auch XMPP-Identifikatoren sind, gesehen werden kann, muss dies auch nicht wie bei Facebook auf die eigene Plattform beschränkt sein. Für Twitter existieren zahlreiche XMPP-Transports[26]. Dass das chinesische Netzwerke Qzone keine XMPP-Kompatibilität aufweist, kann aufgrund von Sprachbarrieren nicht abschließend festgestellt werden.

[26] Ein Beispiel hierfür wäre https://www.tweet.im/ (Abruf 09.09.2012)

Für die Übertragung von Gefühlszuständen wurde die XMPP-Erweiterung XEP-0107: User Mood [43] spezifiziert. Wie in Kapitel 2.2 und in der Analyse im Kapitel 3.2 festgestellt wurde, sind Emotionen sehr komplex und vielschichtig. Sie werden in mehreren Dimensionen wie Emotionskategorie, Ausmaß und Bewertung der Emotion sowie einer Handlungstendenz abgebildet. Zwar erfüllt XEP-0107 die Kriterien, jedoch kann es dieses komplexe System der Abbildung nicht wiedergeben. Abhilfe kann hier die in Kapitel 3.2.2.2 vorgestellte EmotionML [4] bieten. Diese Auszeichnungssprache wurde vom W3C speziell für die Abbildung von Emotionen geschaffen und als Plugin-Sprache zur Kombination mit möglichst vielen anderen Sprachen definiert. Dadurch kann sie die Emotion wesentlich granularer und mit mehr Möglichkeiten und Zusatzinformationen abbilden, als dies bei XEP-0107 der Fall ist. Weiterhin sind bei letzteren in den dort spezifizierten XML-Elementen Emotionen und Handlungstendenzen vermischt (glücklich = Emotion, abgelenkt = Handlungstendenz). Weiterhin wurde im Jahre 2003 mit der Entwicklung von XEP-0107 begonnen, was viele Jahre war, bevor das W3C mit der EmotionML eine universelle Pluginsprache speziell für Emotionen zu entwickeln begann. Schon allein aus Gründen der möglichen Interoperabilität mit anderen Systemen ist EmotionML für diese Erweiterung des XMPP eine bessere Wahl.

4.1.2 Einbindung von EmotionML in eine XMPP-Nachricht

Wie die Analyse in Kapitel 4.1.1 ergab, muss eine Möglichkeit gefunden werden, EmotionML in eine XMPP-Nachricht zu integrieren. Die Basis dafür detaillierte Betrachtung in Kapitel 3.2.2.1, wo der grundlegende Aufbau einer XMPP-Nachricht erläutert wird. Wie in diesem Kapitel beschrieben ist, besteht eine reine Chatnachricht ohne Erweiterungen aus einer XML-Struktur, deren als Rootknoten fungierender <message>-Knoten als Kindelement einen <body>-Knoten aufweist, der wiederum als Textknoten die eigentliche Chatnachricht beherbergt, wie dies exemplarisch in Quelltext 7 zur Darstellung kommt. Ist wie hier ein Rootknoten gegeben und wird daher das EmotionML in eine andere Auszeichnungssprache eingebettet, so besagt die EmotionML-Spezifikation, dass der die Emotionen umschließende <emotionml>-Tag entfällt (vgl. [4] Abschnitt 2.1.1).

```
<?xml version="1.0" encoding="utf-8" ?>
<message from='romeo@example.net/home'
         to='juliet@example.org'
         type='chat'>
    <body>Mir geht es gut.</body>
</message>
```

Quelltext 7: XMPP-Nachricht "Mir geht es gut."

Eine Umschließung des <body>-Tags mit EmotionML nach dem Beispiel im Abschnitt 5.1 des EmotionML-Standardentwurfs ist nicht möglich, da es nach RFC6121 [45] ein direktes Kindelement von <message> ist. Weiterhin besagt dieser RFC in Abschnitt 5.2.3, dass der in <body> übertragene Text menschenlesbar und XMLcodiert sein soll. Dadurch ist es auch nicht möglich, dass EmotionML als Kindelement des <body>-Tags den eigentlichen Nachrichtentext umschließend einzusetzen. Somit bleibt für die Integration von EmotionML in XMPP als einzige Möglichkeit die <emotion>-Knoten als Schwesternknoten des <body>-Knoten einzupflegen, wie dies in Quelltext 8 dargestellt wird. Um eine Unterscheidung auch bei der Übertragung mehrerer Emotionen gewährleisten zu können, wird im Attribut id jedes <emotion>-Tags ein eindeutiger Bezeichner gemäß Abschnitt 2.1.2 EmotionML vom Typ xsd:ID (vgl. [71] Abschnitt 3.4.8) notiert. Da das EmotionML in eine andere Auszeichnungssprache eingebettet wird, muss das Attribut version sowie der Namensraum von EmotionML dem <emotion>-Element hinzugefügt werden.

```
<?xml version="1.0" encoding="utf-8" ?>
<message from='romeo@example.net/pda'
         to='juliet@example.org'
         type='chat'>
    <emotion id="emotion1" version="1.0"
             xmlns="http://www.w3.org/2009/10/emotionml"
             category-set="http://www.w3.org/TR/emotion-voc/xml#big6">
        <category name="happiness" value="0.8"/>
    </emotion>
    <body>Mir geht es gut.</body>
</message>
```

Quelltext 8: XMPP-Nachricht mit eingebetteter Emotion in EmotionML

Wird nur ein <emotion>-Bereich im XMPP-Paket versendet, so ist die Wahrscheinlichkeit sehr hoch, dass die dort beschriebene Emotion dem gesamten Inhalt des <body>-Tags zugeordnet werden soll. Dies ist auch der Fall, wenn mehrere <emotion>-Bereiche dem gesamten Inhalt zugeordnet werden sollen (bspw. aufgrund einer Erweiterung der Emotion mit einem eigenen Emotionsvokabular oder zwei verschiedenen Emotionskategorien). Wird nun aber ein Text übertragen, dessen Teilbereiche mehrere Emotionen repräsentieren, so müssen diese auch getrennt voneinander in mehreren <emotion>-Bereichen angegeben werden. Somit ist es nötig eine Verbindung in Form einer Referenz von einem Teil des Textes zu seiner zugehörigen Emotion zu ermöglichen. Als Beispiel für dieses Szenario sei der übertragene Text „Du hattest vor mir gestanden und mich verzaubert, bis du dich plötzlich umgedreht hast und gingst.“ angeführt. Im ersten Teilsatz ist anzunehmen, dass dessen Autor fröhlich gestimmt gar freudig erregt war. Beim zweiten Teil des Satzes schlägt diese Emotion doch plötzlich in eine nicht erfreuliche Emotion um, die die vorhergehende unterbricht. Es sind also zwei völlig verschiedene Emotionen beobachtbar.

Die Emotion ist dem Text untergeordnet. Daher ist ein Verweis vom Text zur Emotion nötig. Ein Endpunkt für die Referenz kann die ID im <emotion>-Tag darstellen. Einen Startpunkt für die Referenz am <body>-Tag, welches den Nachrichtentext enthält, ist nicht möglich, da in dessen Definition in Abschnitt 5.2.3 der RFC6121 außer das Sprachattribut xml:lang keine Attribute definiert sind[27]. Da in <body> nur menschlich lesbarer Text vorkommen soll, es somit nur einen Textknoten beinhaltet, ist es auch nicht möglich dieses Element durch Kindelemente für mehrere Referenzen zu entsprechenden Emotionen vorzubereiten[28]. Eine Unterteilung in mehrere <body>-Elemente je nach Emotion ist ebenfalls nicht realisierbar, da alternative <body>-Elemente sich nicht im Inhalt, sondern nur in der Sprache unterscheiden sollen. Der entsprechende Sprachcode wird dabei in xml:lang angegeben[29]. Gleiches gilt für ein <subject>-Element, welches einen Betreff zu der im <body> übertragenen Nachricht darstellen soll.

Eine Referenz vom Text der Nachricht aus ist durch die definierten Richtlinien des XMPP also nicht möglich. Somit muss die Alternative einer Rückreferenz von der Emotion zum ihr zugrundeliegenden Text in Erwägung gezogen werden. Im Abschnitt 2.4.1 von EmotionML ist eine Referenzierung von der Emotion zu einer anderen Ressource definiert. Dieses geschieht über das <reference>-Element, welches ein Kindelement von <emotion> ist und dessen Attribut uri eine URI der zu referenzierenden Ressource beinhalten muss. Das

[27] „There are no attributes defined for the <body/> element, with the exception of the 'xml:lang' attribute.“ [45]
[28] „The <body/> element MUST NOT contain mixed content” [45]
[29] „Multiple instances of the <body/> element [...] providing alternate versions of the same body [...] with a distinct language value” [45]

XPointer-Framework bietet mit dem xpointer()-Schema [74] die Möglichkeit Teile eines XML-Dokumentes mit Hilfe der XML Path Language (XPath) [75] zu referenzieren. Da die XMPP-Nachricht bei der Übertragung keinen eigenen URI aufweist, muss die Referenzierung relativ nach RFC1808 [76] geschehen. Für die Einschränkung der gesamten Zeichenkette des Textknotens in <body> auf einen für die Emotion relevanten Teilbereich kann die Funktion substring(), definiert in Abschnitt 7.4.3 der separaten Funktionsreferenz von XPath [77], zu Hilfe genommen werden. Der erste Parameter dieser Funktion definiert den zu behandelnden Text, während der zweite Parameter den Beginn der resultierenden Zeichenkette angibt, die maximal die im dritten Parameter angegebene Anzahl weiterer Zeichen haben soll. Für eine Referenzierung von der Emotion zum ersten Teilsatz des gewählten Beispiels, welcher Zeichen 1 bis Zeichen 48 der Zeichenkette enthält, ist somit eine Referenz wie in Quelltext 9 anzugeben.

```
<reference uri=".#xpointer(substring(/message/body/text(),1,48))" />
```

Quelltext 9: Mögliche Referenz zu einem Textbereich via XPointer

4.1.3 Prüfung der Kompatibilität mit anderen Erweiterungen am Beispiel XEP-0071

Im nächsten Schritt wird die universelle Nutzung der im Kapitel 4.1.2 durch EmotionML in eine XMPP-Nachricht eingebetteten Emotion durch die Verwendung in Kombination mit einer XMPP-Erweiterung evaluiert und bei Bedarf angepasst. Das XMPP kann durch eine Reihe von Erweiterungen, den so genannten XEP ergänzt werden[30]. Eine gerade im Kontext des Instant-Messagings häufig genutzte Erweiterung ist dabei XEP-0071 XHTML-IM [78] zur Übertragung von Chatnachrichten durch die Extensible Hypertext Markup Language (XHTML), womit eine Textformatierung ermöglicht wird. Ein Beispiel hierfür ist in Quelltext 10 dargestellt. Deshalb soll exemplarisch an dieser Erweiterung das problemlose Zusammenspiel mit anderen Erweiterungen, speziell im Hinblick auf Referenzierungen zu den Emotionen, evaluiert werden. XEP-0071 erweitert das XMPP um die Möglichkeit Chatnachrichten in XHTML 1.0 [79] zu versenden. Dabei wird ein kompletter HTML-Bereich inklusive <html>-Tag in die XMPP-Nachricht eingebettet. Somit können die aus dem HTML-Kontext bekannten Referenzierungsmöglichkeiten betrachtet werden. Da die XHTML-Nachricht wie die Emotion auch als Schwesternelement des <body>-Element notiert ist, kommt es hier zu keinerlei Beeinflussungen. Eine Einbettung des EmotionML in die XHTML-Nachricht ist zwar möglich, widerspricht aber der Anforderung, dass Text und Emotionen voneinander getrennt werden sollen. Weiterhin ergibt sich ein zusätzlicher Overhead, da in diesem Fall die Emotion doppelt übertragen werden müsste, nämlich einmal in der XHTML-Nachricht und einmal in der XMPP-Nachricht direkt, da nicht sichergestellt werden kann, dass ein genutzter Chatclient die Erweiterung XEP-0071 unterstützt.

Die sehr häufig genutzte Referenzierung durch einen Hyperlink mit Hilfe des <a>-Tag in HTML scheidet für die Nutzung aus, da dieser Link dem Nutzer höchstwahrscheinlich angezeigt und zur Interaktion bereitgestellt werden würde. Daher muss eine Möglichkeit gefunden werden, eine maschinell verarbeitbare, aber nicht für den Nutzer unmittelbar sichtbare Referenz darzustellen. HTML kennt für die Referenzierung zu anderen Quellen weiterhin die Attribute rel und rev. Während rel den Typ einer Referenz vom beherbergenden Dokument zu einer anderen im href-Attribut angegebenen Ressource angibt, stellt rev dabei

[30] Eine Liste der XEP ist auf der Seite der XMPP Standards Foundation unter http://xmpp.org/xmpp-protocols/xmpp-extensions/ zu finden (Abruf 10.08.2012)

in gleicher Art und Weise eine Rückwärtsbeziehung zum angegebenen Dokument dar (vgl. [80]). Obwohl in der Spezifikation viele Linktypen vordefiniert sind[31], lassen sich diese durch eigene erweitern[32]. Hiermit könnte ein Linktyp definiert werden, der für eine Referenz zu einer Emotion verwendbar ist. Weiterhin ist eine Verwendung von <link> im <head>-Bereich des XHTML zu ungenau, da hier keine eindeutige Referenzierung ausgehend von Teilen der Textnachricht erfolgen kann.

```
<?xml version="1.0" encoding="utf-8" ?>
<message from='romeo@example.net/home'
         to='juliet@example.org'
         type='chat'>
    <body>Mir geht es gut.</body>
    <html xmlns='http://jabber.org/protocol/xhtml-im'>
        <body xmlns='http://www.w3.org/1999/xhtml'>
            <p>Mir geht es gut.</p>
        </body>
    </html>
</message>
```

Quelltext 10: XMPP Nachricht mit Erweiterung nach XEP-0071 (XHTML-IM)

Eine weitere Möglichkeit, logische und maschinenverwertbare Verbindungen zwischen mehreren Ressourcen zu schaffen, existiert mittels Techniken des Semantic Web, speziell durch die Verwendung von RDF (Kapitel 2.3). Es gibt viele verschiedene Möglichkeiten RDF-Daten in eine XHTML-Seite zu integrieren (vgl. [81]). Da eine Recherche kein spezielles Prädikat hervorbrachte, mit welchem eine Relation zu einer Emotionsrepräsentation angedeutet werden kann, wird allgemein rdfs:seeAlso (vgl. [19] Abschnitt 5.4) als Verweis zu einer weiteren das Subjekt beschreibenden Ressource gewählt. Eine direkte Einbindung von RDF mittels eines HTML-Kommentares, wie sie einfach zu realisieren wäre, hat im semantischen Sinne das Problem, dass der Interpreter durch den Kommentar angewiesen wird, die Inhalte des Kommentares nicht zu beachten. So auch das in den Kommentar eingebettete RDF.

Eine relativ selten betrachtete Form der Integration von RDF in HTML ist Embedded RDF (eRDF) [82]. Dabei werden vorhandene XHTML-Attribute verwendet (allen voran id, class und im Kopf der XHTML rel im <link>-Tag), um in ihnen Teile von RDF-Tripeln zu definieren, welche dann mittels eines eRDF-Parsers oder einer XSL-Transformation (XSLT) [83] in eine allseits bekannte RDF-Repräsentation umgewandelt werden können. Dieses Verfahren ist dahingehend als verwendbar für die Einbettung von RDF in eine XEP-0071-Nachricht anzusehen, als dass das XHTML valide gegen seien Dokumenttypdefinition (DTD) (vgl. [21] Abschnitt 2.8) geprüft werden kann.

Eine weitere Form der Einbettung RDF-ähnlicher Inhalte in ein XHTML-Dokument stellen Mikroformate dar [84]. Hier werden, ähnlich wie bei eRDF, spezielle Attributwerte in vorhandene XHTML-Attribute eingesetzt, die dann mittels Microformats-Parser in RDF umgewandelt werden können. Der Nachteil dieser Technologie gegenüber eRDF ist, dass es nur eine Reihe speziell definierter Typen gibt, die bspw. für Personendaten oder Eventbeschreibungen genutzt werden können. Eine Verwendung von rdfs:seeAlso ist mit Mikroformaten jedoch nicht ohne weiteres möglich.

[31] Liste der Linktypen unter http://www.w3.org/TR/xhtml-modularization/abstraction.html#dt_LinkTypes (Abruf 09.09.2012)

[32] „Authors may wish to define additional link types […]"

Ähnlichkeit mit Mikroformaten und eRDF weist auch RDF in Attributes (RDFa) [85] auf. Hierbei werden teilweise vorhandene XHTML-Attribute genutzt. Weiterhin werden auch neue XHTML-Attribute hinzugefügt, die die Bestandteile der Tripel des RDF aufnehmen sollen, damit diese dann von einem RDFa-Parser als RDF verarbeitet werden können. Diese Erweiterung ist dahingehend als kritisch einzustufen, da das XHTML in diesem Fall nicht mehr erfolgreich gegen seine DTD validiert werden kann. Dem soll durch die Einbindung einer alternativen DTD für RDFa Abhilfe geschaffen werden. Die Einbindung der DTD ist jedoch im Kontext einer XMPP-Nachricht nicht möglich, da eine DTD immer vor dem Root-Knoten eingebunden werden muss. Würde das aber hier geschehen, so würde sich die DTD auf die gesamte XMPP-Nachricht beziehen und nicht nur auf ihren XHTML-Teil. Die Kreation einer für diesen Kontext komplett gültigen DTD kann ebenfalls nicht erfolgen, da das XMPP ein erweiterbares Protokoll ist und in Zukunft weitere heute noch unbekannte Erweiterungen erfolgreich mit der DTD validiert werden müssen. Ähnliche Probleme treten auch bei dem im Kontext von HTML5 (Hypertext Markup Language Version 5) aufkommenden Microdata auf, wo ähnlich wie bei RDFa neue HTML-Attribute eingeführt werden [86].

Mit Hilfe von Angaben im Kopfbereich der XHMTL-Nachricht gibt es noch weitere betrachtungswerte Möglichkeiten der Einbindung von RDF. Eine Möglichkeit ist die Einbindung eines Object-Tags (vgl. [87] Abschnitt B.3 – Optional Elements in head). Dabei wird ein RDF-Dokument base64-codiert [88] als inline-Daten des <object>-Tags angegeben [89]. Diese Variante ist auch praktikabel für den gegebenen Anwendungsfall. Eine weitere praktikable Lösung stellt die Einbindung des RDF in einem <script>-Tag mit dem MIME-Type [90] von RDF, nämlich application/rdf+xml, dar. Noch eine Möglichkeit von XHTML zu RDF zu referenzieren, ist der Verweis zu einer RDF-Datei mittels <link>-Tag im Header des XHTML. Das ist jedoch für den vorliegenden Anwendungsfall nicht praktikabel, da ein zweites Dokument benötigt wird. Ähnlich verhält es sich bei GRDDL (Gleaning Resource Descriptions from Dialects of Languages) [91], welches mit Hilfe einer im Header angegebenen XSL-Transformation RDF-Daten aus dem Dokument extrahiert.

4.1.4 Umsetzungsentscheidung für Erweiterung des XMPP

Die Analyse in den Kapiteln 4.1.2 und 4.1.3 hat ergeben, dass sowohl bei der einfachen Einbindung der Emotion in eine XMPP-Nachricht, als auch in Kombination mit einer Erweiterung durch eine XHTML-Nachricht nach XEP-0071 eine erfolgreiche Einbettung des EmotionML als Schwesternelement des <body>-Elements möglich ist. Zwar zeigt die Analyse auch, dass es in beiden Fällen möglich ist, eine maschinenlesbare Verknüpfung zwischen dem Text und der Emotion herzustellen, jedoch muss dies über unterschiedliche Technologien erfolgen. Es ist daher anzunehmen, dass sich ähnliche Inkonsistenzen auch mit anderen Protokollerweiterungen ergeben würden. Zudem kann so keine einheitliche Implementierung erreicht werden, bei der eine schnelle und einfache Referenzierung vom Text zur ihm untergeordneten Emotion möglich ist.

Um eine einfache Übertragung mehrerer Emotionen zu einem Text zu ermöglichen, soll die Textnachricht in emotionale Blöcke aufgeteilt werden. Diese Blöcke werden dann mit ihren zugehörigen Emotionen, die nicht separat mit dem Text referenziert sind, übertragen. Standardmäßig sendet XMPP mit der Nachricht keinen Zeitstempel. Um die korrekte Reihenfolge der nahezu gleichzeitig abgesendeten Nachrichten beim Empfänger auch bei unterschiedlichen Netzlaufzeiten ermitteln zu können, wird XEP-0203: Delayed Delivery [92] zu Hilfe genommen. Diese XEP ermöglicht, dass in einer XMPP-Nachricht angegeben

werden kann, wann diese von wem versendet wurde. Dazu wird der Nachricht ein Element <delay> der Nachricht hinzugefügt. Obwohl dessen Attribut from für die Angabe des Absenders im vorliegenden Szenario nicht benötigt wird, wird es mit seinem entsprechenden Wert doch notiert, der Empfehlung des XEP-0203 folgend. Für den Anwendungsfall ist jedoch wichtig die Angabe des Attributes stamp. Hier wird eine Zeitangabe nach XEP-0082: XMPP Date and Time Profiles [93] angegeben. Dieser Standard definiert explizit den Aufbau verschiedener Zeitangaben für das XMPP. Abschnitt 3.2 von XEP-0082 folgend, soll für vorliegendes Szenario ein vollständiger Timestamp mit Datum, Zeit, Millisekunden und Zeitzone als Wert des Attributes stamp gewählt werden. Hierbei ist weiterhin darauf zu achten, dass sich die versendeten Nachrichten chronologisch um mindestens eine Millisekunde in der Absendezeit unterscheiden. Wird eine Nachricht nicht in einzelne emotionale Blöcke zerlegt, so braucht die Notation der Absendezeit nach XEP-0203 keine Beachtung zu finden.

4.2 Definition einer Systemarchitektur

Wie sich in der vorherigen Analyse herausgestellt hat, ist EmotionML das zur Abbildung von Emotionen präferierte Protokoll, welches auch in der Lösung genutzt werden soll. Für eine Nutzung im Bereich des Instant Messagings wird dieses in eine XMPP-Chatnachricht integriert und vom Sender zum Empfänger gemäß der XMPP-Spezifikationen versendet. Daher ergibt sich eine dezentralisierte Lösung im Bereich des Instant-Messaging, bei der ein XMPP-Server einem anderen die Chatnachricht übergeben kann. Dieses Vorgehen wird in Abbildung 16: Übersichtsbild Systemarchitektur im wolkenhaft abgebildeten Teil der Übertragung über das Internet dargestellt. Die Analyse in Kapitel 3.2.3 ergab weiterhin, dass es zwar Frameworks zur Emotionserkennung gibt, diese jedoch größtenteils proprietäre Ausgaben der Emotionen aufweisen. Um eine klare und flexible Lösung abbilden zu können, ist es empfehlenswert, dass nur standardisierte Emotionsdarstellungen in EmotionML der Verarbeitung zugeführt werden. Daher ist es nötig im Vorfeld die proprietären Emotionsrepräsentationen über vorgelagerte Filter in EmotionML zu wandeln. Da die Lösung als Open-Source-Software verbreitet werden soll, kann hier aus rechtlichen Gründen nicht immer eine Umwandlung proprietärer Formate in EmotionML innerhalb der Lösung erfolgen. Daher werden diese Umwandlungen in dynamisch erweiterbare Programmteile als Adapter ausgelagert und über eine Schnittstelle mit der Lösung verbunden (in Abbildung 16 grün dargestellt). Dieses Verfahren macht es möglich, dass emotionserkennende Algorithmen direkt in die dem Adapter vorgeschalteten Programmteile übernommen werden können. Somit soll an dieser Stelle eine vollwertige Pluginschnittstelle entstehen. Wie Kapitel 3.2.3 weiterhin zeigte, ist eine Erkennung von Emotionen anhand des vom Nutzer eingegebenen Textes möglich. Um über die Pluginschnittstelle den Algorithmen Daten für die textbasierte Emotionserkennung bereitstellen zu können, muss ein bidirektionaler Datenfluss erfolgen können, da die Erkennung der Emotion aus Text wieder einen Einfluss auf die Nachricht selbst haben kann, wie dies z.B. bei der Extraktion von Emoticons der Fall ist. Aufgrund dessen, dass es möglich ist, neben Text auch andere Informationen von der Lösung bereitstellen zu lassen, ist bei der Definition der Pluginschnittstelle darauf zu achten, dass diese einfach erweiterbar ist. Da eine für den Nutzer verständliche Darstellung der Emotion nicht nur durch die von der Lösung unterstützten Mittel wie der Ausgabe von Textnachrichten möglich ist, muss der empfangene Nachrichtentext inklusive in EmotionML notierter Emotion an einen externen Algorithmus für die weitere Emotionsverarbeitung weitergegeben werden können. Das kann eine spezielle Abwandlung der Pluginschnittstelle bewirken (in Abbildung 16 orange dargestellt). Da Emotionen aus mehreren verschiedenen Quellen über die

Pluginschnittstelle der Lösung zugeführt werden, müssen diese miteinander verarbeitet und zueinander über einen Algorithmus gewichtet werden können, damit diese dann folglich als konkret zu der Chatnachricht eindeutigen Emotionen weitergeben werden können (in Abbildung 16 blau dargestellt).

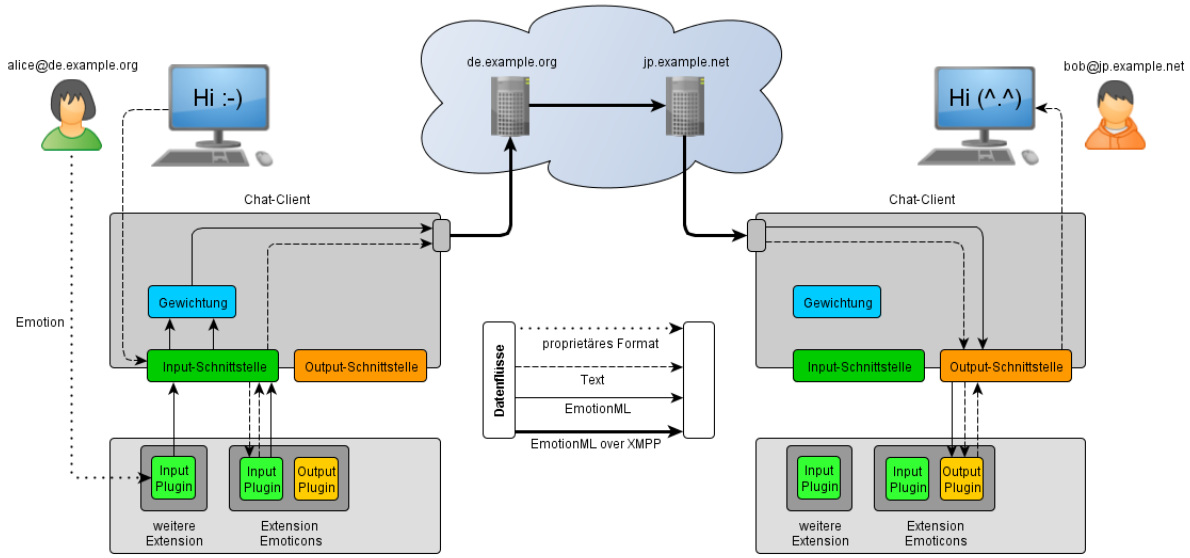


Abbildung 16: Übersichtsbild Systemarchitektur

4.3 Definition eines Chatclients als Prototyp

Um die Umsetzbarkeit der Lösung aufzeigen zu können, wird ein einfacher Chatclient als Prototyp implementiert. Hierbei gilt es vor allem zu berücksichtigen, dass die Systemarchitektur in Kombination mit den verwendeten Protokollen nicht alle Anforderungen an die Lösung erfüllen kann. So lässt sich Anforderung 8 (Darstellung der Emotion beim Empfänger unterstützen) nur umsetzen, wenn auch ein Plugin für die Manipulation der Ausgabe existiert. Hierfür soll im Rahmen der Implementierung eine geeignete Komponente entwickelt werden. Dies trifft auch für eine Komponente zu, die ihren Fokus auf die verbesserte Übertragung emotionaler Aspekte in der interkulturellen Kommunikation legt, wie dies durch Anforderung 5 gefordert wird.

Obwohl immer nur ein Nutzer den Chatclient in Gebrauch haben kann, soll eine Mehrmandantenfähigkeit hergestellt werden. Neben der notwendigen Eingabe von Benutzername und Passwort, um sich mit dem Chatnetzwerk zu verbinden, soll jedem Nutzer ein Verzeichnis zugewiesen werden, in dem alle nutzerrelevanten Informationen von der Anwendung abgelegt werden können. Das primäre Ziel des Chatclients ist es, unformatierten Text und eine Emotion an den Empfänger zu senden. Eine Übertragung mehrerer Emotionen pro Chatnachricht, die gemäß Kapitel 4.1.4 eine Teilung der Nachricht mit sich ziehen würde, soll nicht Teil der Implementierung des Prototyps sein, da hier eine Echtzeitanalyse der Emotion während der Eingabe der Nachricht durch den Nutzer erforderlich wäre. Eine tiefere Interaktion mit angebundenen Systemen zur Emotionserkennung wäre nötig, was die Flexibilität der Lösung im Rahmen des Prototyps einschränken und dem Wesen eines Prototyps nicht gerecht werden würde.

Der Chatclient wird als Windowsanwendung mit einer grafischen Benutzeroberfläche (engl. „Graphical User Interface“ = GUI) implementiert. Um eine einfache Weiterentwicklung und

Verwendung im Sinne des Open-Source-Gedankens zu fördern, wird die Benutzeroberfläche in englischer Sprache gestaltet. Zur Umsetzung der Funktionalität eines Chatclients im Rahmen eines Prototyps sind zwei Komponenten erforderlich. Die in Abbildung 17 durch eine grafische Eingabemaske dargestellte Möglichkeit eines Logins ermöglicht die Autorisierung gegenüber einem XMPP-Server durch die Eingabe von Benutzernamen und Passwort eines bestehenden Benutzerkontos. Erfolgt eine fehlerhafte Eingabe von Benutzernamen und/oder Passwort oder treten sonstige Probleme beim Verbindungsaufbau auf, so ist eine Fehlermeldung mit Hilfe einer Dialogbox[33] dem Nutzer mitzuteilen. Die Übermittlung der eingegebenen sensiblen Daten des Nutzers ist über eine gesicherte Verbindung zu realisieren, um schon im Stadium des Prototyps Aspekten des Datenschutzes gerecht zu werden. Ist die Autorisierung gegenüber dem XMPP-Server erfolgreich abgeschlossen, erfolgt eine automatische Schließung des Loginformulars und der eigentliche Chatclient übernimmt die weitere notwendige Interaktion mit dem Nutzer. Eine exemplarische Oberfläche für diesen ist in Abbildung 18 dargestellt.

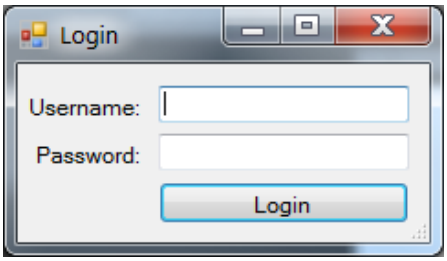


Abbildung 17: Loginformular Chatclient

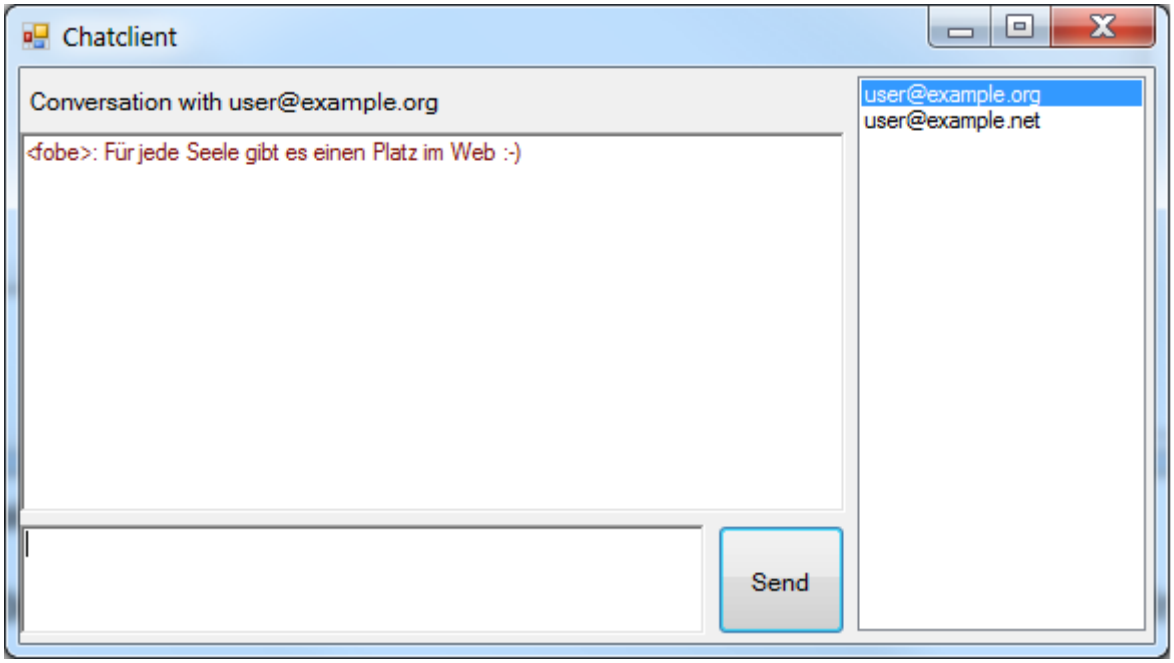


Abbildung 18: Oberfläche des Chatclients

In dem Chatclient müssen primär drei Steuerelemente aufzufinden sein:

1. Eine Liste von mit dem Benutzerkonto verbundenen Kontakten, die eine Wahl des Empfängers einer Nachricht ermöglicht (Liste rechts in Abbildung 18)
2. Ein Eingabefeld für Text, der an einen Empfänger übermittelt werden soll (in Abbildung 18 Eingabefeld unten)
3. Ein Darstellungsbereich, welcher den Verlauf eingehender und ausgehender Nachrichten abbildet (großes Textfeld links oben in Abbildung 18)

[33] Ein im Vordergrund befindliches Fenster, welches dem Nutzer präsent Informationen in den Vordergrund übermittelt und von diesem bestätigt werden muss.

Die verbundenen Kontakte sind dabei vom XMPP-Server abzufragen und verfügbare Kontakte in der Liste darzustellen. Erfolgt eine Verfügbarkeitsänderung verbundener Kontakte (sie kommen online oder gehen offline) ist diese Liste entsprechend zeitnah zu aktualisieren. Die in das Eingabefeld für Nachrichten eingetragene Mitteilung an den Chatpartner wird durch Drücken eines Knopfes an diesen versendet. Um die Nutzbarkeit der Anwendung zu erhöhen, kann dieser Versand auch durch die Betätigung der Eingabetaste angestoßen werden. Die Nachricht wird nach erfolgreichem Versand im Darstellungsbereich für ein- und ausgehende Nachrichten notiert. Der Sender einer Nachricht ist dabei mit einer Trennung gefolgt vor der Darstellung seiner übersendeten Nachricht zu notieren. Wird eine weitere Nachricht in diesem Bereich dargestellt, so erfolgt eine Trennung von der vorherigen durch einen Zeilenumbruch. Zur besseren Unterscheidung von Benutzer und Chatpartner sind die Nachrichten der Benutzer unterschiedlich farblich darzustellen, wobei jedem Chatpartner eine genau eine dynamisch vergebene Farbe zugewiesen wird.

4.4 Definition einer Pluginschnittstelle zur Emotionsverarbeitung

Wie in Kapitel 4.2 durch die Systemarchitektur unter Zuhilfenahme von Abbildung 16 beschrieben wurde, ist eine leistungsfähige Pluginschnittstelle für die Implementierung der Lösung nötig. Diese muss eine bidirektionale Verbindung zu vorgeschalteten Algorithmen herstellen können und dabei ein Auslesen und Verändern des vom Nutzer eingegebenen Textes vor dem Versand ermöglichen. Weiterhin müssen Emotionen und Text, die beim Empfang einer Nachricht vom Chatclient aus dieser extrahiert werden, zur weiteren Verarbeitung über die Pluginschnittstelle an auswertende externe Algorithmen weitergegeben werden. Diese können dann wieder eine Beeinflussung der Textanzeige im Chatclient des Empfängers bewirken oder die Emotion anderweitig außerhalb des Programms beliebig darstellen. Daher findet, wie dies in Abbildung 16 zu sehen ist, eine Unterscheidung von **Input-Plugins und Output-Plugins** statt. Ein Input-Plugin ist dabei für die Verarbeitung aller Emotionen verantwortlich, die der Anwendung zugeführt werden, und sorgt gleichzeitig dafür, dass nur in EmotionML abgebildete Emotionen der weiteren Verarbeitung durch die Anwendung zugeführt werden. Ein Output-Plugin hingegen ist verantwortlich für die Weitergabe empfangener Daten wie Emotion und Text an externe Programme. Beide Plugin-Sorten bilden dabei für die Anwendung eine Schnittstelle zu einem evtl. vorgeschalteten emotionsverarbeitenden Algorithmus, so dass diese über eine definierte Schnittstelle mit diesem Daten austauschen kann. Weiterhin ermöglichen sie so eine bidirektionale Verbindung zwischen Anwendung und dem Algorithmus, um den Text der Chatnachricht unter emotionalen Gesichtspunkten verarbeiten zu können.

4.4.1 Extensions und Plugins

Um die Beeinflussung von Plugins verschiedener Hersteller untereinander auszuschließen, werden ein oder mehrere Plugins zu hersteller- bzw. aufgabenspezifischen Paketen gebündelt. Diese Pakete werden **Extensions** genannt. Die Anwendung kann also durch mehrere Extensions erweitert werden, die dann wiederum mehrere Plugins beinhalten können, welche dann selbst ein Input-Plugin oder ein Output-Plugin darstellen. Die Abschottung der einzelnen Plugins und vor allem der einzelnen Extensions voneinander erfolgt dabei über ein festgelegtes Namensraumschema. Erfolgt bei der Entwicklung der Extension keine komplette Beachtung dieses Namensraumschemas, kann ein die Funktionsfähigkeit der Extension nicht sichergestellt werden.

Abbildung 19 verdeutlicht anhand der Ordnerstruktur der Anwendung den Aufbau einer Extension im Dateisystem. Die für eine Extension relevanten Bestandteile der Ordnerstruktur sind fett hervorgehoben. Zu jeder Extension gehört im Verzeichnis *extensions* ein Ordner, der den eindeutigen Schlüssel der Extension aufweist. Dieser Name – in Abbildung 19 blau hervorgehoben – bildet dabei die unterscheidbare Basis des Namensraumes von Extensions und muss in URL-kodierter Form vorliegen (vgl. [20] Abschnitt 2). Somit ist ausgeschlossen, dass zwei Extensions gleichen Namens der Anwendung zur Verfügung stehen. Eine zentrale Vergabe der Extensionnamen erfolgt nicht. Um hier Konflikte bei der Benennung von Extensions zu vermeiden, wird auf die Kooperation der entwickelnden Hersteller gesetzt. Gleiches System hat sich bereits bei anderen Open-Source-Projekten wie bei den Plugins für die Produkte Firefox und Thunderbird der Mozilla Foundation bewährt[34]. Weiterhin kann durch Umbenennung der Extension ein möglicher Konflikt behoben werden.

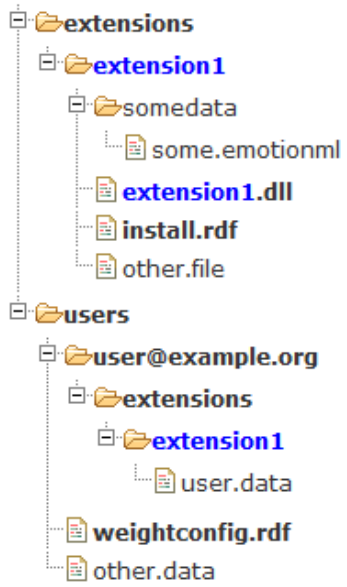


Abbildung 19: Ordnerstruktur für Erweiterungen

Im Verzeichnis einer Extension sind primär zwei Dateien aufzufinden. Zum einen existiert dort eine Datei *install.rdf*. Dies ist eine Konfigurationsdatei, die die Extension mit ihren Plugins und diversen Einstellungen und Informationen bei der Anwendung anmeldet. Nähere Informationen hierzu sind in Kapiteln 4.4.2 und 4.4.3 notiert. Weiterhin ist eine Dynamic Link Library (DLL) vorhanden, deren Dateiname dem Schlüssel der Extension entspricht. Diese stellt mittels des in Kapitel 4.4.3 beschriebenen Verfahrens den Einsprungpunkt von der Anwendung in die Extension dar. Weiterhin wird jeder Extension im Verzeichnis *users* eine Speichermöglichkeit eingeräumt. Diese befindet sich im nutzerspezifischen Bereich in einem Unterverzeichnis des Ordners *extensions* und weist ebenfalls den Schlüssel der Extension als Ordnernamen auf. Hier ist es für eine Extension möglich nutzerspezifische Einstellungen zu speichern. Somit besteht die Möglichkeit, dass der Nutzer sein Verzeichnis entnehmen und mit allen Einstellungen einem anderen Client zuführen kann. Neben den hier aufgeführten Restriktionen in Bezug auf Extensions, deren Dateien und Speicherbereichen, ist eine mögliche weitere Einteilung von den der Extension zugewiesenen Ordnern dem Hersteller der Extension überlassen.

[34] Hier bekommen die Plugins eine interne Identifikation, die vom Entwickler frei gewählt werden kann.

4.4.2 Konfigurationsdateien

Die *install.rdf*, welche jeder Extension direkt im Extensionverzeichnis beiliegen muss, bietet eine nähere Beschreibung der Extension und tätigt somit eine Konfiguration der nutzbaren Plugins gegenüber der eigentlichen Anwendung. Um hier eine gute Erweiterbarkeit und Flexibilität gewährleisten zu können, wird die Konfigurationsdatei in RDF-XML notiert. Somit wird eine einfache Verwendung in verschiedenen Programmen nach Anforderung 9 dahingehend ermöglicht, dass die Konfiguration offenen Standards folgt und von Programmen semantisch verarbeitbar ist. Auch in dieser Konfigurationsdatei spielt der Namensraum der Extension eine entscheidende Rolle. Die Konfiguration von Extensions und deren Plugin wird dabei im Pfad *urn:emotionchat:config:plugins* notiert. Aufgrund der lokalen Abgeschiedenheit und dem möglichen Einsatz in nicht öffentlich zugänglichen Netzen, wird eine Benennung durch einen URN (Uniform Resource Name) gewählt. Die Benennung der URI, die als Bezeichnung des Subjektes der jeweiligen RDF-Tripel dient, folgt dabei dem festen Benennungsschema *urn:emotionchat:config:plugins:[Extensionname]:[Pluginname]*. Somit sind immer mindestens zwei RDF-Subjekte vorhanden, nämlich eines mit Informationen über die Extension selbst ergänzt durch die Beschreibungen der jeweiligen Plugins. Die Extension selbst wird durch allgemeine Eigenschaften beschrieben. Dabei kommen primär die Konventionen für Metadaten der Dublin Core Metadata Initiative [22] zum Einsatz. Diese und weitere in diesem Kontext sinnvolle Prädikate könnten Tabelle 7 entnommen werden. Eine Erweiterung durch zusätzliche Ontologien ist hier möglich. Ein für die Anwendung spezifisches Prädikat stellt hierbei *emochat:version* dar, womit eine Version für die Extension angegeben werden kann.

Prädikat	**Aussage**
dcterms:title	Titel
dcterms:description	Beschreibung
dcterms:creator	Autor
dcterms:issued	Veröffentlichungsdatum
dcterms:modified	Letzte Änderung
dcterms:rights	Rechtliche Elemente
foaf:homepage	Projekthomepage
emochat:version	Version

Tabelle 7: Mögliche RDF-Properties einer Extension

Ein Plugin erbt programmintern alle nach Tabelle 7 gemachten Angaben über die Extension, kann diese aber durch erneute Definition überschreiben. Gemäß Tabelle 8 werden weitere Prädikate definiert. Um die Plugins semantisch korrekt der entsprechenden Extension zuzuweisen, wird empfohlen mittels *dcterms:isPartOf* von der Pluginkonfiguration auf die Extensionkonfiguration zu verweisen. Dies soll jedoch für die Funktionsweise der Anwendung nicht nötig sein.

URI	**Unterelement von**	**Beschreibung**
dcterms:isPartOf	Pluginkonfiguration	Zuordnung eines Plugins zu einer Extension
emochat:stream	Pluginkonfiguration	Einleitung eines Input- oder Output-Plugins
emochat:Input	emochat:stream	Plugin ist ein Input-Plugin
emochat:Output	emochat:stream	Plugin ist ein Output-Plugin
emochat:url	emochat:Input emochat:Output	URL an welche die Emotion gesendet wird

Tabelle 8: Erweiterte Konfigurationsmöglichkeiten für Plugins

Unbedingt notwendig ist jedoch die Definition, ob es sich um ein Input- oder Output-Plugin handelt. Diese wird über das Prädikat *emochat:stream* eingeleitet. Diesem RDF-Knoten wird der RDF-Typ *emochat:Input* zugeordnet, wenn es sich um ein Input-Plugin handelt, *emochat:Output* sollte es ein Output-Plugin sein. Um das Wesen dieser Pluginkonfiguration weiter zu spezifizieren, kann ihm ein weiterer RDF-Typ gemäß Tabelle 9 zugeordnet werden. Erfolgt eine Typisierung mit dem RDF-Typ *urn:emotionchat:config:type:Plugin*, so wird das entsprechende Plugin der Extension genutzt. Diese Pluginart externer Plugins kann jedoch durch Anwendung eingebaute Plugins genutzt werden, wie das bei einem speziellen mit dem RDF-Typ *urn:emotionchat:config:type:Url* definierten **Urlplugin** der Fall ist.

Typ	Beschreibung
urn:emotionchat:config:type:Plugin	Emotionsverarbeitung erfolgt über ein klassisches Plugin in der DLL
urn:emotionchat:config:type:Url	Emotionen werden an eine URL gesendet oder von dieser empfangen

Tabelle 9: Erweiterte Typen für Plugins

Dieses bildet den häufigen Anwendungsfall ab, dass das Empfangen einer in EmotionML kodierten Emotion durch den Aufruf einer beliebigen URL geschehen soll. Um dieses Nutzungsszenario umsetzen zu können, wird eine entsprechende URL mit Hilfe des Prädikates *emochat:url* angegeben. Weiterhin können in der URL Platzhalter gemäß Tabelle 10 definiert werden, die dann beim Aufruf entsprechender URL durch die Anwendung mit Inhalten gefüllt werden. Somit hat diese Konfiguration eine Templatefunktion inne. Ist das Plugin als Output-Plugin definiert, so geschieht die Übertragung des EmotionML im Körper des HTTP-Paketes mit Hilfe der POST-Methode. Bei einer Konfiguration als Input-Plugin erfolgt der Zugriff auf die URL via GET.

Platzhalter	Bedeutung
%JID%	Wird ersetzt mit JID des Nutzers
%MESSAGE%	Wird ersetzt mit Nachricht des Nutzers

Tabelle 10: Platzhalter in URL beim Url-Plugin

4.4.3 Interaktion zwischen Chatclient und Plugins

Damit Plugins mit der Anwendung Emotionen austauschen können, muss eine entsprechende Schnittstelle definiert werden. Da es sich hier um eine öffentliche Schnittstelle handelt, ist es empfehlenswert, dass diese in zukünftigen möglichen Programmversionen keine Änderung erfährt, um somit notwendig werdende Abwärtskompatibilitäten zu umgehen. Trotzdem muss eine einfache Erweiterung später möglich sein. Weiterhin empfiehlt sich eine Minimierung der notwendigen Restriktionen, um eine einfache Entwicklung von Plugins und Extensions zu gewährleisten. Durch die Konfiguration in der *install.rdf* der Extension ist jedem Plugin ein eindeutiger Schlüssel innerhalb dieser zugeordnet, so dass es hierüber angesprochen werden kann. Um nur einen klaren Einsprungpunkt in die DLL für alle Plugins zu haben, wird ein Manager definiert, der vom Extensionentwickler in der Extension implementiert werden muss. Diesem Manager wird dann der Schlüssel des Plugins übergeben und er liefert eine entsprechende Plugininstanz zurück, die dann von der Anwendung genutzt werden kann. Somit bleibt auch die, in der Extension intern verwendete, Verwaltung der Plugins dem Extensionentwickler freigestellt. Weiterhin besteht so die Möglichkeit, dass mit Hilfe dieses Managers Ressourcen für mehrere Plugins gemeinsam genutzt werden können. Abbildung 20 zeigt ein entsprechendes Klassendiagramm, welches die Aufgabenverteilung zwischen Extension und Anwendung verdeutlicht. Die orangenen

Bestandteile sind dabei Klassen und Interfaces, die von der Anwendung bereitgestellt werden. Die türkis dargestellten Klassen werden von der in der DLL befindlichen Extension über den Manager bereitgestellt.

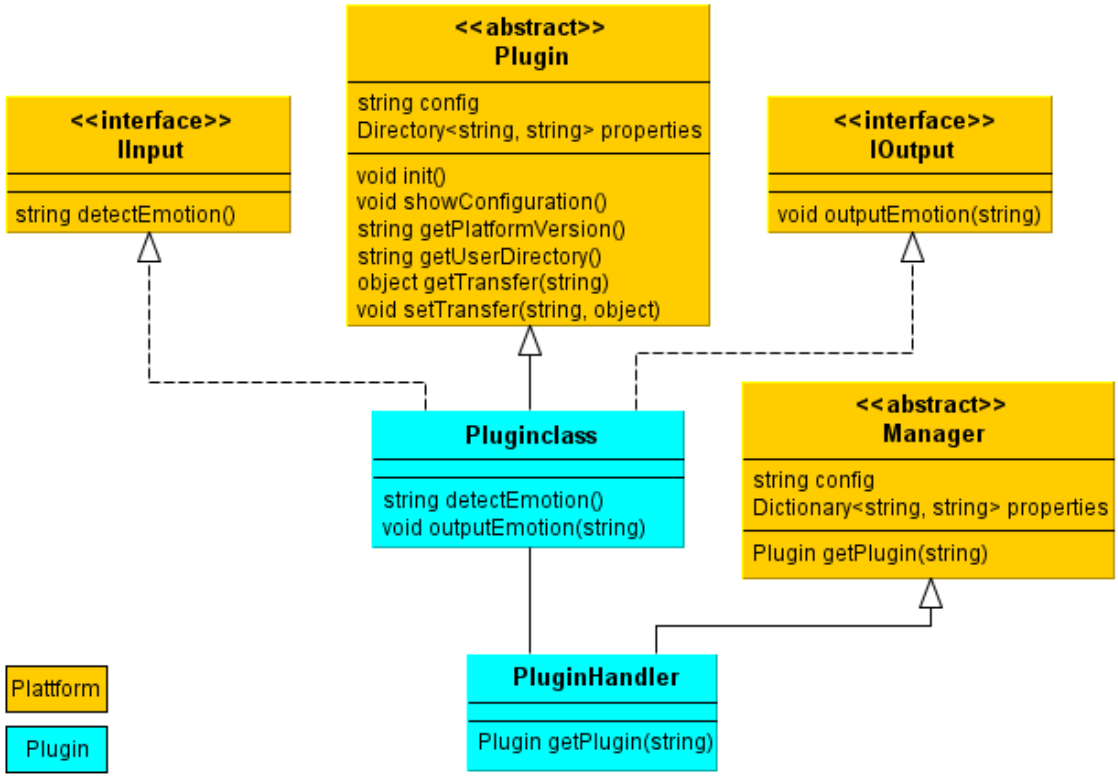


Abbildung 20: Klassendiagramm Pluginschnittstelle

Die Hauptaufgabe des Managers, der mit der Beispielklasse *PluginHandler* durch den Extensionentwickler implementiert wird, ist die Kreation von Plugininstanzen für die Anwendung. Hierfür muss *PluginHandler* die abstrakte Klasse *Manager* implementieren. Dadurch ist es möglich, dass die Anwendung die Methode *getPlugin* des Managers aufruft und dieser als Parameter den Namen eines Plugins übergibt. Diese Methode gibt eine Instanz eines Plugins zurück, welche von der abstrakten Klasse *Plugin*, die durch die Anwendung bereitgestellt wird, erben muss. Diese abstrakte Klasse beinhaltet Möglichkeiten, damit das Plugin auf Daten der Anwendung zugreifen kann. Die Konfiguration des Plugins wird bei dessen Initialisierung durch die Anwendung in den Attributen der Klasse abgelegt. Eine Besonderheit beim Datenaustausch stellen die Methoden *getTransfer* und *setTransfer* dar. Durch sie ist ein Austausch beliebiger Informationen zwischen den Plugins über die Anwendung mit Hilfe von Schlüssel-Wert-Paaren möglich. Dies kann u.a. dazu genutzt werden, einen vom Nutzer eingegebenen Text für die erkennungstechnische Emotionsanalyse zu empfangen und entsprechend zu modifizieren. Die Plugininstanz kann also transparent eigene Methoden aufrufen und somit Zugriff auf die Daten der Anwendung erlangen. Weitere Informationen über die einzelnen Methoden der Klasse Plugin sind in Tabelle 11 ersichtlich. Durch die Existenz der abstrakten Klasse *Plugin* außerhalb des Plugins, kann die Schnittstelle zu den Plugins seitens der Anwendung bei deren Weiterentwicklung ebenfalls so erweitert werden, dass bis dahin implementierte Plugins weiterhin unterstützt werden können. Damit eine Initialisierung des Plugins möglich ist, wird

eine Methode *init* als Hook[35] definiert. Somit ist es notwendig, dass diese Methode des Plugins aufgerufen wird, bevor mit dem Plugin zum ersten Mal Daten ausgetauscht werden.

Methode	Beschreibung
init	Hook für Initialisierungsprozess des Plugins.
showConfiguration	Hook zum Anzeigen eines Konfigurationsdialog
getPlatformVersion	Liefert die Version der Anwendungsplattform
getUserDirectory	Liefert den Pfad zu dem für die Extension vorgesehenen Verzeichnis im Nutzerverzeichnis
getTransfer	Greift auf mit anderen Plugins und/oder der Anwendung gemeinsam genutzten Informationen mit Hilfe des im Parameters angegebenen Schlüssels zu
setTransfer	Modifiziert die unter dem im ersten Parameter angegebenen Schlüssel abgelegten gemeinsam mit Plugins und/oder Anwendung genutzten Informationen

Tabelle 11: Methoden der abstrakten Pluginklasse

Die primäre Aufgabe von Plugins besteht im Austausch n EmotionML kodierter Emotionen mit der Anwendung. Damit dies möglich ist, ist es nötig, dass das Plugin die Interfaces *IInput* und/oder *IOutput* implementiert. Um einer Verwechselung von Input-Plugins und Output-Plugins entgegenzuwirken und die Verwendung eines Plugins sowohl als Input-Plugin wie auch als Output-Plugin zu ermöglichen, werden zwei separate Interfaces definiert. *IInput* gibt an, dass es sich um ein Input-Plugin handelt, welches durch die Methode *detectEmotion* der Anwendung eine in EmotionML kodierte Emotion zuführt. Im Gegenteil dazu wird einem Plugin, welches *IOutput* implementiert, über die Methode *outputEmotion* eine in EmotionML kodierte Emotion übergeben, die vom Nutzer empfangen wurde.

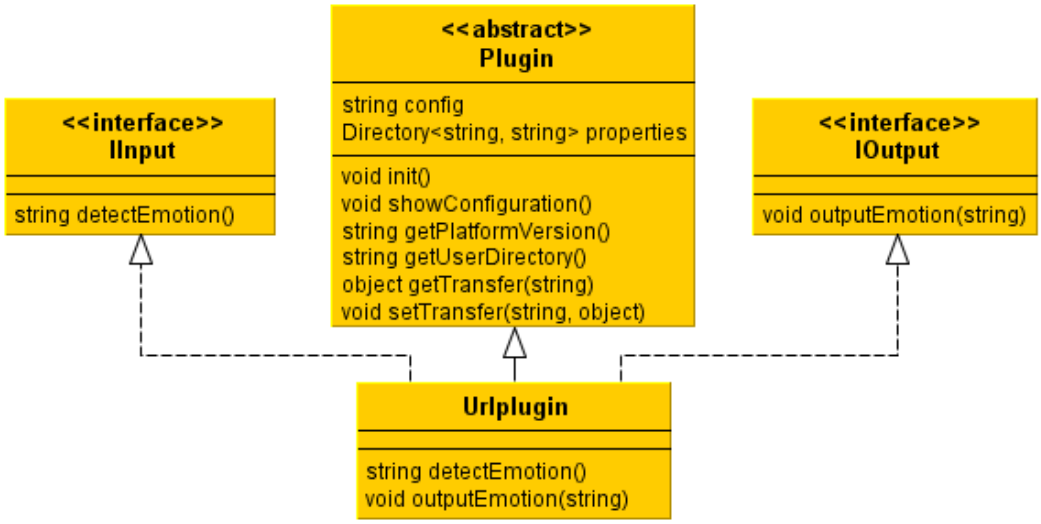


Abbildung 21: Klassendiagramm für Urlplugin

Soll nun, wie in Kapitel 4.4.2 beschrieben, die Emotion von einer URL abgerufen bzw. an eine URL gesendet werden, so verhält sich die gerade vorgestellte Pluginschnittstelle anders, da dann eine interne Programmerweiterung vorzufinden ist und keine externen

[35] Eine Methode ohne Inhalt, die von ableitenden Klassen überschrieben werden kann. Sie wird im Programmverlauf aufgerufen, so dass damit in der ableitenden Klasse zusätzliche Codebestandteile bspw. für Initialialisierungen oder Kontrollen eingebracht werden können, es aber nicht müssen.

Programmbestandteile genutzt werden. Eine für die Plugininstanzverwaltung notwendige Verwendung eines Managers ist somit nicht notwendig. Die dem Klassendiagramm aus Abbildung 21 dargestellte Klasse *Urlplugin* ersetzt die Pluginklasse im ersten Szenario und implementiert die Interfaces *IInput* und *IOutput*. Somit weist diese interne Pluginimplementierung die gleiche Grundstruktur wie eine Externe auf und kann somit der Programmabarbeitung transparent zugeführt werden.

4.5 Gewichtung von Emotionen anhand einer Nutzerkonfiguration

Aufgrund der Tatsache, dass Emotionen aus mehreren Quellen gemäß Anforderung 7 für das Instant Messaging genutzt werden sollen, ist es nötig die Emotionen der verschiedenen Quellen über eine geeignete Funktionalität zu einer konkreten Emotion zusammenzuführen. Wie in Kapitel 3.2.3 festgestellt wurde, gibt es sehr viele Möglichkeiten der Emotionsäußerung und nicht jeder Mensch ist dazu in der Lage, seine Emotionen gleichwertig über jeden Kommunikationskanal zu äußern. Daher ist bei der Aggregation der Emotionen darauf zu achten, dass eine Gewichtung gemäß der Fähigkeiten des Nutzers erfolgt. Um dies zu gewährleisten, ist eine nutzerspezifische Konfiguration der Gewichtung der aus den einzelnen Plugins aggregierten Emotionen nötig. Zu diesem Zweck wird im Nutzerverzeichnis des angemeldeten Nutzers die Konfigurationsdatei *weightconfig.rdf* abgelegt (zur Verdeutlichung siehe Abbildung 19). Diese Konfigurationsdatei ist wie schon die Pluginkonfiguration in RDF verfasst. Die Basis bildet eine BlankNode[36], der über das Prädikat *emochat:weighting* eine Liste von BlankNodes zugeordnet ist, die dann jeweils ein Schlüssel-Wert-Paar für die Konfiguration der Gewichtung eines jeden Plugins haben. Über das Prädikat *emochat:plugin* wird der BlankNode das Plugin zugewiesen, bei dem die im Prädikat *emochat:weightvalue* angegebene Gewichtung bei der Berechnung der Gesamtemotion Anwendung finden soll. Je höher die Zahl, desto mehr Gewicht hat die durch das Plugin zugeführte Emotion. Entscheidend ist jedoch das Verhältnis zwischen den einzelnen Gewichtungskonfigurationen der Plugins zueinander. Ein Beispiel einer Datei für die Konfiguration der Gewichtung zweier Plugins ist in Anhang C notiert.

Für die Berechnung des resultierenden Emotionswertes sind drei Parameter je Plugin von Bedeutung:

1. **Der erkannte Emotionswert**: Der durch das Plugin zugeführte Wert der Intensität des Emotionsbestandteiles (vgl. value-Attribut in EmotionML). Ist dieser Wert nicht in dem vom Plugin zurückgegebenen EmotionML beinhaltet, so wird für die Berechnung ein Wert von 0.5 angenommen.

2. **Die Erkennungssicherheit**: Die durch das Plugin zugeführte Güte der Erkennung des Emotionsbestandteiles (vgl. confidence-Attribut in EmotionML). Wird dieser Wert nicht im EmotionML des Plugins angegeben, so wird auch hier für die Berechnung ein Wert von 0.5 angenommen.

3. **Die Nutzerspezifische Gewichtung**: Dies ist der Wert, welcher in der nutzerspezifischen Gewichtungskonfiguration festgelegt wurde. Ist für ein Plugin,

[36] Eine BlankNode ist ein RDF-Knoten, dem kein URI zugewiesen wurde und der selbst kein Prädikat darstellt.

welches der Anwendung eine Emotion in EmotionML zuführt keine Konfiguration angegeben, so wird hier ein Wert von 50 angenommen.

Für eine erfolgreiche Bearbeitung ist zu beachten, dass die Emotionen des EmotionML dem selben Set (category-set, dimension-set, …) zugeordnet sein müssen. Sind mehrere Sets vorhanden, so wird dies mit den meisten zugeführten Emotionen genommen. Bei der Berechnung ist weiterhin zu beachten, dass nur die Plugins in die Berechnung eingehen, die zum betrachteten Zeitpunkt der Anwendung eine für die Berechnung relevante Emotion zugeführt haben. Wurde vom Plugin keine verwert- bzw. vergleichbare Emotion eingebracht, so fällt es aus der Berechnung heraus. Daraus ergibt sich folgende Berechnungsformel:

$$Emotionswert_{Gesamtemotion} = \sum_{i=1}^{\substack{Anzahl \\ Plugins}} \frac{Emotionswert_i * Gewicht_i * Erkennungssicherheit_i}{\sum_{k=1}^{Anzahl\ Plugins} Gewicht_k * Erkennungssicherheit_k}$$

4.6 Zusammenfassung Lösungskonzept

Im Rahmen des Lösungskonzepts ist eine flexible Systemarchitektur entstanden, die sehr gut erweitert werden kann. Einer im Fokus stehenden Anwendung werden mit Hilfe vorgelagerter Plugins in EmotionML abgebildete Emotionen zugeführt. Dabei können über eine erweiterbare Schnittstelle zwischen Plugins und Anwendung Daten bidirektional ausgetauscht werden. Nachdem die Emotionen über Plugins zugeführt wurden, erfolgt eine Bündelung der verschiedenen Eingaben zu einer aggregierten und in EmotionML abgebildeten Emotion. Dabei findet gleichzeitig eine nutzerspezifische Gewichtung untereinander statt. Neben der Zuführung von Emotionen in die Anwendung mithilfe der Plugins ist auch deren Auslieferung von der Anwendung an die Plugins möglich, so dass eine geeignete Darstellung der Emotion realisiert werden kann.

Weiterhin wird im Lösungskonzept ein Prototyp in Form eines Chatclients definiert. Dieser nutzt dabei den vorgestellten Prozess der Emotionsaggregation und -repräsentation. Da eine Analyse ergeben hat, dass XMPP für die Übertragung von Chatnachrichten und EmotionML für die Abbildung von Emotionen am besten geeignet sind, erfolgt eine Kombination dieser beiden Protokolle, so dass eine in EmotionML notierte Emotion in eine XMPP-Nachricht eingebettet werden kann.

5 Implementierung

Die Umsetzbarkeit der erarbeiteten Lösung soll im weiteren Verlauf der Arbeit durch den Einsatz eines Prototyps gezeigt werden. Hierfür wird ein in Kapitel 4.3 vorgestellter Chatclient implementiert, der neben einen Benutzertext auch Emotionen mit Hilfe des in Kapitel 4.1.4 definierten Protokolls versenden kann. Um die Umsetzung der Implementierung zu erleichtern, werden zu Beginn geeignete und nutzbare Programmbibliotheken bestimmt. Danach erfolgt ein softwaretechnischer Überblick. Damit eine bessere Übersicht und Wiederverwendbarkeit der Komponenten ermöglicht werden kann, wird die Software in zwei Komponenten eingeteilt. Dies ist zum einen eine Plattform, welche die Interaktion mit den in Kapitel 4.4.1 vorgestellten Extensions und Plugins realisiert und die Emotionen gemäß der in Kapitel 4.5 definierten Vorschrift zusammenfasst und gewichtet. Zum anderen ist dies der Chatclient als Anwendung, der diese Emotionen zur weiteren Verarbeitung nutzt. Im Anschluss dessen erfolgt die Beschreibung zweier Beispielplugins. Um eine flexible

Weiternutzung der Software im Rahmen des Open-Source-Gedankens zu ermöglichen, wird zuletzt eine Betrachtung zur Wiederverwendbarkeit der Bestandteile der Softwarelösung durchgeführt.

Für den praktischen Einsatz der Anwendung ist eine möglichst breite Unterstützung vorhandener Softwaresysteme zur Emotionserkennung förderlich. Da diese, wie Kapitel 3.2.3 zeigt, oft direkt Sensoren beim Benutzer benötigen, wird die Implementierung als Desktop-Anwendung umgesetzt. Eine Umsetzung als browserbasierte Webanwendung würde aufgrund der Abgeschlossenheit des Browserinhalts gegenüber dem Betriebssystem des Benutzers diese Interaktion erschweren, da zusätzliche Browser-Plugins installiert oder entsprechende Schnittstellen im Browser definiert und umgesetzt werden müssten. Eine Abbildung mehrerer Emotionen pro Nachricht sollen nicht Bestandteil der Implementierung des Prototyps sein, da für die zu diesem Zweck benötigte Echtzeitanalyse eingehender Emotionen ein tieferes Zusammenspiel mit der jeweiligen Software der Emotionserkennung notwendig wäre. Dies jedoch übersteigt das Wesen eines Prototyps. Um die Flexibilität der Anwendungsplattform weiter zu steigern, indem eine gleichzeitige Unterstützung mehrerer Anwendungen ermöglicht wird, wird die Plattform durch einen Singleton[37] implementiert. Durch dieses Vorgehen ist die Existenz einer Startklasse nötig, in der die Programmausführung beginnt und die Anwendung instanziiert wird. Die Implementierung erfolgt in der Programmiersprache C#, da die betreuende Professur diese Sprache für eigene Implementierungen bevorzugt einsetzt.

Ein bevorzugter Einsatz erfolgt auch bei einer Programmbibliothek für die Verarbeitung von RDF zugunsten von *dotNetRDF*[38]. Auf vielfältige Art und Weise erleichtert diese die Entwicklung und Interaktion mit RDF. Im Rahmen der Anwendung werden primär die Möglichkeiten des Frameworks zur Verarbeitung von und Navigation zwischen RDF-Tripeln sowie dem Lesen und Schreiben von RDF-XML genutzt. Das beispielhafte Laden einer RDF-XML-Datei sowie die Erstellung einer Liste von Tripeln um ein Subjekt wird in Quelltext 11 dargestellt. Hier werden je eine Instanz eines Graphen und eines RDF-XML-Parsers instanziiert. Folgend lädt der Parser das RDF aus der Datei example.rdf in den Graphen. Über dessen Methode *GetUriNode*, wird der entsprechende RDF-Knoten – in diesem Beispiel http://example.com/ – zurückgegeben. Mit diesem kann unter Zuhilfenahme der Methode *GetTriplesWithSubject* des Graphen eine Liste mit Tripeln erstellt werden, die diesen RDF-Knoten als Subject aufweisen.

```
Graph graph = new Graph();
RdfXmlParser rdfParser = new RdfXmlParser();
rdfParser.Load(graph, "example.rdf");
IUriNode node = graph.GetUriNode("http://example.com/");
List<Triple> tripleCollection =  new List<Triple>(
      graph.GetTriplesWithSubject(node)
);
```

Quelltext 11: Beispiel dotNetRDF

[37] Ein in der Softwareentwicklung genutztes Entwurfsmuster, welches sicherstellt, dass zu einer Klasse maximal ein Objekt existiert

[38] Weitere Informationen sowie Dokumentation sind auf der Projektseite unter http://www.dotnetrdf.org/ verfügbar. (Abruf 07.09.2012)

5.1 EmotionML-Bibliothek

Da die Anwendung intensiv mit EmotionML [4] arbeitet, empfiehlt sich eine Programmbibliothek für EmotionML als Unterstützung, die EmotionML erstellen und in Objektinstanzen überführen kann. Recherchen konnten keine Bibliothek für C# entdecken, die für die Umsetzung der Anwendung genutzt werden kann. Dies ist auch nicht verwunderlich, da der Standard recht speziell und noch in der Entwicklung ist. Um die freie Nutzung auch mit anderen Projekten zu ermöglichen (siehe Anforderung 9), wurde beschlossen selbst eine EmotionML-Bibliothek für C# zu implementieren. Da für die Entwicklung des Prototyps große Teile von EmotionML genutzt werden können und daher umgesetzt werden müssen, ist es nur noch ein geringer zusätzlicher Aufwand den Standard möglichst vollständig zu implementieren. Dies betrifft primär die Abbildung der Vokabulare für EmotionML sowie die in Verbindung mit Zeiten stehenden Attribute des <emotion>-Elementes. Zusätzlich hat die EmotionML betreuende W3C Multimodal Interaction Working Group aufgrund der Veröffentlichung von EmotionML als Candidate Recommendation einen „Call for Implementations“ gestartet [94] und Testfälle für einen „Implementation Report“ veröffentlicht [95]. Weiterhin wurde der Endtermin hierfür noch einmal hinausgezögert [96], um weitere Einsendungen zu erhalten. Somit bietet es sich an, die für den Prototyp entwickelte EmotionML-Bibliothek an dieser Stelle vorzustellen, und im Anschluss an die Ausarbeitung der Öffentlichkeit bspw. über die Plattform github[39] zur Nutzung zugänglich zu machen.

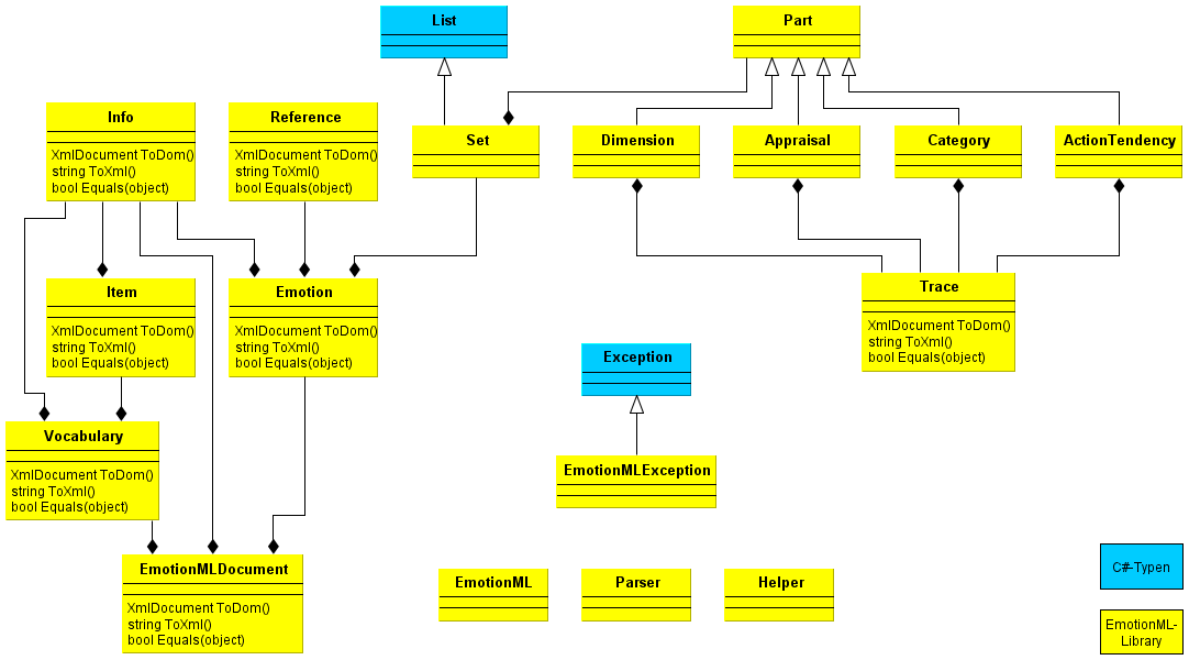


Abbildung 22: Vereinfachtes Klassendiagramm EmotionML-Bibliothek

Wie in Kapitel 3.2.2.2 betrachtet wurde, lässt sich EmotionML auf zweierlei Arten nutzen. Auf der einen Seite existiert EmotionML als Dokumentenformat, was zu eigenständigen Dokumenten in EmotionML führt. Auf der anderen Seite soll EmotionML als Plugin-Sprache fungieren, die in andere Sprachen eingebettet und somit mit diesen kombiniert werden kann. Daher muss die Verarbeitung von EmotionML sowohl dokumentbasiert als auch emotionsbasiert möglich sein. Wie das vereinfachte Klassendiagramm in Abbildung 22 zeigt,

[39] Ein sehr bekannter Hosting-Dienst für Software-Entwicklungsprojekte unter https://github.com/

spielt die Komposition im EmotionML eine große Rolle, da Bestandteile wie der <info>-Bereich mehrfach genutzt werden oder Elemente wie <category>, <dimension>, <appraisal> und <action-tendency> einen sehr ähnlichen Aufbau aufweisen. Somit findet sich diese Tatsache auch in der Vererbungshierarchie der Programmbibliothek wieder.

Allgemein kann festgestellt werden, dass es günstig ist jedes XML-Element durch eine eigene Klasse in der Implementierung zu repräsentieren. Ein EmotionML-Dokument kann aus mehreren Emotionen und Vokabularen bestehen. Da es zu beiden Typen eigene Klassen gibt, ist die Speicherung in einer Liste kein Problem in *EmotionMLDocument*. Ein wenig anders verhält es sich jedoch bei einer Emotion, die wiederum mehrere Kategorien etc. beinhalten kann. Diese in einer einfachen Liste zu speichern ist ungünstig, da ihnen immer auch ein Set (bei Kategorien also ein category-set) zugeordnet ist. Somit ist ein speziell erweiterter Listentyp nötig, der diese Referenz zu einem Vokabular mit abbilden kann. Weiterhin existiert eine Klasse *EmotionMLException* zur Behandlung von Ausnahmen. Diese einfache Ableitung der C#-eigenen Klasse *Exception* dient primär der besseren Unterscheidung, dass die Exception im Bereich der Programmbibliothek auftrat, so dass dies in einem Stacktrace[40] leichter ersichtlich ist und gezielter behandelt werden kann. EmotionML besteht jedoch nicht nur aus XML-Elementen an sich. Es gibt zu dieser Auszeichnungssprache weitere Metainformationen wie Version, empfohlene Dateiendung oder Namensraum mit empfohlenem Präfix. Diese Informationen werden als Konstanten zusammen mit einigen Informationen über die EmotionML-Bibliothek, wie implementierter Standard oder Version, durch die Klasse *EmotionML* zur Verfügung gestellt. Ein Parser für EmotionML wird in einer separaten Klasse *Parser* hinzugefügt. Diesem werden Dokumente oder Dokumentfragmente in EmotionML übergeben und entsprechend Objektinstanzen für die Inhalte instanziiert.

Die Erstellung des resultierenden XML wird zum Zwecke der flexibleren Handhabung in den Klassen der jeweiligen EmotionML-Elemente implementiert. Hierfür wird jede dieser Klassen mit den Methoden *ToDom()*, welche eine *XmlDocument*-Instanz zurückliefert, und *ToXml()* für die Generierung einer die XML-Zeichenkette, versehen. Somit ist es möglich, bei einem fertig eingerichteten EmotionML-Document die benötigten Teilbereiche der resultierenden XML von den jeweiligen beinhalteten Unterobjekten erstellen zu lassen und diese dann zur kompletten XML zusammenzusetzen.

Objekte können miteinander auf Gleichheit verglichen werden. Üblicherweise wird hierfür im C# die Methode *Equals(object)* verwendet. Um einen Vergleich der einzelnen Bestandteile innerhalb der EmotionML-Bibliothek zu ermöglichen, wird die C#-interne Methode überschrieben und entsprechend erweitert. Eine Besonderheit, die entgegen des gewohnten Verhaltens dieser Methode in der Programmbibliothek hinzukommt, ist die Unterstützung eines zweiten Parameters, welcher ein Array von Zeichenketten darstellt. Mit diesem ist teilweise eine Angabe möglich, welche Bestandteile des Elements nicht in die Gleichheitsbetrachtung einbezogen werden sollen. Somit wird es bspw. möglich zwei Emotionsinstanzen miteinander zu vergleichen ohne dabei den <info>-Bereich zu berücksichtigen. Die unterstützten Schlüsselworte und Möglichkeiten können der programminternen Dokumentation entnommen werden.

[40] Eine Aufrufkaskade in Abarbeitung befindlicher Methoden, die oft im Falle eines Programmabsturzes ausgegeben wird und somit die Diagnose des Fehlers vereinfacht.

In der Programmbibliothek ist ein Parser für EmotionML enthalten. Dieser kann übergebene EmotionML-Dokumente in entsprechende Objektinstanzen umwandeln, die dann im weiteren Programmverlauf normal genutzt werden können. Weiterhin ist hiermit eine Kontrolle der übergebenen EmotionML-Dokumente möglich. Die Überprüfung der Dokumente erfolgt einerseits mittels integrierter Programmlogik, wird aber auch durch die Validierung anhand der im Standard angegebenen XML-Schemas [4] ergänzt. Die hierfür benötigten XML-Schemas sind zusammen mit den Standardvokabularien für die Emotionsbestandteile als interne Ressourcen über die *Helper*-Klasse der Programmbibliothek zugreifbar. In diese *Helper*-Klasse sind zusätzlich spezielle Komponenten der Validierung diverser Inhalte ausgelagert.

Wird die EmotionML-Bibliothek anhand der Testfälle des Implementation Report [95] bewertet, so fällt auf, dass alle Testfälle bis auf die Unterbedingungen (sub contraints) erfolgreich durchlaufen werden. Diese Unterbedingungen – zu finden in den Testfällen 114, 117, 120, 123, 161, 164, 167, 170, 212, 222, 232, 242, 412 und 417 – prüfen dabei das korrekte Zusammenspiel zwischen Vokabular und Emotionsrepräsentation sowie die Angabe korrekter MIME-Typen ab. Diese Bereiche werden für die Lösung der Problemstellung dieser Arbeit nicht benötigt, resultieren aber in hohen Aufwänden seitens der Implementierung. Deshalb wurde entschieden diese Bereiche nicht in dieser Version der EmotionML-Bibliothek zu implementieren. Eine vollständige Auflistung der Tests des Implementation Report sowie das Ergebnis seiner Durchführung im Vergleich zu anderen Implementierungen sind in Anhang D zu finden. Die XML-Repräsentation dieses Implementation Reports ist in Anhang E notiert.

5.2 XMPP-Bibliothek

Um Aufwände bei der Implementierung zu veringern und die Wartbarkeit zu vereinfachen, soll eine unterstützende Programmbibliothek für die Verwendung des XMPP eingesetzt werden. Dies hat zusätzlich noch den Vorteil, dass der Einsatz der Implementierung in Software, die diese oder eine ähnliche Grundlage ebenfalls nutzt, leichter verwendbar ist. Weiterhin geschieht es somit, dass die verwendete XMPP-Bibliothek unabhängig vom Projekt verbessert und erweitert werden kann, was wiederum indirekt der Anwendung zu Gute kommen kann. Bei der Wahl einer geeigneten Programmbibliothek soll vor allem ein Augenmerk auf die Offenheit und freie Verwendungsmöglichkeit der Software innerhalb von Open Source geachtet werden. Eine anhaltende Projektaktivität sorgt dafür, dass die Bibliothek gepflegt und möglicherweise sogar weiterentwickelt wird. Eine gute Dokumentation der Software zeugt weiterhin von einer professionellen Softwareentwicklung und erleichtert die Einarbeitung und Verwendung während der Implementierung der Anwendung. Um die Verbreitung von XMPP zu unterstützen und den Einstieg in die Entwicklung mit dieser Technologie zu vereinfachen, pflegt die XMPP Standards Foundation auf ihrer Webseite eine Liste mit nutzbaren Programmbibliotheken verschiedener Programmiersprachen [97]. Darunter sind auch fünf Programmbibliotheken für die Programmiersprache C#:

- IP*Works Internet Toolkit
- agsXMPP SDK
- Matrix
- jabber-net
- SoapBox Studio

Die Bibliotheken *IP*Works Internet Toolkit*, *Matrix* und *SoapBox Studio* sind kommerzieller Natur und scheiden daher aus der weiteren Betrachtung im Rahmen von Open Source aus. Eine nähere Betrachtung des Projektes *jabber-net*, welches bei Google Code gehostet[41] ist, ist aufgrund dessen Lizenz GNU Lesser General Public License (GPL) empfehlenswert. Bei genauerer Analyse der Projektseite fällt jedoch auf, dass die letzte Version zum Download aus dem Jahre 2008 ist. Ein Blick in deren Versionskontrollsystem Subversion (SVN) legt weiterhin offen, dass die Entwicklung in den Jahren 2002 bis 2008 gut und stetig voranschritt, im Oktober 2008 jedoch ein plötzliches Ende nahm. Ein Hotfix im Jahr 2009 auf Versionsnummer 2.1.2 hat es nicht einmal in den Downloadbereich geschafft. Eine Abgeschlossenheit des Projektes kann ausgeschlossen werden, da in dessen Bugtracker[42] weiterhin Fehler eingestellt und alte Einträge noch nicht komplett abgearbeitet wurden.

Der letzte Vertreter ist das *agsXMPP SDK*. Da diese Software, die genau wie Matrix von AG-Software entwickelt wird, unter einer dualen Lizenz veröffentlicht ist, ist eine Nutzung in Open-Source-Projekten der GPL möglich. Durch die Fokussierung der AG-Software auf XMPP-Technologien, ist eine stetige Wartung und Weiterentwicklung des Software Development Kit (SDK) sehr wahrscheinlich. Dem Programmpaket liegt nach dem Download eine gute Dokumentation zusammen mit einigen auch komplexen Beispielimplementationen bei. Da hierdurch alle grundlegenden Kriterien an eine XMPP-Bibliothek erfüllt werden, wird das agsXMPP SDK bei der Implementierung der Anwendung genutzt.

Um mit dem agsXMPP SDK die übertragene XMPP-Nachricht so zu modifizieren, dass in dieser EmotionML unter Zuhilfenahme der in Kapitel 5.1 vorgestellten EmotionML-Bibliothek eingebettet werden kann, muss ein neuer Knotentyp definiert werden. Da das SDK nicht auf die XML-Typen von C# vertraut, sondern eigene implementiert, muss dieser Knotentyp als Klasse definiert werden, die von der Klasse *agsXMPP.Xml.Dom.Node* erbt. Aufgrund der Komplexität des EmotionML ist es nur möglich einen neuen Knotentyp durch eine Ableitung der Klasse *agsXMPP.Xml.Dom.Document* zu generieren. Diese Klasse bietet eine Methode *LoadXml()*, mit deren Hilfe vorhandenes XML in die Instanz geladen werden kann. Somit lässt sich die Ausgabe der Methode *ToXml()* der Emotionsinstanz einfach als Eingabe für den Knotentyp – am besten in dessen Konstruktor – direkt bei dessen Initialisierung verwenden, wie das Beispiel in Quelltext 12 zeigt.

```
class Emotiondoc : Document
{
      public Emotiondoc(EmotionmlLib.Emotion emotion)
      {
            this.Namespace = EmotionML.NAMESPACE;
            this.LoadXml(emotion.ToXml());
      }
}
```

Quelltext 12: Transformation von EmotionML in XMPP-Knotentyp

Im Anschluss muss der instanziierte Knoten des neu geschaffenen Typs mit Hilfe der Methode *AddChild()* der Klasse *agsXMPP.protocol.client.Message* der XMPP-Nachricht hinzugefügt werden. Ein Beispiel hierfür stellt Quelltext 13 dar, bei dem der instanziierte Knoten in der Variablen *emotiondoc* gespeichert ist.

[41] Weitere Informationen auf der Projektseite unter http://code.google.com/p/jabber-net/ (Abruf 31.08.2012)
[42] Softwaresystem zur Verwaltung von Programmfehlern und Aufgaben

```
Message msg = new Message();
msg.To = new Jid("user@example.org");
msg.AddChild(emotiondoc);
msg.Body = "Nachricht mit Emotion.";
```

Quelltext 13: Einbettung des XMPP-Knotentyps in eine XMPP-Nachricht

5.3 Implementierung der Anwendung

Zur besseren Übersicht und Wartbarkeit der Anwendung wird diese im mehrere Teile untergliedert. Eine daraus resultierende Dateistruktur ist in Abbildung 23 dargestellt.

Aufgrund der Trennung in eine emotionsverarbeitende Plattform und eine sie nutzende Anwendung in Form des Chatclients ist diese Trennung auch in der Dateistruktur ersichtlich. Die Quellen, die der Implementierung des Chatclients dienen, befinden sich im Ordner *application*. Weiterhin existiert ein Verzeichnis *extensions*, welches in die kompilierte Fassung der Anwendung übernommen wird. In ihm befinden sich die Extensions. Im Verzeichnis *lib* sind die drei verwendeten Programmbibliotheken (agsXMPP SDK, dotNetRDF und die EmotionML-Bibliothek) abgelegt. Weitere Programmbibliotheken können hinzugefügt werden. Für die emotionsverarbeitende Plattform sind Quellen im Verzeichnis *platform* verfügbar. Sämtliche Elemente, die direkt die Plugins betreffen oder gemeinsam mit diesen genutzt werden, befinden sich im Verzeichnis *plugin*. Dies sind, neben den in Kapitel 4.4 vorgestellten Schnittstellen und abstrakten Klassen für die Plugins, auch das im gleichen Kapitel beschriebene *Urlplugin* sowie eine *PluginException*, die eine bessere Behandlung aller in Zusammenhang mit den Plugins stehenden Fehler ermöglicht. Im Verzeichnis *users*, welches mit in die kompilierte Anwendung übernommen wird, werden zu deren Laufzeit alle nutzerspezifischen Konfigurationen notiert. Des Weiteren beherbergt dieses Verzeichnis einen Beispielnutzer mitsamt Konfigurationsdatei für die Gewichtung. Dabei wurde für die Beispielplugins bereits eine entsprechende Konfiguration vorgenommen. Um Namenskonflikte dieses Beispielnutzers mit einem realen Benutzer zu verhindern, wird der Name user@example.org gemäß RFC 2606 [98] gewählt.

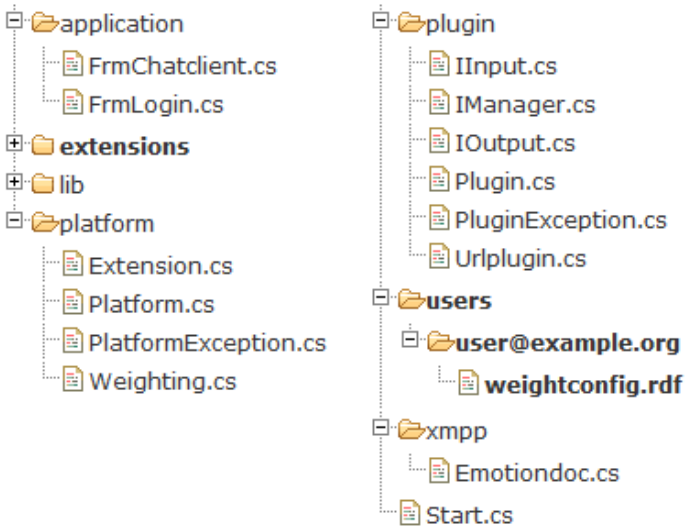


Abbildung 23: Dateistruktur des C#-Projekts vom Prototyp

Die **Plattform** bildet die Basis für eine oder mehrere Anwendungen. Sie übernimmt die komplette Verwaltung der Extensions sowie der in ihnen enthaltenen Plugins an zentraler Stelle. Des Weiteren realisiert sie jegliche Interaktion mit den Plugins und übernimmt die in Kapitel 4.5 beschriebene Gewichtung der über die Input-Plugins eingehenden Emotionen.

Dabei ist zu beachten, dass die Emotionen denselben Sets (category, dimension, appraisal, action-tendency) zugeordnet sein müssen, da sie sich sonst nicht nach einer Gewichtung in EmotionML darstellen lassen (vgl. Kapitel 3.2.2.2). Das jeweilige Set mit den meisten Eintragen wird genutzt. Zusätzlich verwaltet die Plattform diverse Daten des Nutzers, die für eine Emotionserkennung von Relevanz sein können oder allgemein für einen Transfer zwischen Anwendung und/oder Plugins genutzt werden. Auf Grund dieser Tatsache ist auch eine grundlegende Benutzerverwaltung notwendig. Um hier jedoch den Anwendungen größtmögliche Freiheiten zu ermöglichen, soll diese nicht sitzungsbasiert sein, d.h. es ist nicht notwendig, dass sich ein Nutzer gegenüber der Plattform registriert und anmeldet. Dies hat den Nachteil, dass eine geeignete Prüfung und Synchronisierung der Identität des Nutzers auf Ebene der Anwendungen stattfinden muss. Sollte dies nicht der Fall sein, so ist es möglich, dass ein Anwender auf die über einen anderen tatsächlichen Anwender unter gleichem Nutzernamen abgelegten Informationen zugreifen kann. Um dies zu umgehen, wird empfohlen den Nutzer anhand dessen E-Mail-Adresse oder einer anderen dem Nutzer eindeutig zuordenbaren Information im System zu führen. Bei einer Weiterentwicklung des Prototyps kann dieses Verfahren durch eine verschlüsselte Speicherung der Benutzerinformationen dahingehend erweitert werden, dass zu einem Benutzernamen mehrere entsprechend verschlüsselte Bereiche existieren. Mit Hilfe des durch den Nutzer übergebenen Schlüssels lässt sich jedoch nur der wirklich für ihn bestimmte Bereich wieder korrekt entschlüsseln. Allerdings wird angeraten, dieses Verfahren aufgrund seiner schlechteren Wartungsmöglichkeiten gegenüber dem zuerst vorgestellten Verfahren soweit möglich zu vermeiden.

Da die Plattform selbst keine Oberfläche aufweist und die Plugins dies auch nicht müssen, ist für die Behandlung von Fehlern ein flexibler Logging-Mechanismus nötig. Hierfür ist die Methode *log* der Klasse *Platform* zuständig. Diese notiert die Textrepräsentation übergebener Objekte mitsamt der Uhrzeit in der Datei *platform.log*.

Bei der Instanziierung der Plattform über das Singleton-Entwurfsmuster werden alle Extensions mit ihren Plugins anhand der in der *install.rdf* notierten Konfiguration geladen und selbst instanziiert. Im nächsten Schritt wird die jeweils betreffende Konfiguration im Attribut *config* der entsprechenden Klasse notiert. Trotzdem somit die Metadaten wie Autor oder Beschreibung als RDF für die Extension bzw. das Plugin vorliegen, werden sie nochmals in Schlüssel-Wert-Paaren im Attribut *properties* gespeichert. Dies erfolgt mit dem Ziel, Erweiterungen die Verwaltung des RDF bzw. des Formates der betreffenden Konfiguration (sollte dies zukünftig geändert werden) abzunehmen. Aufgrund der sofortigen Initialisierung ist seitens der Plattform gewährleistet, dass eine Anwendung schnellstmöglich Emotionen abfragen kann. Dies erfolgt analog zu der Pluginschnittstelle über die Methoden *detectEmotion()* und *outputEmotion()*. Als optionalen Parameter kann diesen der Benutzername übergeben werden, damit eine nutzerspezifische Gewichtung erfolgen kann.

Der gemäß der Definition aus Kapitel 4.3 umgesetzte **Chatclient** besteht aus zwei Windows Forms, nämlich

- *FrmChatclient* für die Funktionalität des eigentlichen Chats und der Verwaltung der Kontakte des Nutzers und
- *FrmLogin* für den Aufbau der mit Hilfe des Nutzers authentifizierten Verbindung zum XMPP-Server.

Mit dem Start der Anwendung erfolgt automatisch die Initialisierung von *FrmChatclient* in der Main-Methode[43]. Dabei wird unter Zuhilfenahme des agsXMPP ein Objekt für die XMPP-Verbindung initiiert und grundlegend konfiguriert. Wird festgestellt, dass sicher der Nutzer nicht gegenüber dem XMPP-Server authentifiziert hat, erscheint *FrmLogin*, so dass der Nutzer eine Anmeldung vollziehen kann. Ist dies geschehen, werden die Kontakte des Nutzers geladen, damit dieser mit dem Chat beginnen kann. Dafür muss der Nutzer einen Gesprächspartner in seiner Kontaktliste auswählen. Erfolgt dies nicht, so ist der Versand der Nachricht deaktiviert. Konnte die Nachricht erfolgreich an den Empfänger abgeschickt werden, so wird sie in das Nachrichtenfenster übernommen. Gleiches erfolgt beim Eingang einer neuen Nachricht.

AgsXMPP ist eventbasiert aufgebaut. Daher ist es möglich, dass sowohl *FrmChatclient* als auch *FrmLogin* jeweils eigene Funktionalitäten an das Event einer erfolgreichen Anmeldung binden können. Während *FrmChatclient* die Kontaktliste lädt, gibt *FrmLogin* den Chat für den Nutzer frei und schließt sich.

5.4 Beispiel-Plugins

Die Chatplattform ist ohne jegliches Plugin nur eine Plattform für den reinen Austausch von Nachrichten ohne jedwede Emotionalität. Da diese erst durch die in den Extensions enthaltenden Plugins der Anwendung zugeführt werden (vgl. Kapitel 4.4.1), sind zwei Beispielextensions implementiert, die ansatzweise die Vorteile und Möglichkeiten der Anwendungsplattform verdeutlichen und als Beispiele für die Entwicklung weiterer Plugins dienen sollen. Zum einen ist dies eine einfache Extension, welche ein Input-Plugin beherbergt, das mit einer statischen EmotionML-Datei interagiert. Weiterhin ist hier in der Konfiguration ein Output-Plugin definiert, welches die Benutzernachricht und deren Emotion an eine URL sendet. Zum zweiten wird ein komplexeres Plugin implementiert, welches ein Input-Plugin beinhaltet, das mit dem Benutzertext arbeitet und anhand einer Konfiguration eine Extraktion von Emoticons aus dem vom Nutzer eingegebenen Text bewerkstelligt. Diesem wird dann eine in EmotionML kodierte Emotion zuwiesen und an die Anwendungsplattform übermittelt. Ein Output-Plugin stellt hier die Ausgabe des Emoticons beim Empfänger dar. Um die interkulturellen Vorteile der Übermittlung standardisierter Emotionen aufzuzeigen, wird dieses ausgegebene Emoticon quasi einer Lokalisierung unterzogen. Somit ist es mit diesem Plugin möglich eine Emotion als Smiley zu senden und als Emoji wieder zu empfangen (vgl. Kapitel 3.2.1.1).

5.4.1 Statische Emotion

Da Plugins nicht zwingend mit der eigentlichen Anwendung ausgeliefert werden müssen und doch relativ separat für sich stehen, wird für dieses Plugin in der C#-Solution ein Unterprojekt mit dem Namen *Plainemotion* angelegt. Dieser Name ist gleichzeitig der interne Schlüssel für die Extension und grenzt deren Namensraum innerhalb der Anwendung ein. In der Konfigurationsdatei werden, neben allgemeinen Beschreibungen zur Extension, zwei Plugins definiert. Das Plugin *plainemotionurl* stellt dabei ein Output-Plugin dar, welches die empfangene Emotion durch das interne URL-Plugins an eine externe Adresse weitergibt. Ein Testscript, verfasst in der Scriptsprache PHP, welches hier zu Testzwecken als Endpunkt bereitgestellt werden kann, ist im

[43] Erste Methode, die beim Start eines Programmes aufgerufen wird.

Anhang B aufgeführt. Es speichert Nutzer, Nachricht und EmotionML von Aufrufen via POST-Methode in einer Datei und gibt sie bei einem Aufruf des Scriptes via GET wieder aus. Die im Quelltext angegebene Datei muss daher mit Schreibrechten konfiguriert werden. Neben diesem Output-Plugin beinhaltet die Extension auch ein Input-Plugin namens *plainemotionfile*, welches eine im Verzeichnis der Extension befindliche EmotionML-Datei ausliest und diese an die Anwendung weiterleitet.

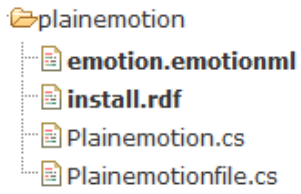


Abbildung 24: Dateien in Extension Plainemotion

Wie Abbildung 24 zeigt, besteht die Extension aus vier Dateien. Die Basis bildet wie bei jeder Extension die Datei *install.rdf* in der die Konfiguration notiert ist und mit der die Extension in der Plattform registriert wird. Die für dieses Plugin notwendige statische EmotionML-Datei ist im EmotionML-Dokument *emotion.emotionml* abgelegt. Weiterhin gehören zur Extension zwei Klassen. Die Klasse *Plainemotion* stellt dabei den Manager der Extension dar. Dieser liefert der Anwendung eine Instanz der Input-Plugin-Klasse *Plainemotionfile* zurück. Die in Abbildung 24 fett gedruckten Dateinamen werden in die kompilierte Fassung der Extension übernommen und durch eine *plainemotion.dll* ergänzt.

5.4.2 Emoticonverarbeitung

Eine kompliziertere und aufwändigere Extension stellt die Extension *Emoticonhandler* dar. Sie besteht aus den Input-Plugin *emoticon2emotion* und dem Output-Plugin *emotion2emoticon*. Das Input-Plugin interagiert mit dem Chattext des Nutzers und extrahiert die Emoticons aus dem Text, um diese dann in eine EmotionML-Repräsentation zu überführen. Das Output-Plugin hingegen nimmt eben diese in EmotionML notierte Emotion entgegen und weist ihr ein Emoticon zu, das dem Chattext des Nutzers beigefügt wird. Sollte im EmotionML ein mit der Smiley Ontology abgebildetes textbasiertes Emoticon übertragen werden, so wird dies angezeigt. Abbildung 25 zeigt die Dateien der Extension. Fett gedruckte Dateien werden dabei in die kompilierte Fassung übernommen, der Rest zu einer emoticonhandler.dll kombiniert. Weiterhin ist ersichtlich, dass im Unterverzeichnis *emoticons* zwei EmotionML-Dateien existieren.

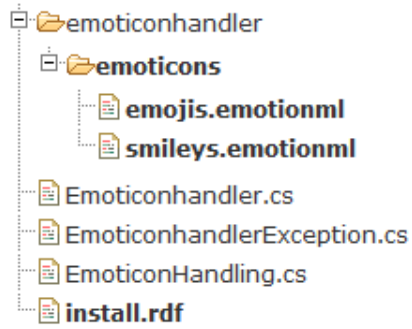


Abbildung 25: Dateien in Extension Emoticonhandler

Diese Dateien dienen der Zuordnung von EmotionML-Emotionen zu den Emoticons (Smiley Ontology [53]) und RDF-Emotionen (Emotion Ontology [55]). Dabei wird zwischen Emojis und Smileys in den Dateien unterschieden. In diesen Konfigurationsdateien sind, wie die

Dateiendung schon verrät, normale EmotionML-Dokumente eingebettet. Sie können also mit der in Kapitel 5.1 definierten EmotionML-Bibliothek verarbeitet werden. In den Dokumenten ist eine Reihe von Emotionen notiert. Jede Emotion ist dabei einer Anzahl Emoticons zugeordnet. Diese Zuordnung erfolgt über eine Einbettung eines RDF-Dokuments in das <info>-Element der in EmotionML dargestellten Emotion. In diesem RDF sind dann Emoticons mit Hilfe der Smiley Ontology definiert. Um die semantische Angabe der Emotion zu komplettieren, wird durch das Prädikat *representsEmotion* der Smiley Ontology ein Emotionswert aus der Emotion Ontology zugeordnet. Aufgrund der in Kapitel 3.2.1 festgestellten Eigenschaft, dass der Ausdruck von Emoticons verstärkt wird durch die Wiederholung der Zeichen, die den Mündern entsprechen, oder aber durch das Einfügen von Zeichen im Netzjargon, wird das Element *<emochat:emoticonregex>* definiert. Diesem kann ein Regulärer Ausdruck zugewiesen werden, der mehreren möglichen Emoticons entspricht. Somit können auch diese Abwandlungen vom Input-Plugin erkannt und verarbeitet werden. Wird eine Emoticonkonfiguration in Schreibweise regulärer Ausdrücke getätigt, so empfiehlt es sich, zusätzlich ein Emoticon mit Hilfe der Smiley Ontology zu definieren, da sonst das Output-Plugin kein passendes Emoticon darstellen kann. Ein Beispiel für die Konfiguration eines Emoticons ist in Quelltext 14 gegeben.

```
<emotion category-set="http://www.w3.org/TR/emotion-voc/xml#big6">
  <info>
    <rdf:RDF><smiley:Emoticon>
    <smiley:representsEmotion
        rdf:resource="http://purl.obolibrary.org/obo/MFOEM_000035"/>
    <smiley:representedBy><smiley:CharacterRepresentation>
      <smiley:isAnimated>false</smiley:isAnimated>
      <!-- :-) :) -->
      <emochat:emoticonregex><![CDATA[/:\-?\)/]]></emochat:emoticonregex>
      <smiley:characters><![CDATA[:-)]]></smiley:characters>
    </smiley:CharacterRepresentation></smiley:representedBy>
    </smiley:Emoticon></rdf:RDF>
  </info>
  <category name="happyness" value="0.75"/>
</emotion>
```

Quelltext 14: Emoticonkonfiguration für Smileys in Extension Emoticonhandler

Neben der globalen Konfiguration von Emoticons existiert noch eine nutzerspezifische Konfiguration in Form einer *config.xml*. Mit dieser in Quelltext 15 exemplarisch dargestellten Konfigurationsdatei bekommt der Nutzer die Möglichkeit, zu bestimmen, ob er sein Emoticon und seine Emotion mittels RDF mit der Nachricht über das Input-Plugin versenden möchte (Pfad */config/include/emoticon* bzw. */config/include/emotion*). Eine weitere Option ist hier die Wahl des zu nutzenden Emoticon-Sets, also Smileys oder Emojis, bei der Ausgabe durch das Output-Plugin (Pfad */config/emoticon/set*). Bei einem weiteren Ausbau dieser Extension könnte hier eine automatische Vorauswahl gemäß einer Softwarelokalisierung stattfinden.

```
<?xml version="1.0" encoding="utf-8" ?>
<config>
  <emoticon>
    <set>smileys</set>
  </emoticon>
  <include>
    <emoticon>true</emoticon>
    <emotion>true</emotion>
  </include>
</config>
```

Quelltext 15: Nutzerkonfiguration im Plugin Emoticonhandler

Wie Abbildung 25 zeigt, wurde für die Extension ein eigener Exception-Typ mit der Klasse *EmoticonhandlerException* definiert. Dies erfolgt, da hierdurch eine bessere Fehlerbehandlung möglich ist. Die Aufgaben des Extension-Managers werden von der Klasse *Emoticonhandler* übernommen. In dieser Extension ist die Besonderheit gegeben, dass die Klasse *EmoticonHandling*, aufgrund der großen Ähnlichkeit der Plugins, sowohl die Aufgaben als Input-Plugin wie auch als Output-Plugin übernimmt. Durch eine Ableitung der Klasse *Plugin* sowie der Implementierung von *IInput* und *IOutput* gemäß Kapitel 4.4.3 ist dies problemlos möglich. Wird eine Emotion mittels XMPP empfangen, so sucht der Bestandteil des Output-Plugins gemäß Nutzerkonfiguration ein passendes Emoticon in den EmotionML-Dateien der Extension. Wird dieses gefunden, so wird es dem Chattext hinzugefügt, bevor dieser dem Nutzer angezeigt wird. Von der Extension wird daher nur ein Emoticon pro XMPP-Nachricht unterstützt. Der Bestandteil des Input-Plugins durchsucht die Chatnachricht des Absenders vor dem Versenden nach Emoticons, die durch die Konfigurationsdateien bestimmt sind. Die zugehörigen EmotionML-Bereiche werden aggregiert und das Emoticon aus dem Chattext vor dem Versand an den Empfänger entfernt. Wählt der Nutzer die Konfiguration, dass sowohl sein verwendetes Emoticon als auch die daraus resultierende Emotion mit der Nachricht versendet werden sollen, damit diese in völliger Klarheit der gemeinten Aussage beim Empfänger ankommen, so tritt die in Quelltext 16 dargestellte Situation auf, dass in der resultierenden XMPP-Nachricht XMPP (grün), EmotionML (blau), Smiley Ontology (orange) und Emotion Ontology (gelb) ineinander integriert sind.

```
<message from='romeo@example.net/pda'
         to='juliet@example.org'
         type='chat'>
  <emotion category-set="http://www.w3.org/TR/emotion-voc/xml#big6">
    <info>
      <rdf:RDF><smiley:Emoticon>
        <smiley:representsEmotion
          rdf:resource="http://purl.obolibrary.org/obo/MFOEM_000042"/>
        <smiley:representedBy><smiley:CharacterRepresentation>
          <smiley:isAnimated
            rdf:datatype="&xsd;boolean">false</smiley:isAnimated>
          <smiley:characters><![CDATA[:-D]]></smiley:characters>
        </smiley:CharacterRepresentation></smiley:representedBy>
      </smiley:Emoticon></rdf:RDF>
    </info>
    <category name="happyness" value="0.85"/>
  </emotion>
  <body>Mir geht es gut.</body>
</message>
```

Quelltext 16: Kombination XMPP, EmotionML, Smiley Ontology, Emotion Ontology

5.5 Wiederverwendbarkeit der Implementierung

In ihrer Implementierung ist die Anwendung mit ihren Komponenten auf einen Chatclient als Prototypen ausgerichtet. Aufgrund des modularen Aufbaus der Anwendung ist der Einsatz der Teilkomponenten jedoch nicht auf dieses Szenario beschränkt. Durch die Nutzung der Aggregation und Gewichtung eingehender Emotionen, kann die Anwendung als Aggregator für Emotionen verwendet werden, da der Versand eines Chattextes optional ist. Auch wenn die Extensions mit ihren Plugins aufgrund der festgelegten Schnittstelle mit der Anwendung nicht direkt in anderen Projekten genutzt werden können, so ist zumindest eine Verwendung ihrer Algorithmen und Konfigurationen möglich. Eine Rekonfiguration der Extensions durch eine Nutzung von URL-Plugins kann bewirken, dass Emotionen beliebiger Onlinequellen

aggregiert und zusammengefasst an einen Onlinedienst weitergegeben werden. Eine Anbindung weiterer Systeme ist somit sehr leicht möglich. Schon die Möglichkeit des Versendens von Emotion, Text und Nutzerkennung an ein im Internet befindliches PHP-Script zeigt auf, dass beliebige Webanwendungen wie bspw. soziale Netzwerke einfach angebunden werden können. Im Bereich der Extension Emoticonhandling ist es speziell die Kombination aus EmotionML, Smiley Ontology und Emotion Ontology, die eine semantisch korrekte Auszeichnung von EmotionML-Inhalten zulässt. Dieses könnte ein Bootstrapping für die RDF-Notationen der Emotionsvokabularien sein, bis hier seitens EmotionML konkrete Standardisierungen der Emotionsvokabularien mit RDF definiert werden.

Ein direkter Einsatz in anderen Projekten ohne jedwede Anpassungen ist durch die EmotionML-Konfigurationsdateien der Extension Emoticonhandling gegeben. Hierdurch lässt sich eine Zuweisung von Emoticon zu Emotion entnehmen. Weiterhin ist über die Zuordnung gleicher Emotionen eine konkrete Zuordnung korrespondierender Paare zwischen Smileys und Emojis möglich. Durch eine weitergehende Konfiguration können bei leichtem Verlust der Semantik auch verschiedene Lokalisierungen des Netzjargons einander zugeordnet werden. Die EmotionML-Bibliothek lässt sich durch ihre Unabhängigkeit problemlos unverändert in weiteren Projekten einsetzen. Nicht nur das Parsen und die Überprüfung von vorhandenem EmotionML ist möglich, sondern durch verschiedene Objektinstanzen auch eine begrenzte Arbeit mit Emotionen in der Programmierung an sich. Zudem kann die Klasse, die mit Hilfe des Frameworks agsXMPP eine XMPP-Nachricht um EmotionML erweitert, entnommen und in anderen Projekten, die auch agsXMPP und die EmotionML-Bibliothek einsetzen, unverändert eingesetzt werden. Soll in diesem Szenario eine andere EmotionML-Bibliothek eingesetzt werden, so sind die notwendigen Änderungen schnell umgesetzt.

5.6 Zusammenfassung Implementierung

Im Rahmen der Implementierung des Lösungskonzeptes in Form eines exemplarischen Chatclients für das XMPP, erfolgt zum Zweck besserer Wartung und Wiederverwendbarkeit eine Trennung zwischen Plattform und Anwendung. Die Plattform kann dabei als Basis für verschiedene Anwendungen genutzt werden. Sie verwaltet die Extensions sowie die darin befindlichen Plugins und realisiert die Interaktion zwischen Anwendung und Plugins. Es erfolgen hier zusätzlich die Bündelung aggregierter Emotionen anhand einer nutzerspezifischen Gewichtung sowie die damit zusammenhängende Benutzerverwaltung. Weiterhin werden zwei Beispielplugins erstellt, die eine Nutzung der Plattform und des Chatclients im Rahmen der betrachteten Problemstellung ermöglichen.

Um die Implementierung des Chatclients zu erleichtern, erfolgt der Einsatz verschiedener Programmbibliotheken nach deren Analyse. Da keine nutzbare Programmbibliothek für EmotionML gefunden wurde, erfolgt eine eigene Definition und Implementierung. Somit sind alle Voraussetzungen geschaffen, den Chatclient als auf die Plattform aufbauende Anwendung zu implementieren. Bei der Nutzung der Extension *Emoticonhandler* tritt dabei die Situation auf, dass in der resultierenden XMPP-Nachricht XMPP, EmotionML, Smiley Ontology und Emotion Ontology ineinander integriert sind. Im Anschluss an die Implementierung daran erfolgt eine Betrachtung zur möglichen Weiterverwendung ihrer Teile für weitere Projekte.

6 Evaluation

Bei der Evaluation der Gesamtlösung in Kombination mit dem implementierten Prototyp wird gleich vorgegangen wie bei der Evaluierung der in Kapitel 3.2 aufgestellten Szenarien anhand der Bewertungskriterien, die in Kapitel 3.3 erfolgte. Die Ergebnisse der Evaluation der Gesamtlösung können Tabelle 12 entnommen werden.

Im ersten Bereich der Bewertungskriterien, der zusammenfassend mit „Zusammenspiel mit externen Systemen“ beschrieben ist, wird die Nutzbarkeit für Instant Messaging als sehr wichtig eingestuft. Aufgrund der Erweiterung des XMPP und die Umsetzung des Prototyps als Chatclient wird diesem sehr gut entsprochen. Die Möglichkeit der Anbindung sozialer Netzwerke ist hingegen nur als gut zu bewerten. Zwar lassen sich mit Hilfe des integrierten URL-Plugins Daten mit beliebigen anderen Webprojekten austauschen, wozu soziale Netzwerke mit einzuordnen sind, jedoch ist bisher seitens sozialer Netzwerke noch keine kompatible Schnittstelle für die Zuführung von Emotionen bekannt. Auch bei der Unterstützung der Emotionserfassung müssen Abstriche geltend gemacht werden, so dass dieser Bereich nur mit gut zu bewerten ist. Zwar kann die Anwendung Daten des Nutzers ,wie seinen eingegebenen Chattext, an die Plugins über eine leicht erweiterbare Schnittstelle weiterleiten, jedoch sind keine generischen Einbindungen verschiedener Sensoren möglich. An dieser Stelle existiert noch Potenzial. So ist es denkbar, dass für angeschlossene Sensoren eine Art Beschreibung vorliegt, die das Wesen des Sensors beschreibt (Kamera, Mikrofon, …). Dieses Anwendungsszenario geht dabei über den Rahmen eines Prototyps hinaus. Eine sehr gute Umsetzung kann jedoch bei der Verwendung offener Protokolle notiert werden. Die im Kern verwendeten Protokolle XMPP und EmotionML sind offen standardisiert und entsprechend dokumentiert. Auch die für die Extension Emoticonhandler genutzten Ontologien für Emoticons und Emotionen sind frei verfügbar. Eine sehr gute Erweiterbarkeit ist auch seitens der Inputvektoren möglich. Durch die flexible Pluginstruktur mit der darauf folgenden Zusammenfassung und Gewichtung von Emotionen ist eine hohe Flexibilität gegeben.

Im Bereich „Implementierung“ sind alle Bewertungskriterien mit sehr gut zu bewerten. Mit Hilfe verschiedener Plugins können beliebig viele unterschiedlichste Quellen für den Emotionsinput herangezogen werden. Dabei ist es auch möglich, dass ein Plugin selbst eine Aggregation von Emotionen aus mehreren Quellen vornimmt. Die Trennung von Emotion und Inhalt ist eines der Ziele dieser Lösung. Somit erfolgt unter Zuhilfenahme von EmotionML und den Objektinstanzen für Emotionen durch die EmotionML-Bibliothek eine Trennung, wann immer diese möglich ist. Zwar erweitert das Beispielplugin Emoticonhandler den empfangenen Nutzertext um ein Emoticon, um mit diesem die empfangene Emotion darzustellen, jedoch ist dies nur ein Mittel zum Zweck einer einfachen und allseits bekannten Darstellung einer Emotionsrepräsentation im Chat-Kontext. Eine Nutzung von Standards erfolgt in der Anwendung überall dort, wo sie machbar und angebracht ist. Dabei werden vor allem XML und daraus abgeleitete Auszeichnungen benutzt. Alle Komponenten der Lösung sowie die benutzten Bibliotheken sind unter einer Open-Source-Lizenz veröffentlicht. Die verwendeten externen Bibliotheken sind auf entsprechenden Projektseiten oder durch Beilagen gut dokumentiert. Ergänzt wird dies durch die Dokumentation der Schnittstellen der Lösung in dieser Arbeit sowie in der Dokumentation der Quelltexte. Mit Hilfe der Lösung ist eine Emotion leicht abzubilden. Wird die Extension Emoticonhandler zu Hilfe genommen, so lässt sich eine Emotion auch gleichzeitig mit mehreren kombinierten Darstellungen auszeichnen (vgl. Quelltext 16).

Zuletzt wird der Bereich „Emotionen“ betrachtet. Hierbei muss eine Emotion eindeutig abgebildet werden können. Dies ist durch die Beschreibung einer Emotion mit Hilfe der EmotionML sehr gut gegeben. Weiterhin lässt sich mit Hilfe der Extension Emoticonhandler die Emotion auch semantisch mittels RDF auszeichnen. Die Umwandlung der Emotion in die Deutungsweise einer anderen Kultur kann als gut angesehen werden. Hierfür ist die Extension Emoticonhandler nötig. Aufgrund der Komplexität dieses Teilgebietes kann deren Umwandlungsmöglichkeit in die Emoticons eines anderen Kulturraumes allerdings nur die Realisierung eines ersten Ansatzes sein. Durch die konsequente Abbildung einer Emotion in EmotionML ist eine sehr gute Serialisierung dieser gegeben. Auch eine Übertragung der Emotion ist sehr gut mit Hilfe der in Kapitel 4.1.2 entwickelten Erweiterung des XMPP möglich. Die Zusammenfassung mehrerer Emotionsrepräsentationen ist sehr gut durch die Anwendung der nutzerspezifischen Gewichtung vorhanden. Auch eine sehr gute Erweiterbarkeit des Vokabulars für die Emotionen ist gegeben, da alle Elemente, die Emotionen in irgendeiner Art auszeichnen, wie EmotionML oder Emotion Ontology, in diesem Punkt flexibel sind und die Anwendung diese Flexibilität beibehält.

Legende:		**Lösung mit Prototyp**
++	sehr gut	
+	gut	
o	neutral	
-	schlecht	
--	sehr schlecht	
Zusammenspiel mit externen Systemen		
für Instant Messaging nutzbar		++
Möglichkeit der Anbindung an soziale Netzwerke		+
Emotionserfassung unterstützen		+
Verwendung offener Protokolle		++
Erweiterbarkeit der Inputvektoren		++
Implementierung		
mehrere Quellen für Emotionsinput		++
Trennung von Emotion und Inhalt		++
Nutzung von Standards		++
Open Source/frei verfüg- und nutzbar		++
dokumentierte Schnittstellen		++
Abbildung einer Emotion		++
Emotionen		
Eindeutigkeit der abgebildeten Emotion		++
Emotion in Deutungsweise einer anderen Kultur umwandeln		+
Serialisierung von Emotionen		++
Übertragung von Emotionen		++
Zusammenfassung mehrerer Emotionsrepräsentationen		++
Vokabular für Emotionen erweiterbar		++
Gesamtgüte		72

Tabelle 12: Evaluierung der Lösung anhand der Bewertungskriterien

Nachdem die Evaluierung abgeschlossen ist, kann gemäß dem in Kapitel 3.1.2 festgelegten Bewertungsschema die Gesamtgüte festgestellt werden. Die Lösung weist hier mit 72 von möglichen 76 Punkten eine sehr hohe Gesamtgüte auf. Im Vergleich dazu schnitten die in Kapitel 3.3 evaluierten Szenarien deutlich schlechter ab. So waren die Ergebnisse von Emotionsmessungen mit 15 Punkten weit abgeschlagen hinter der Nutzung von Emoticons

und Netzjargon mit 27 Punkten. Selbst die Nutzung von Auszeichnungssprachen und Ontologien, als bester Vertreter jener Evaluation, kann nicht mit dem Ergebnis der Lösung mithalten. Weiterhin konnte die vorgestellte Lösung in allen Teilbereichen erfolgreich punkten, was bei den restlichen Szenarien nicht der Fall war.

7 Schlussfolgerung

Die vorliegende Arbeit hat ergeben, dass eine Serialisierung und Übertragung von eindeutigen Emotionen möglich ist. Wie der als Prototyp implementierte Chatclient zeigt, kann eine Nutzung im Rahmen des Instant Messagings oder sozialer Netzwerke vollzogen. Durch die Serialisierung der Emotion mit Hilfe standardisierter Auszeichnungssprachen wie EmotionML, kann die emotionale interkulturelle Kommunikation verbessert werden. Weiterhin ist eine vereinfachte automatisierte Verarbeitung der Emotionsrepräsentation möglich.

Eine Analyse verschiedener gängiger Szenarien zur Abbildung von Emotionen in der textuellen Kommunikation hat ergeben, dass sowohl eine reine Abbildung mit Emoticons, wie auch eine Abbildung mit Hilfe von Auszeichnungssprachen und/oder relevanten Ontologien allein keine Lösung darstellt. Gleich verhält es sich auch bei der Nutzung von Algorithmen zur automatisierten Emotionserkennung. Ein angepasstes Zusammenspiel verschiedener Auszeichnungssprachen und Ontologien mit der bekannten Repräsentation von Emotion durch Emoticons und Elementen des Netzjargons führt jedoch zum gewünschten Ziel.

Im Rahmen der Arbeit entsteht ein Prototyp, der eine flexible und damit gut erweiterbare Systemarchitektur aufweist. Über Pluginschnittstellen können einfach emotionsverarbeitende Systeme mit der Anwendung kombiniert werden. Dadurch sind sowohl die Zuführung von Emotionen in die Anwendung aus verschiedensten Quellen, als auch das Abgreifen von Emotionen zur weiteren, von der Anwendung unabhängigen Verarbeitung möglich. Diese Möglichkeiten bei der Eingabe von Emotionen machen es jedoch notwendig, dass eine nutzerspezifische Gewichtung eingehender Emotionen entwickelt wird und zur Implementierung kommt. Mit ihrer Hilfe lassen sich automatisch Emotionen aus den verschiedenen Quellen aggregieren und zu einer gemeinsamen Emotion zusammenfassen. Zwar ist der Prototyp als Chatclient implementiert, durch die Nutzung des XMPP sind jedoch mit nur kleinen Änderungen viele weitere Einsatzmöglichkeiten denkbar. Um dies zu ermöglichen, ist die Erweiterung des XMPP für eine Einbettung von EmotionML notwendig. Das Plugin Emoticonhandler ergänzt diese Erweiterung nochmals dahingehend, dass Daten in RDF in das EmotionML eingebettet werden. Somit ist eine semantisch klare Zuordnung zwischen EmotionML, Emoticon und Emotion möglich, obwohl die Arbeitsgruppe um EmotionML für die Definition der Emotionsvokabularien kein RDF benutzt hat.

Diese Kombination von XMPP mit EmotionML und Ontologien zur semantischen Auszeichnung von Emoticons und Emotionen lässt sich unverändert auch für weitere Projekte einsetzen. Gleich verhält es sich auch mit der zur Erstellung des Prototyps benötigten C#-Bibliothek für EmotionML, welche den bisherigen Standard fast vollständig implementiert (siehe Anhang D). Weitere Synergieeffekte für andere Projekte entstehen durch die intensive Beschäftigung mit den Standards selbst. So erfolgte an die Arbeitsgruppe um Smiley Ontology ein Korrekturvorschlag für einen Klassennamen in dieser Ontologie [99]. Weiterhin war es möglich der Arbeitsgruppe um EmotionML einige Vorschläge für Verbesserungen zu unterbreiten. Hierbei wurde der Vorschlag eingereicht, statt der Nutzung

von Timestamps bei den entsprechenden Attributen einer Emotion eher auf den Typ xsd:dateTime zu setzen und somit Annotationen von Emotionen mit Zeiten vor dem 01.01.1970 zuzulassen [100]. Dieser Vorschlag wurde zugunsten der besseren Unterstützung der Extensible MultiModal Annotation Markup Language (EMMA) abgelehnt [101]. Einem gleichzeitig eingereichten Verbesserungsantrag, der die klarere Nutzung von Millisekunden in den Beispielen für Zeitstempel vorschlägt, wurde allerdings statt gegeben [102]. Ebenfalls erreicht werden konnte die einheitliche Verwendung des Präfixes *emo* im EmotionML-Standardisierungsdokument [103]. Das Ergebnis einer weiterhin eingereichten Anfrage zu Gleitkommazahlen im <trace>-Element steht hingegen zum Zeitpunkt der Ausarbeitung noch aus [104]. Im Anschluss an die Finalisierung dieser Arbeit wird für die implementierte EmotionML-Bibliothek der in Anhang E befindliche Implementation Report eingereicht, um hier auch wieder Signale zurückzugeben, dass eine Implementierung des momentanen Standards gut möglich ist.

Aufgrund der Eigenschaft der Implementierung als Prototyp, weist die Anwendung noch verbesserungswürdige Elemente auf. Zum einen ist dies die Unterstützung mehrerer Emotionen pro Nachricht, deren Möglichkeit in der protokollseitigen Umsetzung in Kapitel 4.1.4 aufgezeigt wurde. Des Weiteren ist mit dem momentanen Verfahren der Abfrage der Emotionen von den Plugins eine Echtzeitunterstützung der Anwendung schwierig. Kapitel 6 zeigt weiterhin, dass die Anbindung an soziale Netzwerke, die Unterstützung der Emotionserfassung und die Umwandlung der Emotion in eine Deutungsweise einer anderen Kultur noch nicht vollumfänglich gelöst sind. Allerdings ist dies durch Ausbau der Plugins sowie weiteren Forschungen möglich. Diese werden auch notwendig, wenn nicht nur spezielle Emotionen durch die Erweiterung Emoticonhandler verarbeitet werden sollen, sondern Emotionsbereiche. Ein großes Problem hierbei ist der fließende Übergang von einer Emotion in eine andere sowie die Unbestimmtheit, inwieweit verschiedene Emotionen miteinander gleich gesetzt werden können. Dies ist auch der Grund, warum nur die Gewichtung eingehender ähnlicher Emotionen vorgenommen werden kann. Weitere Forschungsmöglichkeiten zur Verbesserung des Prototyps resultieren in einer besseren Emotionserkennung. Verschiedene alltäglich nutzbare Darstellungen für Emotionen können untersucht werden, damit die übertragene Emotion wieder mit möglichst wenig Verlust vom Empfänger aufgenommen werden kann. Des Weiteren wurde bei Gesprächen über die Lösung und das Chatverhalten verschiedener Personen festgestellt, dass eine gender- und altersspezifische Verwendung von Netzjargon und Emoticons gegeben sein könnte. Auch eine Erweiterung des Chatclients und des von ihm benutzen erweiterten XMPP ist möglich. So könnte die Anwendung die Emotionserkennung unterstützen, indem sie eine flexible Schnittstelle für verschiedenste Sensoren wie Kameras oder Mikrofonrn anbietet. Diese können zusätzlich mit semantischen Metadaten versorgt und ihre Daten so den Plugins auf eine flexible Art und Weise zur Verfügung gestellt werden. Eine weitere Möglichkeit ist die nutzerspezifische Rechteverwaltung und die bessere Abgrenzung der Plugins voneinander. So könnte ein Nutzer zum Schutz seiner Privatsphäre bestimmen, welches Plugin welche Daten auslesen bzw. abändern darf. Eine weitere Untersuchung wert wäre die Unterstützung von Plugins in anderen Programmiersprachen. Eine Verbesserung des erweiterten XMPP ist möglich durch die Definition eines Publish-Subscribe-Mechanismus für das in XMPP eingebettete EmotionML. Somit könnte eine Basisemotion des Nutzers angegeben werden, die dann folgend von den Einzelemotionen der Nachrichten überlagert wird.

Literaturverzeichnis

[1] Statista GmbH, „Internetnutzer nach Häufigkeit der Nutzung von Instant Messaging von 2007 bis 2011 (in Millionen),“ 10 2011. [Online]. http://de.statista.com/statistik/daten/studie/168909/umfrage/internet---haeufigkeit-der-nutzung-von-instant-messaging/. [Zugriff am 02.09.2012].

[2] Statista GmbH, „Anzahl der Nutzer (in Mio.) sozialer Netzwerke in ausgewählten Ländern im Jahr 2011 und Prognose für 2014,“ 02 2012. [Online]. http://de.statista.com/statistik/daten/studie/219669/umfrage/prognose-nutzer-sozialer-netzwerke-ausgewaehlte-laender/. [Zugriff am 02.09.2012].

[3] G. DUMBRAVĂ und A. KORONKA, „WRITING FOR BUSINESS PURPOSES: ELEMENTS OF EMAIL ETIQUETTE,“ *Annals of the University of Petroşani,* Bd. 6, pp. 61-64, 2006.

[4] World Wide Web Consortium, „Emotion Markup Language (EmotionML) 1.0 - W3C Candidate Recommendation,“ 10.05.2012. [Online]. http://www.w3.org/TR/emotionml/. [Zugriff am 22.07.2012].

[5] Bertelsmann Lexikon, Gütersloh: Verlagsgruppe Bertelsmann GmbH, 1990.

[6] G. Krämer und S. Quappe, Interkulturelle Kommunikation mit NLP, Uni-Edition, 2006.

[7] G. Maletzke, Interkulturelle Kommunikation, VS Verlag für Sozialwissenschaften, 1996.

[8] F. S. v. Thun, Miteinander reden 1, rororo, 2001.

[9] C. E. Shannon und W. Weaver, Mathematical Theory of Communication, University of Illinois Press, 1949.

[10] E. Broszinsky-Schwabe, „Interaktionsrituale,“ in s *Interkulturelle Kommunikation*, 2011, pp. 161-176.

[11] Brockhaus Enzyklopädie, Mannheim: F.A. Brockhaus AG, 2006.

[12] H. Schmidt-Atzert, „Lehrbuch der Emotionspsychologie,“ Stuttgart, Berlin, Köln, 1996, p. 18.

[13] E. Douglas-Cowie, J.-C. Martin, L. Devillers und C. Cox, „HUMAINE deliverable D5g - Mid Term Report on Database Exemplar Progress,“ 2006.

[14] J. Fontaine, K. Scherer, E. Roesch und P. Ellsworth, „The world of emotions is not two-dimensional,“ *Psychological Science: 18 (12),* pp. 1050-1057, 2007.

[15] A. Ortony, G. L. Clore und A. Collins, „The Cognitive Structure of Emotions,“ Cambridge University Press, 1990.

[16] N. Frijda, „The Emotions,“ Cambridge University Press, 1986.

[17] T. Berners-Lee, „The Semantic Web,“ *Scientific American,* 01.05.2001.

[18] P. Hitzler, M. Kroetzsch, S. Rudolph und Y. Sure, „Semantic Web,“ Springer, 2009.

[19] World Wide Web Consortium, „RDF Primer,“ 10.02.2004. [Online]. http://www.w3.org/TR/rdf-primer/. [Zugriff am 28.07.2012].

[20] T. Berners-Lee, R. Fielding und L. Masinter, „RFC3986: Uniform Resource Identifier (URI): Generic Syntax,“ 01 2005. [Online]. http://tools.ietf.org/html/rfc3986. [Zugriff am 21.08.2012].

[21] World Wide Web Consortium, „Extensible Markup Language (XML) 1.1 (Second

Edition),“ 29.09.2006. [Online]. http://www.w3.org/TR/xml11/. [Zugriff am 24.07.2012].

[22] DCMI Usage Board, „DCMI Metadata Terms,“ 14.06.2012. [Online]. http://dublincore.org/documents/dcmi-terms/. [Zugriff am 27.08.2012].

[23] D. Brickley und L. Miller, „FOAF Vocabulary Specification 0.98,“ 09.08.2010. [Online]. http://xmlns.com/foaf/spec/. [Zugriff am 27.08.2012].

[24] F. Radulovic und N. Milikic, „Smiley Ontology,“ 2009.

[25] World Wide Web Consortium, „W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes,“ 05.04.2012. [Online]. http://www.w3.org/TR/xmlschema11-2/. [Zugriff am 25.07.2012].

[26] J. B. Walther und K. P. D’Addario, „The Impacts of Emoticons on Message Interpretation in Computer-Mediated Communication,“ in s *Social Science Computer Review*, Bd. 19 no. 3, 2001, pp. 324-347.

[27] Despair, Inc., „DESPAIR, INC. SECURES OFFICIAL TRADEMARK REGISTRATION FOR ":-(",“ 02.01.2001. [Online]. http://despair.com/frownonthis.html. [Zugriff am 22.05.2012].

[28] Associated Newspapers Ltd., „OMG! ;-) is no longer free to use as Russian businessman trademarks 'wink' emoticon,“ 13.12.2008. [Online]. http://www.dailymail.co.uk/news/article-1094086/OMG---longer-free-use-Russian-businessman-trademarks-wink-emoticon.html. [Zugriff am 22.05.2012].

[29] M. Stankovic, „Beyond Social Semantic Web, Talk at DERI Galway,“ 26.03.2009. [Online]. http://www.slideshare.net/milstan/beyond-social-semantic-web. [Zugriff am 07.08.2012].

[30] J. Baird, „Original Bboard Thread in which :-) was proposed,“ 2002. [Online]. http://www.cs.cmu.edu/~sef/Orig-Smiley.htm. [Zugriff am 22.07.2012].

[31] M. Yuki, W. W. Maddux und T. Masuda, „Are the windows to the soul the same in the East and West? Cultural differences in using the eyes and mouth as cues to recognize emotions in Japan and the United States,“ 2005.

[32] J. 8. LEE, „Is That an Emoticon in 1862?,“ 19.01.2012. [Online]. http://cityroom.blogs.nytimes.com/2009/01/19/hfo-emoticon/. [Zugriff am 22.07.2012].

[33] J. Azuma und H. Maurer, „From Emoticon to Universal Symbolic Signs: Can Written Language Survive in Cyberspace?,“ *3rd International Microlearning Conference,* Bd. Micromedia and Corporate Learning, pp. 106-122, 2007.

[34] A. I. Google Inc., „Proposal for Encoding Emoji Symbols,“ 05.03.2009. [Online]. https://sites.google.com/site/unicodesymbols/Home/emoji-symbols/proposal-text. [Zugriff am 22.07.2012].

[35] D. K. Harigaya, „The History of Smiley Marks,“ 05.02.2001. [Online]. http://staff.aist.go.jp/k.harigaya/doc/kao_his.html. [Zugriff am 22.05.2012].

[36] T. Hiroe, „Japanese Smileys(Emoticons),“ 2006. [Online]. http://club.pep.ne.jp/~hiroette/en/facemarks/index.html. [Zugriff am 23.07.2012].

[37] IBM Corporation, „Code Page CPGID 00437,“ 1984. [Online]. ftp://ftp.software.ibm.com/software/globalization/gcoc/attachments/CP00437.pdf. [Zugriff am 23.07.2012].

[38] The Unicode Consortium, „The Unicode Standard 6.0 - Miscellaneous Symbols,“ [Online]. http://unicode.org/charts/PDF/U2600.pdf. [Zugriff am 22.07.2012].

[39] The Unicode Consortium, "The Unicode Standard 6.0 - Emoticons," [Online]. http://unicode.org/charts/PDF/U1F600.pdf. [Accessed 22.07.2012].

[40] K. Leidlmair, „Blogs and Chats: Some Critical Remarks on Electronic Communication," *Didactics of Microlearning. Concepts, Discourses and Examples,* pp. 187-199, 2007.

[41] M. Elkan, „Chat, chatsprog og smileys," 22.05.2012. [Online]. http://www.elkan.dk/sprog/chat_smileys.asp. [Zugriff am 24.07.2012].

[42] P. Saint-Andre, „XEP-0001: XMPP Extension Protocols," 10.03.2010. [Online]. http://xmpp.org/extensions/xep-0001.html. [Zugriff am 05.08.2012].

[43] P. Saint-Andre und R. Meijer, „XEP-0107: User Mood," 29.10.2008. [Online]. http://xmpp.org/extensions/xep-0107.html. [Zugriff am 24.07.2012].

[44] P. Saint-Andre, „RFC6120: Extensible Messaging and Presence Protocol (XMPP): Core," 03 2011. [Online]. http://tools.ietf.org/html/rfc6120. [Zugriff am 11.08.2012].

[45] P. Saint-Andre, „RFC6121: Extensible Messaging and Presence Protocol (XMPP)," 03 2011. [Online]. http://tools.ietf.org/html/rfc6121. [Zugriff am 25.07.2012].

[46] P. Saint-Andre, „RFC6122: Extensible Messaging and Presence Protocol (XMPP): Address Format," 03 2011. [Online]. http://tools.ietf.org/html/rfc6122. [Zugriff am 11.08.2012].

[47] P. Saint-Andre und D. Smith, „XEP-0100: Gateway Interaction," 05.10.2005. [Online]. http://xmpp.org/extensions/xep-0100.html. [Zugriff am 05.08.2012].

[48] S. Ludwig, J. Beda, P. Saint-Andre, R. McQueen, S. Egan und J. Hildebrand, „XEP-0166: Jingle," 23.12.2009. [Online]. http://xmpp.org/extensions/xep-0166.html. [Zugriff am 05.08.2012].

[49] W3C Emotion Incubator Group, „W3C Incubator Group Report 10 July 2007," 10.07.2007. [Online]. http://www.w3.org/2005/Incubator/emotion/XGR-emotion/. [Zugriff am 06.08.2012].

[50] World Wide Web Consortium, „Vocabularies for EmotionML," 10.05.2012. [Online]. http://www.w3.org/TR/emotion-voc/. [Zugriff am 06.08.2012].

[51] World Wide Web Consortium, „Media Fragments URI 1.0 (basic)," 15.03.2012. [Online]. http://www.w3.org/TR/media-frags/. [Zugriff am 06.08.2012].

[52] M. Schröder, „Re: [emo] Response to your comment on the W3C EmotionML LCWD (ISSUE-175)," 22.06.2011. [Online]. http://lists.w3.org/Archives/Public/www-multimodal/2011Jun/0008.html. [Zugriff am 06.08.2012].

[53] N. Milikic, F. Radulovic, N. Kolundzija und A. Svitlica, „Smiley Ontology Specification," 30.04.2010. [Online]. http://www.smileyontology.com/spec/2010/SMILEY-20100430/. [Zugriff am 22.07.2012].

[54] J. Hastings, W. Ceusters, B. Smith und K. Mulligan, „The Emotion Ontology: Enabling Interdisciplinary Research in the Affective Sciences," Karlsruhe, Germany, 2011.

[55] J. Hastings, „emotion-ontology," 27.06.2012. [Online]. https://code.google.com/p/emotion-ontology/source/browse/trunk/ontology/MFOEM.owl. [Zugriff am 08.08.2012].

[56] B. Schuller, „Automatische Emotionserkennung aus sprachlicher und manueller Interaktion," 2006.

[57] D. Waskul und M. Douglass, „Cyberself: The Emergence of Self in On-Line Chat," *The Information Society: An International Journal,* Bde. %1 von %213, Issue 4, pp. 375-397,

1997.

[58] S. KATO, Y. KATO und D. SCOTT, „Relationships between Emotional States and Emoticons in Mobile Phone Email Communication in Japan," *International Jl. on E-Learning 8,* pp. 385-401, 2009.

[59] Statista GmbH, „Anteil der Internetnutzer, die Mobiltelefone und Smartphones besitzen in Deutschland von 2008 bis 2010," 2011.

[60] Statista GmbH, „Anteil der Mobilfunkkunden in Europa, die Instant Messaging Dienste nutzen, im Zeitraum von 2007 bis 2013," 2009.

[61] R. Cowie, E. Douglas-Cowie, K. Karpouzis, G. Caridakis, M. Wallace und S. Kollias, „Recognition of Emotional States in Natural Human-Computer Interaction," 2998.

[62] T. Zhang und G. Venture, „Emotion Recognition from Walk Pattern," The 21st Annual Conference of the Japanese Neural Network Society, 2011.

[63] T. K. Ming, „The study of ECG in relation to human emotions," 2008.

[64] K. H. Kim, S. Bang und S. Kim, „Emotion recognition system using short-term monitoring of physiological signals," *Med. Biol. Eng. Comput. 42,* pp. 419-427, 2004.

[65] R. Horlings, D. Datcu und L. J. M. Rothkrantz, „Emotion Recognition using Brain Activity," 2009.

[66] C. D. Katsis, N. Katertsidis, G. Ganiatsas und D. I. Fotiadis, „Toward Emotion Recognition in Car-Racing Drivers: A Biosignal Processing Approach," *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, VOL. 38, NO. 3,,* pp. 502-512, 05 2008.

[67] J. M. VALK und J. GROEN, „Electrical Resistance of the Skin During Induced Emotional Stress," *Psychosomatic Medicine,* Bd. 12 no. 5, pp. 303-314, 01.09.1950.

[68] V. Liao und R. Z. Moghaddam, „Measuring Everyday life Stress and Emotions by Affectiva Q Sensor," 2010.

[69] A. Azcarate, F. Hageloh, K. van de Sande und R. Valenti, „Automatic facial emotion recognition," Universiteit van Amsterdam, 2005.

[70] Statista GmbH, „Marktanteil der Instant Messager-Anbieter in Deutschland im Jahr 2008 (in Prozent)," 07 2008. [Online]. http://de.statista.com/statistik/daten/studie/37694/umfrage/marktanteil-der-instant-messaging-anbieter-in-deutschland-in-2008/. [Zugriff am 12.08.2012].

[71] J. LaPorte, „Worldwide Instant Messaging Market Share - July '08," 06.08.2008. [Online]. http://billionsconnected.com/blog/2008/08/global-im-market-share-im-usage/. [Zugriff am 12.08.2012].

[72] M. Seer, „Weltkarte der sozialen Netzwerke 2011," 08.03.2011. [Online]. http://t3n.de/news/weltkarte-sozialen-netzwerke-2011-300237/. [Zugriff am 12.08.2012].

[73] Statista GmbH, „Anzahl der weltweiten Nutzer von Google+ von Junli 2011 bis April 2012 (in Millionen)," 05 2012. [Online]. http://de.statista.com/statistik/daten/studie/215589/umfrage/prognose-zu-den-weltweiten-nutzerzahlen-von-google-plus/. [Zugriff am 12.08.2012].

[74] World Wide Web Consortium, „XPointer xpointer() Scheme," 19.12.2002. [Online]. http://www.w3.org/TR/xptr-xpointer/. [Zugriff am 28.07.2012].

[75] World Wide Web Consortium, „XML Path Language (XPath) 2.0 (Second Edition),"

14.12.2010. [Online]. http://www.w3.org/TR/xpath20/. [Zugriff am 28.07.2012].

[76] R. Fielding, „RFC1808: Relative Uniform Resource Locators,“ 06 1995. [Online]. http://tools.ietf.org/html/rfc1808. [Zugriff am 28.07.2012].

[77] World Wide Web Consortium, „XQuery 1.0 and XPath 2.0 Functions and Operators (Second Edition),“ 14.12.2010. [Online]. http://www.w3.org/TR/xpath-functions/. [Zugriff am 28.07.2012].

[78] P. Saint-Andre, „XEP-0071: XHTML-IM,“ 03.09.2008. [Online]. http://xmpp.org/extensions/xep-0071.html. [Zugriff am 25.07.2012].

[79] World Wide Web Consortium, „XHTML™ 1.0 The Extensible HyperText Markup Language (Second Edition),“ 26.01.2000. [Online]. http://www.w3.org/TR/xhtml1/. [Zugriff am 28.07.2012].

[80] Wold Wide Web Consortium, „HTML 4.01 Specification - 12 Links,“ 24.12.1999. [Online]. http://www.w3.org/TR/html401/struct/links.html#adef-rel. [Zugriff am 09.09.2012].

[81] M. Gaedke, „Vorlesungsmaterialien XML-Werkzeuge WS10/11,“ pp. 236-249, 28.01.2011.

[82] I. Davis, „Introducing Embedded RDF,“ 19.10.2005. [Online]. http://lists.w3.org/Archives/Public/semantic-web/2005Oct/0176.html. [Zugriff am 28.07.2012].

[83] World Wide Web Consortium, „XSL Transformations (XSLT) Version 2.0,“ 23.01.2007. [Online]. http://www.w3.org/TR/xslt20/. [Zugriff am 28.07.2012].

[84] microformats community, „microformats wiki,“ 17.07.2012. [Online]. http://microformats.org/wiki/Main_Page#Specifications. [Zugriff am 28.07.2012].

[85] World Wide Web Consortium, „RDFa Core 1.1,“ 07.06.2012. [Online]. http://www.w3.org/TR/rdfa-syntax/. [Zugriff am 28.07.2012].

[86] Web Hypertext Application Technology Working Group, „HTML Living Standard - 5 Microdata,“ 27.07.2012. [Online]. http://www.whatwg.org/specs/web-apps/current-work/multipage/microdata.html. [Zugriff am 28.07.2012].

[87] World Wide Web Consortium, „XHTML™ Basic 1.1 - Second Edition,“ 23.11.2010. [Online]. http://www.w3.org/TR/xhtml-basic/. [Zugriff am 29.07.2012].

[88] S. Josefsson, „RFC4648: The Base16, Base32, and Base64 Data Encodings,“ 10 2006. [Online]. http://tools.ietf.org/html/rfc4648. [Zugriff am 29.07.2012].

[89] S. B. Palmer, „RDF in HTML: Approaches,“ 31.05.2002. [Online]. http://infomesh.net/2002/rdfinhtml/#objectOrScript. [Zugriff am 29.07.2012].

[90] N. Freed und N. Borenstein, „RFC2045: Multipurpose Internet Mail Extensions (MIME) Part One,“ 11 1996. [Online]. http://tools.ietf.org/html/rfc2045. [Zugriff am 29.07.2012].

[91] World Wide Web Consortium, „Gleaning Resource Descriptions from Dialects of Languages (GRDDL),“ 11.09.2007. [Online]. http://www.w3.org/TR/grddl/. [Zugriff am 29.07.2012].

[92] P. Saint-Andre, „XEP-0203: Delayed Delivery,“ 15.09.2009. [Online]. http://xmpp.org/extensions/xep-0203.html. [Zugriff am 29.07.2012].

[93] P. Saint-Andre, „XEP-0082: XMPP Date and Time Profiles,“ 28.05.2003. [Online]. http://xmpp.org/extensions/xep-0082.html. [Zugriff am 29.07.2012].

[94] D. Dahl, „Announcing the Candidate Recommendation of Emotion Markup Language,“

17.05.2012. [Online]. http://lists.w3.org/Archives/Public/www-multimodal/2012May/0010.html. [Zugriff am 29.08.2012].

[95] W3C Multimodal Interaction Working, „EmotionML 1.0: Implementation Report Plan," 10.05.2012. [Online]. http://www.w3.org/2002/mmi/2012/emotionml-irp/. [Zugriff am 29.08.2012].

[96] D. Dahl, „EmotionML Implementation Report period extended," 19.06.2012. [Online]. http://lists.w3.org/Archives/Public/www-multimodal/2012Jun/0023.html. [Zugriff am 29.08.2012].

[97] XMPP Standards Foundation, „Libraries," 01.06.2009. [Online]. http://xmpp.org/xmpp-software/libraries/. [Zugriff am 31.08.2012].

[98] D. Eastlake und A. Panitz, „RFC2606: Reserved Top Level DNS Names," 06 1999. [Online]. http://tools.ietf.org/html/rfc2606. [Zugriff am 07.09.2012].

[99] G. Fobe, „little typo in specification," 08.08.2012. [Online]. https://groups.google.com/forum/?fromgroups=#!topic/smiley-ontology/qLdZGzC7b2M. [Zugriff am 04.09.2012].

[100] G. Fobe, „Wrong use of Timestamps in EmotionML," 21.06.2011. [Online]. http://lists.w3.org/Archives/Public/www-multimodal/2011Jun/0007.html. [Zugriff am 04.09.2012].

[101] M. Schröder, „Response to EmotionML LCWD comment ISSUE-191: Wrong use of timestamps in EmotionML," 07.09.2011. [Online]. http://lists.w3.org/Archives/Public/www-multimodal/2011Sep/0000.html. [Zugriff am 04.09.2012].

[102] M. Schröder, „Response to EmotionML LCWD comment ISSUE-192: DateTime instead milliseconds in EmotionML timestamps," 07.09.2011. [Online]. http://lists.w3.org/Archives/Public/www-multimodal/2011Sep/0001.html. [Zugriff am 04.09.2012].

[103] F. Burkhardt, „AW: EmotionML namespace prefix," 07.08.2012. [Online]. http://lists.w3.org/Archives/Public/www-multimodal/2012Aug/0001.html. [Zugriff am 04.09.2012].

[104] G. Fobe, „[emo] floating point number in <trace> element," 18.08.2012. [Online]. http://lists.w3.org/Archives/Public/www-multimodal/2012Aug/0005.html. [Zugriff am 04.09.2012].

[105] M. Schröder, „[emo] EmotionML Implementation report for MARY TTS," 14.06.2012. [Online]. http://lists.w3.org/Archives/Public/www-multimodal/2012Jun/0019.html. [Zugriff am 31.08.2012].

[106] F. Burkhardt, „[emo] EmotionML Implementation report for Speechalyzer," 18.06.2012. [Online]. http://lists.w3.org/Archives/Public/www-multimodal/2012Jun/0022.html. [Zugriff am 31.08.2012].

[107] A. Kazemzadeh, „Fwd: AW: Call for Implementations of EmotionML published," 29.08.2012. [Online]. http://lists.w3.org/Archives/Public/www-multimodal/2012Sep/0000.html. [Zugriff am 29.08.2012].

[108] The Unicode Consortium, „The Unicode Standard 6.0 - CJK Unified Ideographs," [Online]. http://www.unicode.org/charts/PDF/U4E00.pdf. [Zugriff am 22.07.2012].

[109] Wikimedia Commons, „File:Puck No212 p64f.png," 29.11.2007. [Online]. http://commons.wikimedia.org/wiki/File:Puck_No212_p64f.png. [Zugriff am 22.05.2012].

Anhänge

Anhang A: Inhalt der beiliegenden CD

- Diese Ausarbeitung im PDF-Format
- Softwareprojekte inklusive Git-Repositorys von
    - EmotionML-Bibliothek
    - Plattform mit Prototyp
    - Extension Plainemotion
    - Extension Emoticonhandler
- Bildschirmfotos aller verwendeten Internetquellen
- Eigene Bilder

Anhang B: PHP-Script zur Entgegennahme der Ausgaben vom Plugin Plainemotion

```php
<?php
/**
 * Part of project Vsr.Hawaii.
 * Outputs emotion with message and user of output plugin in plainemotion.
 * (c) 2012 Gerhard Fobe - public domain
 */
define('OUTPUT_FILE', 'emotionmessages.log');

function saveInput() {
	//read input
	$httpBody = @file_get_contents('php://input');

	$user = !isset($_GET['user'])
		? 'no user defined.'
		: $_GET['user'];

	$message = !isset($_GET['message'])
		? 'no message defined.'
		: $_GET['message'];

	$emotion = !isset($httpBody)
		? 'no EmotionML defined.'
		: $httpBody;

	//output in file
	$fp = @fopen(OUTPUT_FILE, 'a');
	if(!$fp) {
		die('Error during opening of file.');
	}
	fputs($fp, 'Time: ' . date('d.m.Y H:i:s') . "\r\n");
	fputs($fp, "User: $user \r\n");
	fputs($fp, "Message: $message \r\n");
	fputs($fp, "EmotionML: \r\n" . $emotion);
	fputs($fp, "\r\n\r\n"); //blank line

	fclose($fp);
}

function outputLog() {
	header('Content-Type: text/plain; charset=utf-8');
	readfile(OUTPUT_FILE);
}


//main
switch($_SERVER['REQUEST_METHOD']) {
case 'POST':
	saveInput();
	break;
case 'GET':
	outputLog();
	break;
default:
	header('HTTP/1.0 501 Not Implemented');
	break;
}
?>
```

Anhang C: Beispielkonfiguration für Gewichtung

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE rdf:RDF [
      <!ENTITY emochat "urn:emotionchat:config:" >
]>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
             xmlns:emochat="&emochat;">

      <!-- Weighting of emotion inputs -->
      <rdf:Description>
            <emochat:weighting>
                  <rdf:Bag>
                        <rdf:li rdf:parseType="Resource">
                              <emochat:plugin rdf:resource="&emochat;plugins:emoticonhandler:emoticon2emotion" />
                              <emochat:weightvalue>70</emochat:weightvalue>
                        </rdf:li>
                        <rdf:li rdf:parseType="Resource">
                              <emochat:plugin rdf:resource="&emochat;config:plugins:plainemotion:plainemotion" />
                              <emochat:weightvalue>20</emochat:weightvalue>
                        </rdf:li>
                  </rdf:Bag>
            </emochat:weighting>
      </rdf:Description>
</rdf:RDF>
```

Anhang D: Validierung und Vergleich der EmotionML-Bibliothek

Validierung der EmotionML-Bibliothek gegen die Testfälle des Implementation Report Plans [95] und Vergleich mit anderen Implementierungen. Die Testergebnisse für die EmotionML-Bibliothek befinden sich in der ersten Spalte.

Legende:

P	= pass
F	= fail
NI	= not implemented
Spec	= in Spezifikation
Req	= benötigt
Sub	= Unterbedingung

Andere Implementierungen:

1. MARY TTS (DFKI GmbH) [105]
2. Speechalyzer (Deutsche Telekom Innovation Laboratories) [106]
3. EMO20Q (USC SAIL Lab) [107]

	Testfälle für EmotionML					**Andere Implement.**		
		Feature	**Spec**	**Req**	**Sub**	**1**	**2**	**3**
	Document structure							
P	100		[2.1.1]	Y	N	P	P	P
P	101	emotionml	[2.1.1]	Y	N	P	P	P
P	102	emotionml	[2.1.1]	Y	N	P	P	P
P	103	emotion	[2.1.1]	N	N	P	P	P
P	104	vocabulary	[2.1.1]	N	N	NI	P	P
P	105	info	[2.1.1]	N	N	NI	P	NI
P	110	version	[2.1.1]	Y	N	P	P	P
P	111	version	[2.1.1]	Y	N	P	P	P
P	112	category-set	[2.1.1]	N	N	P	P	P
P	113	category-set	[2.1.1]	Y	N	P	P	P
NI	114		[2.1.1]	Y	Y	P	P	P
P	115	dimension-set	[2.1.1]	N	N	P	P	NI
P	116	dimension-set	[2.1.1]	Y	N	P	P	NI
NI	117		[2.1.1]	Y	Y	P	P	NI
P	118	appraisal-set	[2.1.1]	N	N	NI	P	NI
P	119	appraisal-set	[2.1.1]	Y	N	NI	P	NI
NI	120		[2.1.1]	Y	Y	NI	P	NI
P	121	action-tendency-set	[2.1.1]	N	N	NI	P	NI
P	122	action-tendency-set	[2.1.1]	Y	N	NI	P	NI
NI	123		[2.1.1]	Y	Y	NI	P	NI
P	124	emotionml	[2.1.1]	N	N	P	NI	NI
P	150	category	[2.1.2]	N	N	P	P	P
P	151	dimension	[2.1.2]	N	N	P	P	NI
P	152	appraisal	[2.1.2]	N	N	NI	P	NI
P	153	action-tendency	[2.1.2]	N	N	NI	P	NI
P	154	reference	[2.1.2]	N	N	NI	P	NI
P	155	info	[2.1.2]	N	N	NI	P	NI
P	156	emotion	[2.1.2]	Y	N	P	P	P
P	157	emotion	[2.1.2]	N	N	P	P	P
P	158	emotion	[2.1.2]	N	N	P	P	P
P	159	category-set	[2.1.2]	N	N	P	P	NI
P	160	category-set	[2.1.2]	Y	N	P	P	P

NI	161		[2.1.2]	Y	Y	P	P	P
P	162	dimension-set	[2.1.2]	N	N	P	P	NI
P	163	dimension-set	[2.1.2]	Y	N	P	P	NI
NI	164		[2.1.2]	Y	Y	P	P	NI
P	165	appraisal-set	[2.1.2]	N	N	NI	P	NI
P	166	appraisal-set	[2.1.2]	Y	N	NI	P	NI
NI	167		[2.1.2]	Y	Y	NI	P	NI
P	168	action-tendency-set	[2.1.2]	N	N	NI	P	NI
P	169	action-tendency-set	[2.1.2]	Y	N	NI	P	NI
NI	170		[2.1.2]	Y	Y	NI	P	NI
P	171	version	[2.1.2]	N	N	P	P	P
P	172	version	[2.1.2]	Y	N	P	P	P
P	173	id	[2.1.2]	N	N	NI	NI	NI
P	174	id	[2.1.2]	Y	N	NI	NI	NI
P	175	start	[2.1.2]	N	N	NI	NI	NI
P	176	end	[2.1.2]	N	N	NI	NI	NI
P	177	duration	[2.1.2]	N	N	NI	NI	NI
P	178	time-ref-uri	[2.1.2]	N	N	NI	NI	NI
P	179	time-ref-anchor-point	[2.1.2]	N	N	NI	NI	NI
P	180	offset-to-start	[2.1.2]	N	N	NI	NI	NI
P	181	expressed-through	[2.1.2]	N	N	NI	NI	NI
P	182	emotion	[2.1.2]	N	N	P	NI	NI
	Representations of emotions and related states							
P	210	category	[2.2.1]	Y	N	P	P	P
P	211	name	[2.2.1]	Y	N	P	P	P
NI	212		[2.2.1]	Y	Y	P	P	P
P	213	name	[2.2.1]	Y	N	P	P	P
P	214	value	[2.2.1]	N	N	NI	P	P
P	215	trace	[2.2.1]	N	N	NI	NI	NI
P	216	value / trace	[2.2.1]	Y	N	NI	NI	P
P	217	confidence	[2.2.1]	N	N	NI	P	NI
P	220	dimension	[2.2.2]	Y	N	P	P	NI
P	221	name	[2.2.2]	Y	N	P	P	NI
NI	222		[2.2.2]	Y	Y	P	P	NI
P	223	name	[2.2.2]	Y	N	P	P	NI
P	224	value / trace	[2.2.2]	Y	N	P	P	NI
P	225	confidence	[2.2.2]	N	N	NI	P	NI
P	230	appraisal	[2.2.3]	Y	N	NI	P	NI
P	231	name	[2.2.3]	Y	N	NI	P	NI
NI	232		[2.2.3]	Y	Y	NI	P	NI
P	233	name	[2.2.3]	Y	N	NI	P	NI
P	234	value	[2.2.3]	N	N	NI	P	NI
P	235	trace	[2.2.3]	N	N	NI	NI	NI
P	236	value / trace	[2.2.3]	Y	N	NI	NI	NI
P	237	confidence	[2.2.3]	N	N	NI	P	NI
P	240	action-tendency	[2.2.4]	Y	N	NI	P	NI
P	241	name	[2.2.4]	Y	N	NI	P	NI
NI	242		[2.2.4]	Y	Y	NI	P	NI
P	243	name	[2.2.4]	Y	N	NI	P	NI
P	244	value	[2.2.4]	N	N	NI	P	NI

P	245	trace	[2.2.4]	N	N	NI	NI	NI
P	246	value / trace	[2.2.4]	Y	N	NI	NI	NI
P	247	confidence	[2.2.4]	N	N	NI	P	NI
	Meta-information							
P	300	confidence	[2.3.1]	Y	N	NI	P	NI
P	301	expressed-through	[2.3.2]	Y	N	P	NI	NI
P	302	info	[2.3.3]	N	N	NI	NI	NI
P	303	info	[2.3.3]	N	N	NI	P	NI
NI	304	info	[2.3.3]	Y	N	NI	P	NI
P	305	id	[2.3.3]	N	N	NI	NI	NI
P	306	id	[2.3.3]	Y	N	NI	NI	NI
	References and time							
P	410	uri	[2.4.1]	Y	N	NI	P	NI
P	411	uri	[2.4.1]	Y	N	NI	P	NI
NI	412		[2.4.1]	N	Y	NI	NI	NI
P	413	role	[2.4.1]	N	N	NI	P	NI
P	414	role	[2.4.1]	Y	N	NI	P	NI
P	415	media-type	[2.4.1]	N	N	NI	NI	NI
P	416	media-type	[2.4.1]	Y	N	NI	NI	NI
NI	417		[2.4.1]	Y	Y	NI	NI	NI
P	420	start	[2.4.2]	Y	N	NI	NI	NI
P	421	end	[2.4.2]	Y	N	NI	NI	NI
P	422	duration	[2.4.2]	Y	N	NI	NI	NI
P	423	time-ref-uri	[2.4.2]	Y	N	NI	NI	NI
P	424	time-ref-anchor-point	[2.4.2]	Y	N	NI	NI	NI
P	425	offset-to-start	[2.4.2]	Y	N	NI	NI	NI
	Scale values							
P	500	value	[2.5.1]	Y	N	P	P	P
P	501	freq	[2.5.2]	Y	N	NI	NI	NI
P	502	freq	[2.5.2]	Y	N	NI	NI	NI
P	503	samples	[2.5.2]	Y	N	NI	NI	NI
P	504	samples	[2.5.2]	Y	N	NI	NI	NI
	Defining vocabularies for representing emotions							
P	600	item	[3.1.1]	Y	N	NI	P	P
P	601	info	[3.1.1]	N	N	NI	NI	NI
P	602	type	[3.1.1]	Y	N	NI	P	P
P	603	type	[3.1.1]	Y	N	NI	P	P
P	604	id	[3.1.1]	Y	N	NI	P	P
P	605	id	[3.1.1]	Y	N	NI	P	P
P	606	info	[3.1.2]	N	N	NI	NI	NI
P	607	name	[3.1.2]	Y	N	NI	P	P
P	608	name	[3.1.2]	Y	N	NI	P	P
	Conformance							
P	700		[4.1]	Y	N	P	P	P

Zusammenfassung:

Nr.	Name	pass	fail	not impl.
	EmotionML-Bibliothek	*105*	*0*	*15*
1	MARY TTS	39	0	81
2	Speechalyzer	83	0	37
3	EMO20Q	33	0	87

Anhang E: XML-Repräsentation des Implementation Reports

```
<system-report name="EmotionmlLibraryCSharp">
<testimonial>
Gerhard Fobe is happy to provide a library for C#, released under FreeBSD
license that can support developers to work with EmotionML. With the help
of the integrated EmotionML-parser it is possible to create related object
instances automatically. Furthermore new or existing object instances can
be converted to EmotionML too. Beside the standalone version of an
EmotionML document including emotions, vocabularies etc. the plug-in
version for the inclusion of emotions in other languages is supported.
</testimonial>
<assert id="100" res="pass"/>
<assert id="101" res="pass"/>
<assert id="102" res="pass"/>
<assert id="103" res="pass"/>
<assert id="104" res="pass"/>
<assert id="105" res="pass"/>
<assert id="110" res="pass"/>
<assert id="111" res="pass"/>
<assert id="112" res="pass"/>
<assert id="113" res="pass"/>
<assert id="114" res="not-impl"/>
<assert id="115" res="pass"/>
<assert id="116" res="pass"/>
<assert id="117" res="not-impl"/>
<assert id="118" res="pass"/>
<assert id="119" res="pass"/>
<assert id="120" res="not-impl"/>
<assert id="121" res="pass"/>
<assert id="122" res="pass"/>
<assert id="123" res="not-impl"/>
<assert id="124" res="pass"/>
<assert id="150" res="pass"/>
<assert id="151" res="pass"/>
<assert id="152" res="pass"/>
<assert id="153" res="pass"/>
<assert id="154" res="pass"/>
<assert id="155" res="pass"/>
<assert id="156" res="pass"/>
<assert id="157" res="pass"/>
<assert id="158" res="pass"/>
<assert id="159" res="pass"/>
<assert id="160" res="pass"/>
<assert id="161" res="not-impl"/>
<assert id="162" res="pass"/>
<assert id="163" res="pass"/>
<assert id="164" res="not-impl"/>
<assert id="165" res="pass"/>
<assert id="166" res="pass"/>
<assert id="167" res="not-impl"/>
<assert id="168" res="pass"/>
<assert id="169" res="pass"/>
<assert id="170" res="not-impl"/>
<assert id="171" res="pass"/>
<assert id="172" res="pass"/>
<assert id="173" res="pass"/>
<assert id="174" res="pass"/>
<assert id="175" res="pass"/>
<assert id="176" res="pass"/>
<assert id="177" res="pass"/>
<assert id="178" res="pass"/>
<assert id="179" res="pass"/>
<assert id="180" res="pass"/>
<assert id="181" res="pass"/>
<assert id="182" res="pass"/>
<assert id="210" res="pass"/>
<assert id="211" res="pass"/>
<assert id="212" res="not-impl"/>
<assert id="213" res="pass"/>
<assert id="214" res="pass"/>
<assert id="215" res="pass"/>
<assert id="216" res="pass"/>
<assert id="217" res="pass"/>
<assert id="220" res="pass"/>
<assert id="221" res="pass"/>
<assert id="222" res="not-impl"/>
<assert id="223" res="pass"/>
<assert id="224" res="pass"/>
<assert id="225" res="pass"/>
<assert id="230" res="pass"/>
<assert id="231" res="pass"/>
<assert id="232" res="not-impl"/>
<assert id="233" res="pass"/>
<assert id="234" res="pass"/>
<assert id="235" res="pass"/>
<assert id="236" res="pass"/>
<assert id="237" res="pass"/>
<assert id="240" res="pass"/>
<assert id="241" res="pass"/>
<assert id="242" res="not-impl"/>
<assert id="243" res="pass"/>
<assert id="244" res="pass"/>
<assert id="245" res="pass"/>
<assert id="246" res="pass"/>
<assert id="247" res="pass"/>
<assert id="300" res="pass"/>
<assert id="301" res="pass"/>
<assert id="302" res="pass"/>
<assert id="303" res="pass"/>
<assert id="304" res="not-impl"/>
<assert id="305" res="pass"/>
<assert id="306" res="pass"/>
<assert id="410" res="pass"/>
<assert id="411" res="pass"/>
<assert id="412" res="not-impl"/>
<assert id="413" res="pass"/>
<assert id="414" res="pass"/>
<assert id="415" res="pass"/>
<assert id="416" res="pass"/>
<assert id="417" res="not-impl"/>
<assert id="420" res="pass"/>
```

```
<assert id="421" res="pass"/>
<assert id="422" res="pass"/>
<assert id="423" res="pass"/>
<assert id="424" res="pass"/>
<assert id="425" res="pass"/>
<assert id="500" res="pass"/>
<assert id="501" res="pass"/>
<assert id="502" res="pass"/>
<assert id="503" res="pass"/>
<assert id="504" res="pass"/>
</system-report>
```

```
<assert id="600" res="pass"/>
<assert id="601" res="pass"/>
<assert id="602" res="pass"/>
<assert id="603" res="pass"/>
<assert id="604" res="pass"/>
<assert id="605" res="pass"/>
<assert id="606" res="pass"/>
<assert id="607" res="pass"/>
<assert id="608" res="pass"/>
<assert id="700" res="pass"/>
```