

Universität Hildesheim
Fachbereich III - Informations- und Kommunikationswissenschaften
Institut für Angewandte Sprachwissenschaft
Institut für Physik und Technik

Rollen in virtuellen Teams

Analyse und Simulation

Autoren: Ralph Kölle, GlennLangemeier

Arbeitsbericht vom 21.12.2005

1 Rollen in virtuellen Teams - Analyse und Simulation

Ziel des VitaminL-Projekts¹ war die Konzeption und Implementierung eines Systems zur Durchführung virtueller Tutorien. Dazu waren einerseits grundlegende Kommunikationswerkzeuge und Editoren zum gemeinsamen Bearbeiten von Quellcode umzusetzen. Als Ergänzung war es außerdem ein Ziel, den Teams auch außerhalb regulärer Tutorien jederzeit tutorielle Unterstützung zur Verfügung zu stellen.

Dieser virtuelle Tutor basiert auf einer Teamanalyse bzgl. des VitaminL-Rollenkonzepts und soll helfend im Rahmen fehlender Rollenkomponenten eingreifen. Generell ist auch ein paralleler Einsatz von menschlichen und computerbasierten Teletutoren denkbar. Auch könnte die Analysekomponente des virtuellen Tutors einen menschlichen Teletutor bei seiner Arbeit unterstützen.

Während das Kernsystem von VitaminL also aus einer kooperativen Programmierumgebung mit Werkzeugen zur gemeinsamen Bearbeitung von Quellcode und zur Kommunikation besteht und virtuellen Teams die Möglichkeit der zeit- und ortsunabhängigen kooperativen Bearbeitung von Übungsaufgaben oder Projekten gibt, spaltet sich die tutorielle Komponente in eine Analyse- und eine Simulationskomponente, die auf der Basis eines Rollenmodells aufeinander aufbauen.

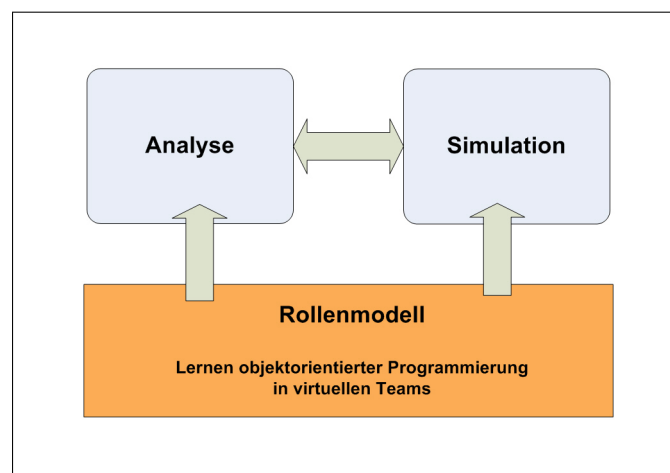


Abbildung 1: Das Rollenmodell als Basis

Dieser Arbeitsbericht beschreibt die Schnittstelle der Analyse- und der Simulationskomponente im VitaminL-System. Er setzt auf dem Bericht vom 11.12.2002 auf, stellt Bezüge her, revidiert auch durchaus Teilideen und beschreibt das technische Konzept beider Komponenten und insbesondere das der Schnittstelle zwischen beiden.

Es folgt zunächst ein kurzer Rückblick auf den vorausgegangenen Arbeitsbericht.

¹vgl. www.vitaminl.de

1.1 Rückblick

Der Arbeitsbericht vom 11.12.2002² gibt zunächst eine begriffliche Einführung im Kontext virtueller Teamarbeit und definiert danach zwei Forschungsthemen.

1. Rollenanalyse in virtuellen Teams: Analysen zur Zusammensetzung eines virtuellen Teams hinsichtlich vorhandener und fehlender Rollen
2. Ergänzung der Rollen in virtuellen Teams: Entwicklung und Evaluierung einer Methodik, um bestehende virtuelle Teams zielgerichtet zu ergänzen, indem fehlende Rollen durch geeignete tutorielle Komponenten simuliert oder unterstützt werden

Beide Themen sowie Abbildung 1 beinhalten als zentralen Begriff den der Rolle bzw. des Rollenmodells. Konzeptionell besteht die Idee der Forschungsthemen aus der Analyse von virtuellen Team bzgl. ihrer Rollenstruktur und der Ergänzung der Teams durch eine tutorielle Softwarekomponente im Rahmen von festgestellten Defiziten bzgl. eines definierten idealen Rollenprofils.

1.2 Grundlage: Das Rollenmodell

„Beim Aufbau virtueller Teams stellt sich die Frage nach der personellen Zusammensetzung. Obwohl die Auswahl von Mitgliedern in erster Linie aufgrund ihrer fachlichen Kompetenz erfolgt lassen sich eine Reihe von Soft-Skills nennen, die für einen optimalen Ablauf der Teamarbeit wichtig erscheinen.“ (Konradt & Hertel 2002, S. 52)

Die Zusammensetzung eines Teams scheint also Einfluß auf die Erreichung des Ziels zu haben. In Konradt & Hertel (2002) wird dabei dem Teamleiter besonderes Gewicht beigemessen: *„Eine Voraussetzung für den Erfolg virtueller Kooperation besteht in der Person des Teamleiters bzw. Moderators.“*

Daneben existieren aber auch noch weitere Rollen in einem Team, die teils notwendig und nützlich sind, teils aber auch kontraproduktiv im Sinne der Zielerreichung wirken. Sader (2002) klassifiziert die Teamrollen in aufgabenorientierte, gruppenprozeßorientierte und individuumszentrierte Rollen, ohne aber auf weitere Details einzugehen. In Kauffeld (2001) werden im Kapitel 4.3.6 mehrere Rollenmodelle näher beschrieben und die einzelnen Rollen charakterisiert. Dort finden wir bspw. die Rollenmodelle von McCann und Margerison (1989), von Spencer und Pruss (1995) oder auch die Teamrollen von Belbins.

Die Ermittlung von Rollen, also die Zuweisung bestimmten Verhaltens an Teammitglieder, ist Thema der Teamdiagnose. Kauffeld 2001 bietet einen umfassenden Überblick über die Prozesse der Teamentwicklung, gängige Rollenmodelle sowie Instrumente und Verfahren der Teamdiagnose. Dabei wird weiter unterschieden in prozessanalytische und strukturanalytische Verfahren (Kauffeld 2001, Kap. 4.2).

²vgl. <http://www.vitamin1.de/downloads/arbeitsbericht2002-12-11.pdf>

Im Bericht vom 11.12.2002 wurden vier Rollenmodell vorgestellt: TMS, Vygotsky, Eunson und Belbin. Das VitaminL-Rollenmodell basiert allerdings auf dem von Spencer&Pruss.

Ähnlich wie bei Belbin sind die Rollen nach technischen, integrierenden und sozialen Rollentypen unterteilt.

Technische Rollen übernehmen Aufgaben, die im Sinne technischer Teilprozesse die direkte Bearbeitung der Aufgabe betreffen. Integrierende Rollen handhaben die Aufgabe eher abstrakt und sorgen u.a. dafür, dass Teillösungen zusammengeführt werden. Soziale Rollen werden auch als kommunikationsorientierte Rollen bezeichnet. Sie schlichten Konflikte und sorgen für ein angenehmes Arbeitsklima.

S&P		VitaminL	Priorität	Typ
Entdecker	⇒	Informationsbeschaffer	++	technisch
Visionär	⇒	Planer	++	integrierend
Trainer	⇒	Berater	++	integrierend
Arbeitstier	⇒	Umsetzer	+	technisch
Friedensstifter	⇒	Schlichter	+	sozial
Pragmatiker	⇒	Problemlöser	+	technisch
Herausforderer	⇒	(In)Fragesteller	o	integrierend
Unparteiischer	⇒	Moderator ³	o	sozial
Bibliothekar	⇒	Archivar	o	technisch
Beichtvater	⇒	Vertrauensperson	-	sozial

Tabelle 1: Rollen bei Spencer&Pruss und bei VitaminL

Die genaue Beschreibung der einzelnen Rollen ist nicht Gegenstand dieses Berichts. Erwähnt sei jedoch, dass in einer ersten Version des Rollenmodells der Planer im hinteren Bereich mit niedriger Priorität zu finden war. Als Erbe des Visionärs (bei Spencer&Pruss) war er eher langfristig orientiert und passte nicht in den Kontext der kurzzeitigen Zusammenarbeit. Andererseits ist er derjenige, der den Arbeitsauftrag als Ganzes überschaut, unabhängig davon, ob dieser lang- oder kurzfristig orientiert ist. Nicht nur die Studierenden halten es für wichtig, dass es jemanden im Team gibt, der den Überblick behält. Daher wurde er in der Rolle des Planers auf Rang 2 nach oben gesetzt.

1.3 Forschungsthema I: Rollenanalyse in virtuellen Teams

Die Zusammensetzung virtueller Teams hinsichtlich vorhandener und fehlender Rollen soll analysiert werden. Grundlage ist die gemeinsame Bearbeitung von Aufgaben während einer Teamsitzung in der VitaminL-Lernumgebung.

³Beim Moderator handelt es sich nicht um die klassische Moderatorenrolle einer Diskussionsgruppe, die die Diskussion lenkt und aktiv unterstützt, sondern eher um einen neutralen Berater, der moderat in die Diskussion eingreift.

Im Bericht vom 11.12.2002 wurde ein Konzept für die Kommunikationsschnittstelle zur Teamdiagnose vorgestellt und verschiedene Diagnoseverfahren diskutiert. Mittlerweile kristallisiert sich ein auf Kennzahlen basierendes Verfahren mit folgenden Kernelementen heraus.

1.3.1 Analyse virtueller Teams

Auf der Basis von Befragungen (mittels eines elektronisch umgesetzten Fragebogens zu Teamrollen nach Spencer & Pruss 1995) und Beobachtungen (d.h. Aufzeichnungen aller Interaktionen eines virtuellen Teams während einer Arbeitssitzung) wurde eine Software-Komponente entwickelt, die Informationen über die Zusammensetzung eines virtuellen Teams während dessen Zusammenarbeit durch Aufzeichnung und Analyse der Kommunikation generiert. Dabei findet eine Codierung der Kommunikation auf Basis der sog. Collaborative Learning Skills statt (vgl. McManus & Aiken 1995; Soller 2004): Kommunikationsakte werden kategorisiert und auf geeignete Satzanfänge abgebildet, die nach Auswahl durch einen Benutzer sinngemäß zu vervollständigen sind, so dass auf eine komplexe inhaltliche Analyse verzichtet werden kann, was wiederum positiv Auswirkungen auf die Ausführungsgeschwindigkeit der Analyse nach sich zieht. Die Ergebnisse können verwendet werden, um dem Team in Problemsituationen angemessene tutorielle Unterstützung anzubieten. Die gesamte tutorielle Komponente besteht aus mehreren Teilkomponenten: Der Analyseagent registriert sämtliche Aktionen, die ein Teammitglied während einer Sitzung im System auslöst, und generiert daraus Informationen über Tendenzen des Teammitglieds bzgl. eines Rollenmodells. Die Aktionen beinhalten Kommunikation zwischen den Mitgliedern, aber auch Dokumentenoperationen im weitesten Sinne (Editieren, Navigieren, Übersetzen und Ausführen).

Die Analyse der Kommunikation bereitet die Information der Analyseagenten bzw. der zugehörigen Teammitglieder auf und liefert Kennzahlen über die Rollensstruktur des Teams nachgelagerte Komponenten weiter. Die Rollensimulation stellt ein virtuelles Teammitglied dar und berücksichtigt dabei so weit wie möglich die Rollenstruktur des virtuellen Teams, um Defizite innerhalb der Gruppe zu kompensieren und angemessenen tutoriellen Support in Problemsituationen anzubieten. Dabei wird auf Eingabedaten sowohl der Analysekomponente als auch der drei nachfolgenden Komponenten zurückgegriffen. Der Kommunikationsagent beobachtet die Kommunikation zwischen den Gruppenmitgliedern und meldet erkannte Problemsituation an die Rollensimulation. Der Codeagent untersucht die geöffneten Quelltextdateien der Teammitglieder auf mögliche Fehler und meldet diese ebenfalls an die Rollensimulation weiter. Der Navigationsagent schließt anhand der Benutzernavigation innerhalb der Client-Anwendung auf Probleme.

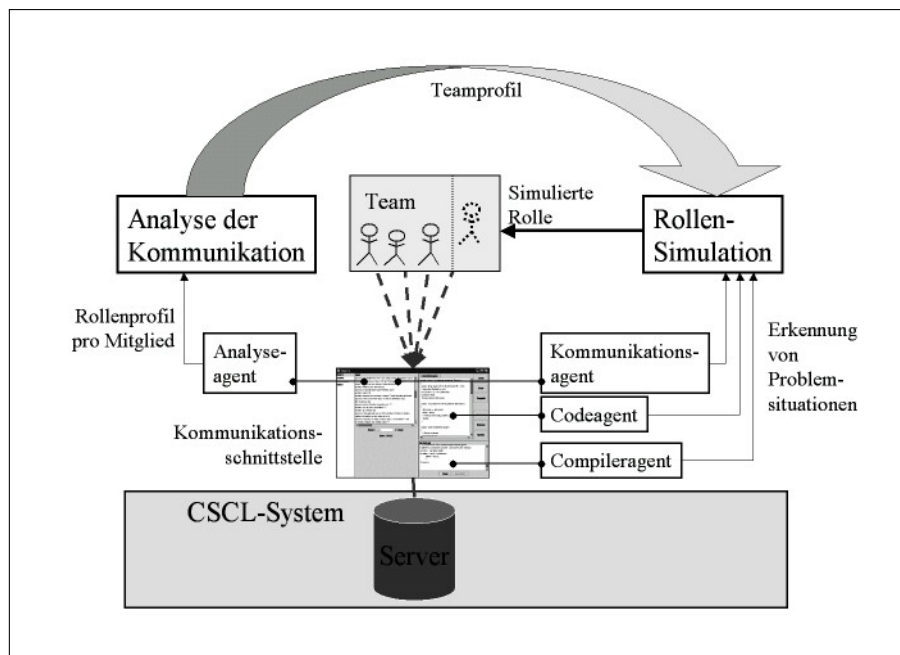


Abbildung 2: Das Gesamtsystem von VitaminL

Hier ist zunächst die Frage zu betrachten, wie die Kommunikation der Teammitglieder herangezogen werden kann, um Rollenstrukturen eines virtuellen Teams aufzudecken. Aus den Ergebnissen dieser Fragestellung ist anschließend eine Software-Komponente zu entwerfen, die das aus dieser Fragestellung resultierende Verfahren während einer Sitzung auf ein virtuelles Team anwendet und die Ergebnisse weiteren Komponenten für die Generierung von angemessenem tutoriellen Support in Problemsituationen zur Verfügung stellt.

1.3.2 CLS++

Die Kommunikationsanalyse verwendet als Ansatz eine strukturierte Kommunikation basierend auf den *Collaborative Learning Skills* (CLS) (vgl. McManus & Aiken 1995, Soller 2004). Die Kommunikation stellt jedoch nur einen Aspekt der Interaktion bei der objektorientierten Programmierung dar, denn letztlich bedeutet *Programmierung* auch *Programme erstellen*. Das Verfassen von Dokumenten (hier: Quelltexte in der objektorientierten Programmiersprache Java) stellt dabei zwar einen Schwerpunkt dar, jedoch sind weitere dokumentenbasierte Operationen wie das Übersetzen der Quelltexte durch den Java-Compiler oder das Ausführen der Programme durch den Java-Interpreter unbedingt zu berücksichtigen. Um nun auch dem Anspruch gerecht zu werden, virtuelle Teams bei der verteilten Programmierung zu unterstützen, müssen Konzepte zum Austausch von Dokumenten umgesetzt werden – dies ist in der VitaminL-IDE als *Shared Documents*-Komponente realisiert.

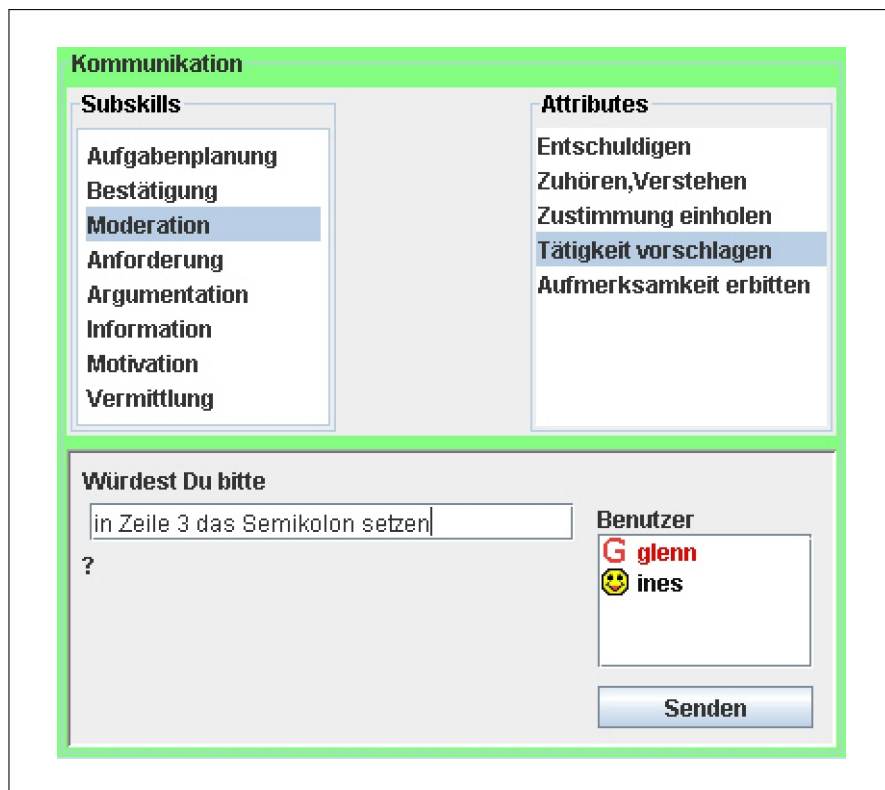


Abbildung 3: Strukturierte Kommunikationsschnittstelle

Insgesamt wird also die eigentliche Kommunikation ergänzt um eine Reihe von dokumentenbasierten Operationen, die als Ganzes den Funktionsumfang der VitaminL-IDE definieren und virtuellen Teams die objektorientierte Software-Entwicklung in Java ermöglichen. In Anlehnung an die objektorientierte Programmiersprache *C++* wird dieser gesamtheitliche Ansatz zur verteilten Programmierung im Rahmen des VitaminL-Projekts *CLS++* genannt.

1.3.3 Erkennen von Problemsituation

Wer ein Problem definiert, hat es schon halb gelöst. Dazu ist es zuallererst erforderlich, diejenigen Probleme näher zu spezifizieren, die während der gemeinsamen, synchronen Bearbeitung von Übungsaufgaben aus der Domäne der objektorientierten Java-Programmierung in einem CSCL-System auftreten können. Erst dann macht es überhaupt Sinn, sich mit der Frage nach den in diesem Kontext identifizierbaren Problemen zu befassen, um diese anschließend geeignet zu kategorisieren.

Das zuvor beschriebene System (s. Abb. 2) muss also in der Lage sein, zuverlässig zu entscheiden, wann und welche Art von Problem vorliegt. Es ist also zunächst wichtig, Problemsituationen während der Zusammenarbeit zu erkennen. Ansatzpunkte für eine solche Erkennung liefern neben der Gruppenkommunikation auch

die von den Mitgliedern bearbeiteten Quelltexte. Ergänzend dazu sollte natürlich auch jedes Gruppenmitglied selbst entscheiden können, ob aus seiner Sicht eine Problemsituation vorliegt, und dementsprechend tutoriellen Support anfordern.

1.3.4 Teamfunktionen und Teamzusammensetzung

„Es ist nun einmal so, dass bestimmte Funktionen erfüllt werden müssen, damit das Team effektiv arbeitet ...“ (Spencer & Pruss 1995, S.58).

Es stellt sich zunächst die Frage, welche Funktionen dies im Kontext der objektorientierten Programmierung sind. Es ist aber nicht nur zu klären, welche Funktionen prinzipiell benötigt werden, sondern vielmehr im konkreten Fall zu ermitteln, welche Funktionen tatsächlich von den Teammitgliedern übernommen werden und in welchem Maße dies geschieht.

Darüber hinaus ist es in diesem Zusammenhang auch sicherlich von Interesse, wie sich gute Teams von schlechten Teams unterscheiden lassen und ob sich infolgedessen möglicherweise optimale Gruppenzusammensetzungen für die synchrone Java-Programmierung in virtuellen Teams festlegen lassen.

1.3.5 Tutorieller Support

In einem tutoriellen System sind die Art, der Zeitpunkt und der Umfang der Unterstützung durch einen (virtuellen) Tutor entscheidend für die Effektivität der Hilfestellung. Erfolgt Support zu früh, ist zu befürchten, dass der gewünschte Lernerfolg ausbleibt, zu späte Hilfestellung mindert die Motivation der Teilnehmer. Hilfestellungen, die zu häufig erfolgen, können schnell als störend empfunden werden.

Ferner ist bei jedem Eingreifen eines Tutors zu berücksichtigen, in was für einer Problemsituation eingegriffen wird. Und schließlich muss die Zusammensetzung der jeweiligen Gruppe Beachtung finden, da unterschiedlich zusammengesetzte Teams sich auch hinsichtlich der benötigten Hilfestellung voneinander unterscheiden werden. Die Frage nach der *richtigen* Tutor-Strategie für virtuelle Teams lässt sich folglich in eine Vielzahl von Einzelfragen zerlegen.

1.3.6 Zielsetzung

Das Ziel der Analysekomponente besteht darin, im Hinblick auf die oben genannten Fragestellungen ein Kennzahlensystem zu entwickeln, welches – als Teil eines virtuellen Tutors – relevante Informationen zur Erkennung potentieller Problemsituationen liefert und als Entscheidungshilfe bei der Anwendung einer tutoriellen Strategie fungiert. Die einzelnen Elementen dieses Kennzahlensystems werden

während einer Gruppensitzung anhand der Aktionen der Teammitglieder berechnet und an nachgelagerte Komponenten weitergereicht, die ihrerseits angemessenen tutoriellen Support in Problemsituationen generieren.

Dazu ist zunächst die Frage zu betrachten, wie die Aktionen der Teammitglieder herangezogen werden können, um Rollenstrukturen eines virtuellen Teams aufzudecken. Desweiteren muss geklärt werden, welche Kennzahlen bei der Erkennung von Problemsituationen hilfreich sind. Die Ergebnisse der dazu notwendigen Untersuchungen sind in einer Software-Komponente umzusetzen und als Analyse-Agent in das Gesamtsystem zu integrieren (s. Abb. 2).

1.4 Forschungsthema II: Simulation von Rollen in virtuellen Teams

Die Simulation einer Rolle bedeutet, dass der virtuelle Tutor eine Rolle einnimmt, die dem Team offenbar fehlt und die für diese Rolle definierten Hilfsfunktionen ausführt.

Diese Hilfsfunktionen sind zu definieren, zu operationalisieren und den Rollen zuzuordnen. Im letzten Bericht wurde als technologischer Ansatz das MIMOR-Modell als Fusionsansatz vorgeschlagen, mit dem aus verschiedensten Informationsquellen Hilfsmaßnahmen ermittelt werden können. Dieser Ansatz wurde mittlerweile verworfen und durch einen Technologiebündel, bestehend aus einem Klassifizierer, einem Chatbot und einer Case-Based-Reasoning-Komponente (CBR) ersetzt.

Dabei setzen der Klassifizierer und der Chatbot die Funktionen der Rollen um, indem relevante Hilfsdokumente ermittelt werden und mittels Chatbot kommuniziert werden können, während den Kern der Schnittstelle zwischen Analyse und Simulation die CBR-Komponente bildet.

Neben dem Klassifizierer und dem Chatbot kommen bei der Operationalisierung der (Java-) Compiler und ein Dokumentenerzeuger zum Einsatz.

Tabelle 2 zeigt die Operationen der zehn VitaminL-Rollen. Die *externe Operation* zeigt, wie der virtuelle Tutor in der entsprechenden Rolle im VitaminL-Client agiert und welche *internen Operationen* dazu verwendet werden. Die Zuordnungen dieser Tabelle bilden somit die zentrale Schaltstelle des tutoriellen Konzepts und bilden die Arbeitsweise eines realen Tutors auf die technische Umsetzung des virtuellen Tutors ab.

Rolle	externe Operation	interne Operation			
		Klassifizierer	Chatbot	Compiler	Dokumenterzeugung
<i>technisch</i>					
Informationsbeschaffer	Präsentieren von Beispielen, Kommunizieren	✓	✓		
Umsetzer	<i>tritt nicht in Erscheinung</i>			✓	
Problemlöser	Präsentieren von übersetzten Fehlermeldungen	✓	✓	✓	✓
Archivar	Finden und Präsentieren benutzter Dateien und Kommunikationshistorie	✓	✓		✓
<i>integrierend</i>					
Planer	Zeitplan, Organisation		✓		✓
Berater	Ermutigung, Aufruf zur Mitarbeit		✓		
(In)Fragesteller	Motivieren inaktiver Mitarbeiter, Anregen einer Diskussion		✓		
<i>sozial</i>					
Schlichter	Schlichten sozialer Konflikte		✓		
Moderator	Off-Topic-Überwachung, Unterstützen des Beraters		✓		
Vertrauensperson	<i>tritt nicht in Erscheinung</i>				

Tabelle 2: Operationalisierung der Rollen

Es zeigt sich, dass jede Rolle bei ihren Aktionen auf den Chatbot zurückgreift, was die Wichtigkeit der Kommunikation im System wiederum manifestiert. Des Weiteren lässt sich an der Tabelle die Trennung nach fachlicher und personenbezogener („sozialer“) Hilfestellung ablesen⁴. Soziale Rollen verwenden ausschließlich den Chatbot, da sie in der Regel kommunikativ helfen, während technische Rollen eher auf technische Operationen (Klassifizierer, Compiler, Dokumentenerzeugung) zugreifen und der Chatbot nur zur Unterstützung verwendet wird.

Die Rolle der Vertrauensperson wird vernachlässigt. Sie ist sehr unstrukturiert, Beobachtungen und Befragungen zeigen, dass sie weitgehend unberücksichtigt bleiben kann⁵. Außerdem tritt auch der Umsetzer nicht in Erscheinung, da die Aufgabe der Umsetzung allein bei den Lernenden liegt. Seine Tätigkeit des Compilierens und die daraus entstandenen Fehlermeldungen kommuniziert er nur intern an den Problemlöser.

⁴*splitting role tutoring*, vgl. Kerres (2005, S.173)

⁵Im Prinzip übernimmt die Vertrauensperson ähnliche Funktionen wie ein Psychater und könnte ähnlich wie ELIZA implementiert werden (vgl. Weizenbaum 1966).

Bei den integrierenden Rollen kann bspw. der Planer zur Visualisierung eines Zeit- oder Organisationsplans auf die Operation der Dokume

Exemplarisch werden im Folgenden die technische Umsetzung des Klassifizierers sowie des Chatbots kurz beschrieben.

1.4.1 Der Klassifizierer

Ein Textklassifizierer wird verwendet, um bei Problemsituationen relevante Informationen in Form von Dokumenten aus einer Dokumentensammlung zu ermitteln, um sie dem Team danach bereitzustellen. Dabei kann es sich prinzipiell um jede Art von Dokument handeln, allerdings muss der Klassifizierer natürlich zur Dokumentenart passen. Außerdem sind klassifizierende Attribute a priori zu bestimmen. Als Dokumententyp sind zunächst Textdateien vorgesehen, Multimedia-Dateien wurden nicht betrachtet.

Zur Klassifizierung wurde ein *Naive Bayes*-Algorithmus⁶ aus der WEKA-Klassenbibliothek⁷ verwendet.

Im Fall des Informationsbeschaffers wird der Klassifizierer dazu verwendet, zu einem mutmaßlich problembehafteten Quelltext-Dokument ein syntaktisch und semantisch korrektes Beispiel aus einer Beispieldokumenten-Kollektion zu ermitteln und der Gruppe zu präsentieren.

Bei dem Verfahren handelt es sich um einen klassischen, für Quellcode optimierten Information Retrieval Prozess. Alle Beispieldokumente werden im Vorfeld indexiert, in einer Problemsituation wird das Problemdokument ebenfalls (nach dem gleichen Verfahren) indexiert und das ähnlichste Dokument aus der Kollektion als Beispiel ausgewählt. Als Indexterme werden Schlüsselwörter von Java⁸ und häufig verwendete Klassen- und Methodennamen aus dem JDK⁹ verwendet. Die Auswahl der Beispiele geschieht rein nach den a priori ausgewählten Indextermen, Metatags sind im ersten Ansatz nicht vorgesehen.

Die Idee besteht darin, den Studierenden bspw. ein Beispiel mit einer *for-Schleife* zu präsentieren, wenn deren Problemdokument selbst eine *for-Schleife* enthält oder das Problem konkret bei der Syntax der *for-Schleife* auftritt.

Abbildung 4 zeigt das Prinzip des Klassifizierens zur Ermittlung und Präsentation von Dokumenten als Hilfestellung.

⁶Dieses in der Praxis häufig zur Textklassifizierung (vgl. Joachims 2002) eingesetzte Verfahren bedient sich der statistischen Modellierung von Wahrscheinlichkeiten nach Bayes.

⁷Weka (Waikato Environment for Knowledge Analysis) ist eine an der Universität von Waikato entwickelte in Java implementierte Klassenbibliothek, die verschiedene induktive Lernverfahren, Methoden zur Datenaufbereitung und zur Evaluation der Lernergebnisse bietet (vgl. Frank & Witten 2001, S.291).

⁸Aufzählung der Java-Schlüsselwörter: z.B.

http://java.sun.com/docs/books/tutorial/java/nutsandbolts/_keywords.html

⁹Java Development Kit

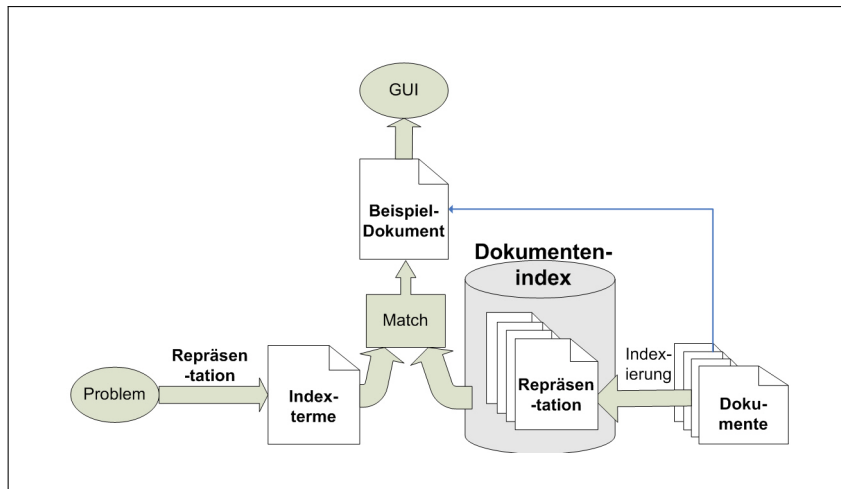


Abbildung 4: Prinzip der Dokumentenauswahl

Ein Problem (hier: das fehlerhafte Dokument) wird mittels geeigneter Indexterme (hier: Java-Schlüsselworte etc.) repräsentiert. Diese Repräsentation wird zur Klassifizierung verwendet und mit dem Dokumentenindex der Beispielsammlung abgeglichen. Das ähnlichste Dokument der Sammlung wird als Beispiel ausgewählt und dem Team in der Benutzeroberfläche präsentiert.

Dieses Konzept ist technisch sehr offen gestaltet. Es lässt sich auf verschiedene Situationen und Anwendungsfälle adaptieren. Durch Austausch der kompletten Dokumentensammlung ist ein einfacher Wechsel der Wissensdomäne denkbar. Der qualitative Anspruch liegt in der Dokumentensammlung, im Beispiel also in der Auswahl geeigneter Beispieldateien und der Auswahl geeigneter beschreibender Attribute. Die Attribute und Dokumente liegen in Textdokumenten vor und lassen sich so ohne Eingriff in die Kernsoftware erweitern, anpassen und natürlich für den Klassifizierer indexieren.

1.4.2 Der ALICE-Chatbot

Die technische Umsetzung der Kommunikation des virtuellen Tutors basiert auf einer professionellen Chatbot-Software ALICE¹⁰. Konzeptionell entscheidend ist auch hier, dass das Konzept offen gestaltet ist und sich so ohne Programmierkenntnisse anpassen lässt. Das ist durch die strenge Trennung des Kommunikationsmodells (Wissensmodell, Wissensbasis, vgl. Abbildung 5) von der Verarbeitungskomponente und insbesondere vom VitaminL-Kernsystem möglich.

¹⁰Artificial Linguistic Internet Computer Entity, <http://www.alicebot.org/>

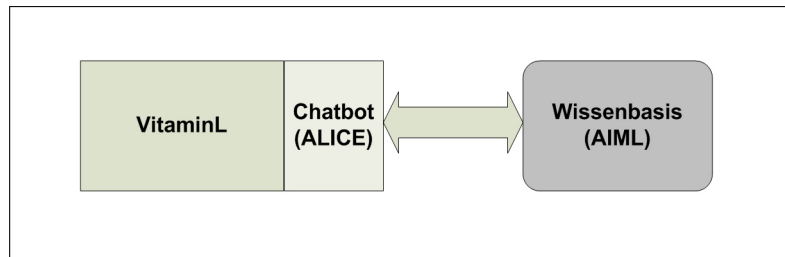


Abbildung 5: Integration des Chatbot

Ein ähnliches Konzept verfolgt auch das Catacomb-Projekt (vgl. Ginsburg 2002), das ebenfalls die Integration der ALICE-Software als Kommunikationskomponente in Web-Portale vorschlägt.

ALICE

Die ALICE-Foundation hat eine Reihe von Implementierungen verschiedener Programmiersprachen hervorgebracht. Für die Integration in VitaminL wurde das quelloffene und in der Programmiersprache Java vorliegende ALICE-D-Derivat verwendet¹¹. Bekannt geworden ist die ALICE-Software hauptsächlich durch den dreifachen Gewinn des Loebner-Preises¹² in der Kategorie des menschen-ähnlichsten Chatbot (*“most human computer“*). Für die Programmierung der Wissensbasis wurde im ALICE-Kontext eine eigene, XML-konforme Modellierungssprache AIML entwickelt¹³. Das Kommunikationsmodell liegt also in Form von externen Textdateien vor und lässt sich so relativ einfach anpassen.

AIML - Artificial Intelligence Markup Language

AIML ist eine auf XML-basierte Auszeichnungssprache, deren grundlegende Einheiten Kategorien, Muster und Templates sind¹⁴.

Mit Kategorien teilt man die Kommunikationsbasis in verschiedene Bereiche, Muster enthalten Kommunikationsphrasen, auf die der Chatbot reagieren kann, die Reaktion erfolgt über verschiedene Templates, von denen eins zufällig als Antwort ausgewählt wird.

Zusätzlich besteht mit dem *<think>*-Tag u.a. die Möglichkeit, innere Zustände des Chatbots in Variablen zu speichern und mit dem *<topic>*-Tag Kategorien in verschiedene Themenbereiche zu unterteilen, um so den Kommunikationsverlauf besser steuern zu können.

Im folgenden Beispiel wird die Variable *topic* auf „*Problem*“ gesetzt. Der Bot „weiß“

¹¹Einen guten Einstieg in die Programmierung auf Basis der ALICE-D-Software gibt: <http://www.alicebot.org/resources/programd/readme.html>

¹²Loebner-Preis: <http://www.loebner.net/Prizef/loebner-prize.html>

¹³Einen guten Einstieg in die grundlegende Architektur und Programmierung von ALICE gibt z.B. Shawar & Atwell (2004).

¹⁴Einen sehr guten Einstieg in die Programmierung AIML-basierter Chatbots gibt Banach (2004).

nun, dass er im Folgenden Kategorien auswählen sollte, die diesem Thema zugeordnet sind.

Syntaktisch sieht eine Kategorie bspw. wie folgt aus:

```
<category>
  <pattern>WIR HABEN EIN PROBLEM</pattern>
  <template>
    <think><set name="topic">Problem</set></think>
    Worum handelt es sich?
  </template>
</category>
```

Im diesem Beispiel reagiert der Bot auf das Muster „*Wir haben ein Problem*“¹⁵ mit der Gegenfrage „*Worum handelt es sich?*“¹⁶.

Die grundlegende Nutzung von AIML wurde schon bei der Beschreibung des Wissensmodells erläutert. Da es auf XML basiert, gibt es eine Reihe von Werkzeugen zum Erzeugen und Editieren von AIML-Dateien. Weitere Vorteile von AIML sind:

- leicht erlernbar, trotzdem sind komplizierte Konstrukte möglich
- lässt sich (wegen XML) leicht erweitern
- AIML-Parser sind in vielen Programmiersprachen (frei) erhältlich

(vgl. Engelmann et al. 2004, S.239f)

In VitaminL ist zunächst nur eine kleine Wissensbasis mit acht Kategorien implementiert worden, da sich die Kommunikation im Rahmen der Informationsbeschaffer-Rolle auf das Präsentieren von Beispieldokumenten, den Aufruf zum Compilieren und Rückfragen zwecks Feedback beschränkt. Jede Kategorie enthält mehrere Antwortalternativen, von denen ALICE nach einem Zufallsprinzip jeweils eine auswählt. Dieses Zufallsprinzip ist in ALICE hinterlegt und macht die Kommunikation natürlicher, da nicht immer der gleiche Satz für dieselbe Aussage ausgewählt wird (vgl. AIML-Tags *<random>* und **).

Bei Rollen, die neben der Kommunikation noch andere Operationen verwenden, müssen diese aufeinander abgestimmt werden. Im Fall des Informationsbeschaffers wird bspw. das Präsentieren von Beispielen durch kommunikative Akte begleitet, damit die Gruppe die Präsentation nicht übersehen kann.

¹⁵Muster werden in AIML ausschließlich mit GROßBUCHSTABEN formuliert.

¹⁶Mittels *<random>*- und **-Tags kann man zusätzlich mehrere Antworten definieren, aus der zufällig eine ausgewählt wird.

1.5 Die Schnittstelle zwischen Analyse und Simulation

Die Schnittstelle zwischen Analyse und Simulation basiert auf fünf Bereichen: Informationen über Problemsituationen, dem Rollenprofil des Teams, Daten über dessen (theoretischen) Kenntnisstand, die Information, bei wem das Problem aufgetreten ist und ergänzende Informationen, die von der Art der Problemsituation abhängen. Das kann im Fall eines fachlichen Problems z.B. die bearbeitete Datei sein, in der das Problem auftritt.

Die Analysekomponente sammelt während der Teamarbeit Daten sämtlicher Aktivitäten und bereitet diese zu Kennzahlen auf. Typische Kennzahlen sind dabei Bearbeitungs-, Kommunikations- und Navigationsanteile jedes Benutzers und der gesamten Gruppe. Aus den Kennzahlen wird ermittelt, ob die Gruppe oder einzelne Benutzer Probleme haben. Für die Ermittlung formaler, fachlicher Probleme im Quelltext wird der Java-Compiler herangezogen.

Zusätzlich ermittelt die Analysekomponente auf Basis der strukturierten Kommunikationsschnittstelle (vgl. Abbildung 3) Rollenprofile der einzelnen Benutzer und des Teams. Das Teamprofil wird an die Simulation übermittelt.

Hinzu kommt die Information, bei welchem Mitglied das Problem aufgetreten ist. Das ist insbesondere bei Problemen sozialer Art von Interesse. Erkennt die Analyse eine andauernde Inaktivität eines Teammitglieds, ist es für die Simulation wichtig, bei wem diese aufgetreten ist, um gezielt reagieren zu können.

1.5.1 Problemsituationen

Eine Problemsituation wird in Anlehnung an den Problem-Begriff bei Zimbardo & Gerrig (2000) als eine Situation definiert, bei der ein Problemlösungsprozess aufgrund eines auftretenden Problems ins Stocken gerät.

Der virtuelle Tutor unterscheidet dabei zwischen fachlichen und sozialen Problemen. Fachliche Probleme werden nach Syntax-, Semantik- und Logikproblemen klassifiziert. Grundlage dieser Klassifizierung sind die Ergebnisse einer Forschungsarbeit zur Erkennung von Problemsituationen bei (Programmier-)Anfängern (vgl. Bischoff 2005). Ein zentrales Ergebnis dieser Arbeit ist die erwähnte Fehlerklassifikation nach Syntax-, Semantik- und Logikfehlern sowie die Ermittlung von Fehlerhäufigkeiten bei der virtuellen Teamarbeit. Demnach ist die häufigste Fehlerart (74%, 166 von 224) die Verletzung der syntaktischen Konventionen von Java, gefolgt von 33 Logikfehlern und 11 Semantikproblemen. Die restlichen 14 protokollierten Fehler ließen sich nicht eindeutig einordnen.

Syntaxfehler werden dabei in Fehler bei *Punktzeichen*, *Klammern*, *Variablen*, *Datentypen/Arrays*, *Klassen/Methoden*, *Vererbung* und *andere* Fehler differenziert (vgl. Bischoff 2005, S.68). Diese können durch den Compiler erkannt werden.

Semantikfehler lassen sich in der Regel erst zur Laufzeit ermitteln und lösen (in Java) meistens eine Exception aus. Dabei kann es sich z.B. um den versuchten Zugriff auf nicht vorhandene Array-Indizes handeln (*ArrayIndexOutOfBoundsException*) oder die Nichtverwendung des Rückgabewerts einer Methode.

Logikfehler sind manchmal nicht eindeutig von Semantikfehlern abzugrenzen. Dazu zählen aufgabenspezifische Fehler (falsch positionierte Ausgabeanweisungen, falsche Startwerte bei Schleifen, falsche Abbruchbedingungen und falsche boolesche Verknüpfungen).

Fachliche Probleme beziehen sich in der Regel auf eine konkret bearbeitete Datei, in der ein Fehler auftritt. Diese Datei wird von der Analysekomponente ermittelt und an die Simulation übergeben, sodass die Hilfe dem Kontext angepasst werden kann.

In die Kategorie der sozialen Probleme fallen:

- Konflikte (bei der Kommunikation)
- Motivationsverlust, mangelnde Motivation
- Offtopic-Aktivitäten bei der Kommunikation
- Inaktivitäten einzelner oder der ganzen Gruppe
- Missverständnisse (bei der Kommunikation)

Diese Probleme sind anhand der Logfileanalyse nach und bei Benutzertests und Sitzungen der Kooperationsveranstaltung ermittelt worden. Eine vertiefende linguistische Analyse, wie sie bei Göldner (2005) stattgefunden hat, wurde nicht durchgeführt¹⁷.

1.6 Case-Based Reasoning

CBR ist eine Technik, die vorhandenes Erfahrungswissen, das in Form von sogenannten Fällen in einer Fallbasis gespeichert ist, zur Lösung neuer Probleme wiederverwendet.

„A case-based reasoner solves new problems by adapting solutions that were used to solve old problems.“ (Riesbeck & Schank 1989, S.25)

Die Analogie zur Arbeitsweise des virtuellen Tutors bei VitaminL ist unmittelbar erkennbar. Es existiert eine Problemsituation, zu der auf Basis weiterer spezieller Eigenschaften (Rollenprofil, Wissensstand) eine Lösung gesucht wird. Nach der Präsentation der Lösung lässt sich durch Benutzer-Feedback die Güte der Lösung prüfen.

¹⁷Göldner (2005) untersuchte im Rahmen ihrer Forschungsarbeit die Kommunikation bei virtuellen Tutorien auf Basis der VitaminL-Chatprotokolle. Mittels Methoden der Angewandten Diskursforschung identifizierte sie Problemsituationen auf drei Analyseebenen: der pragmalinguistische Ebene, der Ebene der Wissenskommunikation und der Ebene des dialogischen Softwareentwurfs. Neben Maßnahmen auf der Verhaltensseite (Trainingsmaßnahmen) schlägt sie systemseitig u.a. Verbesserungen zur effektiveren Referenzierung von Chatbeiträgen und Suchmöglichkeiten in älteren Beiträgen vor (vgl. Göldner 2005, S.96ff).

Für eine CBR-Wissensbasis sind die Probleme zu formalisieren. Neben den genannten Problemsituationen fachlicher oder sozialer Art wird ein Problem außerdem durch die Rollenprofile identifiziert. Hinzu kommen Metadaten zum Kenntnisstand der Lerner in Form eines numerischen Wertes, als Maß kommt z.B. die Semesterwoche in Frage. Ein Problem im Sinne eines Problem-Lösungspaares (ein Fall des CBR) besteht folglich aus:

1. der Problemsituation (fachlich, sozial)
2. der Fehlerklasse bei fachlichen Problemen (Syntax, Semantik, Logik)
3. der Fehlerunterklasse (bei Syntax: Punktzeichen, Klammern,...)
4. den zehn Rollenprofilen
5. dem Kenntnisstand der Lerner (Semesterwoche)

Die Rollenprofile fließen jedoch nicht unmittelbar ein, sondern es wird ein *relatives Rollenprofil* der Gruppe ermittelt, indem die aktuellen Werte der einzelnen Rollen von den Werten des *idealen Rollenprofils* abgezogen werden. So lassen sich an negativen Werten unmittelbar Rollendefizite erkennen.

Eine Lösung im Konzept des virtuellen Tutor besteht selbst wiederum aus einem Paar: einer kommunikativen und einer fachlichen (Teil-)Lösung. Wie schon angedeutet kann der virtuelle Tutor im Chat mit der Gruppe kommunizieren und im Multi-Document-Editor beliebige Dokumente öffnen.

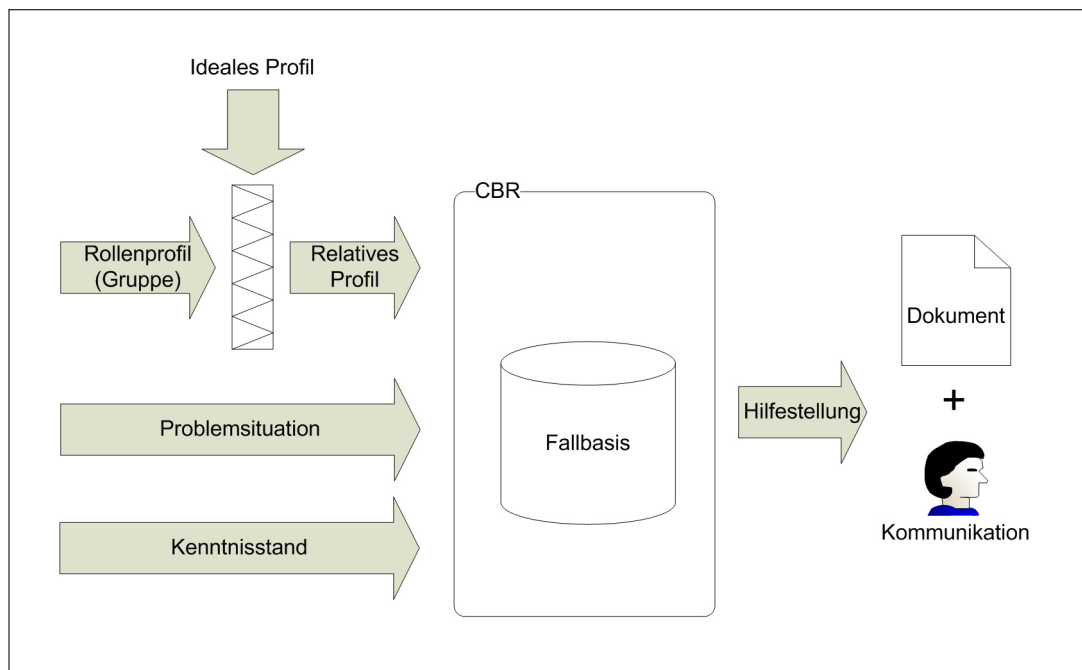


Abbildung 6: Einsatz des CBR für Hilfestellungen

Grundsätzlich kann ein virtueller Tutor natürlich auch fachliche Hilfe in ausschließlich kommunikativer Form anbieten. Allerdings lassen sich viele Inhalte durch Präsentation eines Dokuments prägnanter und effizienter erläutern als durch ausschließliche Verbalisierung. Insbesondere bei der Programmierung werden häufig Beispiele zur Veranschaulichung von Sachverhalten verwendet. Natürlich müssen nicht in jedem Fall Dokumente präsentiert werden. Hilfestellungen sozialer Art wie bspw. Motivation zur Mitarbeit werden tendenziell eher kommunikativ stattfinden¹⁸.

1.7 Vision

Der virtuelle Tutor sollte zum einen auf der Analyseseite zuverlässig Problemsituationen erkennen und diese an die Simulationskomponente weiterreichen, die im Rahmen der Rolle, bei der ein Defizit zu erkennen ist, geeignete Hilfsmaßnahmen durchführt. Diese Hilfsmaßnahmen bestehen in der Regel aus der Präsentation einer inhaltstragenden Datei und der begleitenden Kommunikation. Allerdings ist es für sozial geprägte Rollen wünschenswert, dass der Chatbot in der Lage ist, *freie* Kommunikation durchzuführen, sodass ein echter Dialog zwischen Studierenden und dem virtuellen Tutor entstehen kann.

¹⁸Die Trennung nach fachlicher und sozialer bzw. personen- oder gruppen-bezogener Unterstützung, wie sie auch Kerres & Jechle (2000) vorschlagen, lässt sich bei Tele-Tutoren organisatorisch und personal-technisch nur äußerst schwer umsetzen. Bei einer software-technischen Umsetzung wie im Falle des virtuellen Tutors stellen sich solche Probleme nicht.

Literatur

[Banach 2004]

BANACH, Zbigniew: Wir erzeugen einen eigenen Bot. In: *Software 2.0 Extra! - Künstliche Intelligenz* (2004), S. 6–11

[Bischoff 2005]

BISCHOFF, Kerstin: *Objektorientierte Softwareentwicklung in virtuellen Teams - Modellierung und Ansätze zur automatischen Erkennung von Problemsituationen*, Universität Hildesheim, Magisterarbeit im Fach *Internationales Informationsmanagement*, 2005

[Engelmann et al. 2004]

ENGELMANN, Hagen; LÁBBATE, Marcello; THIEL, Ulrich: Beratungsdialo g im WWW: Ein konversationales Modell und seine Implementierung. In: BEKAVAC, B.; HERGET, J.; RITTBERGER, M. (Hrsg.): *Information zwischen Kultur und Marktwissenschaft; Proceedings des 9. Internationalen Symposiums für Informationswissenschaft (ISI 2004)*. Konstanz: UVK Verlagsgesellschaft mbH, 2004 (Schriften zur Informationswissenschaft 42), S. 227–245

[Frank & Witten 2001]

FRANK, Eibe.; WITTEN, Ian H.: *Data Mining. Praktische Werkzeuge und Techniken für das maschinelle Lernen*. München und Wien: Carl Hanser Verlag, 2001. – Aus dem Amerikanischen übersetzt. (2000, *Practical Machine Learning Tools and Techniques with Java Implementation*. San Fransisco: Morgan Kaufmann Publishers)

[Ginsburg 2002]

GINSBURG, Mark: The catacomb project: building a user-centered portal the conversational way. In: *WIDM '02: Proceedings of the 4th international workshop on Web information and data management*. New York: ACM Press, 2002, S. 84–87

[Göldner 2005]

GÖLDNER, Antje: *Computerunterstützte Kommunikation in virtuellen Teams. Klassifikations- und Lösungsansätze für Problemsituationen in der Chatkommunikation im Rahmen objektorientierter Programmierung*, Universität Hildesheim, Magisterarbeit im Fach *Internationales Informationsmanagement*, 2005

[Joachims 2002]

JOACHIMS, Thorsten: *Learning to Classify Text Using Support Vector Machines: Methods, Theory and Algorithms*. Norwell: Kluwer Academic Publishers, 2002

[Kauffeld 2001]

KAUFFELD, Simone: *Teamdiagnose*. Göttingen, Bern, Toronto, Seattle: Hogrefe, 2001

[Kerres 2005]

KERRES, Michael: Didaktisches Design und eLearning. In: MILLER, D. (Hrsg.): *E-Learning. Eine multiperspektivische Standortbestimmung*. Haupt, 2005, S. 156–182

[Kerres & Jechle 2000]

KERRES, Michael; JECHLE, Thomas: Betreuung des mediengestützten Lernens in

telemedialen Lernumgebungen. In: *Unterrichtswissenschaft* 28 (2000), Nr. 3, S. 257–277

[Konradt & Hertel 2002]

KONRADT, Udo; HERTEL, Guido: *Management virtueller Teams – Von der Telearbeit zum virtuellen Unternehmen*. Weinheim, Basel: Beltz, 2002

[McManus & Aiken 1995]

MCMANUS, Margaret M.; AIKEN, Robert M.: Monitoring computer-based problem solving. In: *Journal of Artificial Intelligence in Education* (1995), Nr. 6, S. 307–336

[Riesbeck & Schank 1989]

RIESBECK, Christopher K.; SCHANK, Roger C.: *Inside Case-Based Reasoning*. Hillsdale: Lawrence Erlbaum Associates, 1989

[Sader 2002]

SADER, Manfred: *Psychologie der Gruppe*. Weinheim: Juventa, 2002

[Shawar & Atwell 2004]

SHAWAR, Bayan A.; ATWELL, Eric: Accessing an Information System by Chatting. In: MEZIANE, F.; MÉTAIS, E. (Hrsg.): *Natural Language Processing and Information Systems, Proceedings of the 9th International Conference on Applications of Natural Languages to Information Systems, NLDB 2004*, Springer, 2004 (Lecture Notes in Computer Science), S. 407–412

[Soller 2004]

SOLLER, Amy: Computational Modeling and Analysis of Knowledge Sharing in Collaborative Distance Learning. In: *User Modeling and User-Adapted Interaction* 14 (2004), January, Nr. 4, S. 351–381

[Spencer & Pruss 1995]

SPENCER, John; PRUSS, Adrian: *Top Teams - Der Königsweg zu mehr Flexibilität, Effizienz und Erfolg im Betrieb*. München: Knauer, 1995

[Weizenbaum 1966]

WEIZENBAUM, Joseph: ELIZA - A Computer Program for the Study of Natural Language Communication between Man and Machine. In: *Communications of the Association for Computing Machinery* 9 (1966), S. 36–45

[Zimbardo & Gerrig 2000]

ZIMBARDO, Philip G.; GERRIG, Richard J.: *Psychologie*. 7. Auflage. Berlin: Springer, 2000

Abbildungsverzeichnis

1	Das Rollenmodell als Basis	1
2	Das Gesamtsystem von VitaminL	5
3	Strukturierte Kommunikationsschnittstelle	6
4	Prinzip der Dokumentenauswahl	11
5	Integration des Chatbot	12
6	Einsatz des CBR für Hilfestellungen	16

Tabellenverzeichnis

1	Rollen bei Spencer&Pruss und bei VitaminL	3
2	Operationalisierung der Rollen	9