

Diss

Stefan Oppl

31. August 2009

Inhaltsübersicht

I. Einführung	1
I. Grundlagen	5
2. Articulation Work	9
3. Mentale Modelle	63
II. Unterstützung	87
4. Anforderungen an ein Werkzeug	91
5. Grundlagen der Implementierung	93
6. Eingabe und Interpretation	131
7. Ausgabe	197
8. Persistierung	231
III. Evaluierung	259
9. Konzeptionelle Einordnung	263
10. Überblick über die empirische Untersuchung	287
11. Evaluierung der Verwendbarkeit des Werkzeugs	305

Inhaltsübersicht

I 2. Evaluierung der erstellten Modelle	329
I 3. Evaluierung der durchgeführten Articulation Work	331
I 4. Schlussbetrachtungen	335
A. Literatur zum Themengebiet Articulation Work	III
 Anhänge	 III
 Verzeichnisse	 XXI
Abbildungsverzeichnis	XXI
Tabellenverzeichnis	XXII
Stichwortverzeichnis	XXIII
Abkürzungsverzeichnis	XXIII
Bildquellen	XXVI
Publikationen im Kontext dieser Arbeit	XXVIII
Literaturverzeichnis	XXXI

Inhaltsverzeichnis

I. Einführung	1
1.1. Forschungsfragen	3
1.2. Aufbau der Arbeit	3
I. Grundlagen	5
2. Articulation Work	9
2.1. Begriffsbestimmung	9
2.2. Ausprägungen von Articulation Work	13
2.2.1. Unterscheidung nach Fjuk, Smørdal und Nurminen	16
2.2.2. Unterscheidung nach Hampson und Junor	19
2.2.3. Unterscheidung nach Færgemann et al.	21
2.2.4. Zusammenfassung	21
2.3. Abzustimmende Arbeitsaspekte	29
2.4. Unterstützung von Articulation Work	29
2.4.1. Vorgehen zur detaillierten Betrachtung	31
2.4.2. Modeling Articulation Work in Software Engineering Processes	33
2.4.3. Taking CSCW seriously: Supporting Articulation Work	35
2.4.4. Supporting articulation work using software configuration management systems	38
2.4.5. Coordination Mechanisms: Towards a Conceptual Foundation of CSCW Systems Design	40
2.4.6. Taking Articulation Work Seriously: An Activity Theoretical Approach	43
2.4.7. TeamSpace: an environment for team articulation work and virtual meetings	45
2.4.8. Supporting different dimensions of adaptability in workflow modeling	46
2.4.9. Mundane knowledge management and microlevel organizational learning: An ethological approach	48

Inhaltsverzeichnis

2.4.10. Modelling Cooperative Work: Chances and Risks of Structuring	50
2.4.11. Recursive Articulation Work in Ariadne: The Alignment of Meanings	52
2.4.12. Combining Communication and Coordination Toward Articulation of Collaborative Activities	54
2.4.13. Interactive Process Models	56
2.4.14. Torres, a Conceptual Framework for Articulation Work across Boundaries	58
2.4.15. Gegenüberstellung und Zusammenfassung	59
2.5. Thought processes und Articulation Work	61
3. Mentale Modelle	63
3.1. Articulation Work und mentale Modelle	63
3.2. Begriffsbestimmung	64
3.3. Bildung und Veränderung mentaler Modelle	66
3.4. Externalisierung mentaler Modelle	71
3.4.1. Methode des lauten Denkens	73
3.4.2. Strukturlegetechniken	76
3.4.3. Concept Mapping	80
3.5. Externalisierung im Rahmen von Articulation Work	83
3.5.1. Durchführungsrahmen	84
3.5.2. Vorgehen	84
3.6. Fazit	86
II. Unterstützung	87
4. Anforderungen an ein Werkzeug	91
5. Grundlagen der Implementierung	93
5.1. Historische Entwicklung von Tangible Interfaces	94
5.2. Konzeptualisierung und Klassifikation von Tangible Interfaces	95
5.2.1. Bricks	97
5.2.2. Graspable User Interfaces	99
5.2.3. Tangible Bits	100
5.2.4. Containers, Tokens und Tools	102
5.2.5. Tangible Objects Meaning	105
5.2.6. Das MCRpd Interaktions-Modell	106
5.2.7. Tokens und Constraints nach Ullmer	109
5.2.8. Degree of Coherence	III

5.2.9.	Tokens und Constraints nach Shaer et al.	114
5.2.10.	Kategorien von TUI-Anwendungen	115
5.2.11.	Taxonomie für Tangible User Interfaces	116
5.2.12.	Tangible Bits: Beyond Pixels	119
5.2.13.	Zusammenfassung	123
5.3.	Tangible Interfaces in kooperativer Verwendung	129
5.4.	Tabletop Interfaces	129
5.4.1.	Historische Entwicklung	129
5.5.	Tangible Interfaces zur Erstellung diagrammatische Modelle	129
5.5.1.	Aktuelle verwandte Ansätze	130
5.6.	Fazit	130
6. Eingabe und Interpretation		131
6.1.	Möglichkeiten zur Erfassung von Benutzerinteraktion	131
6.1.1.	In Frage kommende technologische Ansätze	132
6.1.2.	In Frage kommende Frameworks	139
6.1.3.	Technologieentscheidung	148
6.2.	Konzeption und Umsetzung der Hardwarekomponenten	154
6.2.1.	Überblick	155
6.2.2.	Tokens und Input-Werkzeuge	156
6.2.3.	Eingabe auf der Tischoberfläche	166
6.3.	Benutzerinteraktion mit dem Werkzeug	168
6.3.1.	Hinzufügen und Verändern von Modellelementen	169
6.3.2.	Benennen von Modellelementen	170
6.3.3.	Verbinden von Modellelementen	171
6.3.4.	Löschen von Elementen und Verbindungen	173
6.3.5.	Einbettung von Zusatzinformation	173
6.3.6.	Kontrolle der Modellierungshistorie	175
6.4.	Erfassung der Benutzerinteraktion durch Software	177
6.4.1.	Interpretation der Rohdaten	177
6.4.2.	Stabilisierung der Erkennungsleistung	180
6.4.3.	Erkennung von Markierungen und Verbindungen	185
6.4.4.	Erkennung von geöffneten Tokens	187
6.4.5.	Benennung von Modellelementen	190
6.4.6.	Festlegung der Bedeutung von Modellelementen	192
6.4.7.	Tracking des Modellzustandes	192
6.4.8.	Verteilung des Modellzustandes	195
6.5.	Zusammenfassung	195
7. Ausgabe		197
7.1.	Auszugebende Information	197

7.2.	Technologische Grundlage der Ausgabe	198
7.2.1.	Ansätze zur kohärenten Ausgabe	199
7.2.2.	Ansätze zur entkoppelten Ausgabe	202
7.2.3.	Technologie-Entscheidung	204
7.2.4.	Frameworks zur Ausgabe	206
7.3.	Ausgabe von Information	209
7.3.1.	Konzept	210
7.3.2.	Architektur	212
7.3.3.	Ausgabe von Information zum Modell	212
7.3.4.	Ausgabe zur Kontrolle des Systems	217
7.4.	Umsetzung der Ausgabe mit Software	222
7.4.1.	Ausgabe des Modellzustands	225
7.4.2.	Ausgabe der Modellierungshistorie	226
7.4.3.	Umsetzung der Wiederherstellungsunterstützung	227
7.5.	Zusammenfassung	228
8.	Persistierung	231
8.1.	Möglichkeiten der Persistenzsicherung	231
8.2.	Topic Maps	232
8.2.1.	Topics, Subjects, Topic Names und Variants	233
8.2.2.	Associations und Roles	236
8.2.3.	Occurrences und Datatypes	237
8.2.4.	Metamodellierung in Topic Maps	237
8.2.5.	Statements und Scopes	241
8.2.6.	Reification	242
8.2.7.	Merging	243
8.3.	Abbildung von Modellen auf Topic Maps	243
8.3.1.	Grundlegende Abbildung	244
8.3.2.	Abbildung des Metamodells	245
8.3.3.	Abgrenzung von Submodellen	248
8.3.4.	Flexibilisierung der Abbildung	250
8.4.	Technische Umsetzung der Persistierung von Modellen	251
8.4.1.	Topic Map Engine	251
8.4.2.	Dynamische Metamodelle	254
8.4.3.	Persistierung	254
8.5.	Export graphischer Repräsentationen	254
8.5.1.	Ausgabeformen	254
8.5.2.	Technische Umsetzung des graphischen Exports	258
8.6.	Zusammenfassung	258

III. Evaluierung	259
9. Konzeptionelle Einordnung	263
9.1. Einordnung in den Bricks-Designraum	263
9.1.1. Abbildung	263
9.1.2. Bewertung	265
9.2. Bestimmung der Eigenschaften des Graspable User Interfaces Ansatz	266
9.2.1. Abbildung	266
9.2.2. Bewertung	267
9.3. Betrachtung im Lichte des Tangible Bits Ansatzes	267
9.3.1. Abbildung	267
9.3.2. Bewertung	268
9.4. Einordnung in das Ordnungssystem von Holmquist et al.	268
9.4.1. Abbildung	268
9.4.2. Bewertung	269
9.5. Einordnung in das Object-Meaning-Kontinuum	270
9.5.1. Abbildung	270
9.5.2. Bewertung	271
9.6. Betrachtung im Lichte des MCRpd-Modells	271
9.6.1. Abbildung	272
9.6.2. Bewertung	273
9.7. Einordnung in den Tokens+Constraints Kontext	273
9.7.1. Einordnung	273
9.7.2. Bewertung	275
9.8. Einordnung in das Framework nach Koleva et al.	275
9.8.1. Abbildung	275
9.8.2. Bewertung	277
9.9. Spezifikation des TAC-Schemas nach Shaer et al.	277
9.9.1. Abbildung	278
9.9.2. Bewertung	280
9.10. Einordnung in die Kategorien von TUI-Anwendungen	280
9.10.1. Abbildung	281
9.10.2. Bewertung	281
9.11. Einordnung in die Taxonomie von Fishkin	281
9.11.1. Abbildung	281
9.11.2. Bewertung	284
9.12. Betrachtung im Lichte der Retrospektive von Ishii	284
9.12.1. Einordnung	285
9.12.2. Bewertung	285
9.13. Zusammenfassung	286
9.13.1. Eignung der konzeptionellen Ansätze zur Beschreibung . . .	286

9.13.2. Verbesserungspotential für das Werkzeug	286
10. Überblick über die empirische Untersuchung	287
10.1. Untersuchungsaspekte	287
10.1.1. Evaluierung des Werkzeugs	288
10.1.2. Evaluierung der Modellrepräsentationen	289
10.1.3. Evaluierung der Articulation Work	290
10.2. Globales Untersuchungsdesign	291
10.2.1. Block 1: Technische Evaluierung	292
10.2.2. Block 2: Aushandlung von Zusammenarbeit 1	293
10.2.3. Block 3: Concept Mapping 1	295
10.2.4. Block 4: Aushandlung von Zusammenarbeit 2	298
10.2.5. Block 5: Concept Mapping 2	300
10.3. Eingesetzte Werkzeuge und Verfahren	302
10.3.1. Werkzeuge	302
10.3.2. Korrelationstests	302
10.3.3. Signifikanztests	302
10.4. Zusammenfassung	302
11. Evaluierung der Verwendbarkeit des Werkzeugs	305
11.1. Hypothesen	305
11.1.1. Konzeptionell begründete Hypothesen	305
11.1.2. Explorativ gebildete Hypothesen	308
11.2. Untersuchungsdesign und Durchführung	309
11.2.1. Operationalisierung	310
11.2.2. Datenbasis	316
11.2.3. Durchführung	316
11.3. Ergebnisse	317
11.3.1. Repräsentation diagrammatischer Modelle	317
11.3.2. Kollaboratives Arbeiten	319
11.3.3. Herstellung von Verbindern	321
11.3.4. Verwendung des Löschtokens	324
12. Evaluierung der erstellten Modelle	329
12.1. Hypothesen	329
12.1.1. Konzeptuell begründete Hypothesen	329
12.1.2. Explorativ gebildete Hypothesen	329
12.2. Untersuchungsdesign und Durchführung	330
12.2.1. Grundlagen	330
12.3. Ergebnisse	330
12.3.1. Connectedness	330

I 3. Evaluierung der durchgeführten Articulation Work	331
I 3.1. Hypothesen	331
I 3.1.1. Konzeptuell begründete Hypothesen	331
I 3.1.2. Explorativ gebildete Hypothesen	331
I 3.2. Untersuchungsdesign und Durchführung	331
I 3.3. Ergebnisse	331
I 4. Schlussbetrachtungen	335
I 4.1. Anwendungsszenarien	335
I 4.1.1. Problembeschreibung und Arbeitsabstimmung	335
I 4.1.2. Concept Mapping	335
I 4.1.3. Strukturaufstellung und Manipulation	335
A. Literatur zum Themengebiet Articulation Work	III
A.1. Literaturquellen	III
A.2. Relevante Literatur	IV
Anhänge	III
Verzeichnisse	XXI
Abbildungsverzeichnis	XXI
Tabellenverzeichnis	XXII
Stichwortverzeichnis	XXIII
Abkürzungsverzeichnis	XXIII
Bildquellen	XXVI
Publikationen im Kontext dieser Arbeit	XXVIII
Literaturverzeichnis	XXXI

I. Einführung

„How people work is one of the best kept secrets in America.“ (Wellman, D. zitiert nach Suchman (1995)).

Diese Aussage David Wellmans diente Lucy Suchman in den 90er-Jahren des 20. Jahrhunderts als Motivator für ihre Forschung über die Natur menschlicher Arbeit im Computerzeitalter und deren Unterstützung durch neue Technologien. Ihre Arbeiten und viele andere (etwa Schmidt und Bannon (1992) oder Sachs (1995)) argumentieren für eine stärkere Berücksichtigung der Rolle des Menschen im Arbeitsprozess.

Wellman spielt mit diesem Zitat auf die oft auftretende Diskrepanz zwischen der (niedergeschriebenen) Definition eines Arbeitsablaufs und dessen tatsächlicher Umsetzung in der konkreten betrieblichen Umgebung an.

Der Druck in der heutigen Geschäftswelt, bestimmte Qualitätskriterien garantiert erfüllen zu können, hat zu einer beinahe flächendeckenden Verbreitung von Qualitätszertifizierungen geführt. Die bekanntesten Vertreter dieser Zertifizierungen sind wohl die Standards aus der Familie der ISO 9000 Normen ISO (2005). In der ISO 9001-Norm ISO (2000), in der die Anforderungen an Qualitätsmanagement-Systeme definiert sind, ist festgeschrieben, dass eine prozessorientierte Organisation eine der Voraussetzungen für erfolgreiches Qualitätsmanagement ist. Ein wesentliches Merkmal einer prozessorientierten Organisation ist, dass ihre organisationalen Prozesse — also ihre Arbeitsorganisation und -abläufe — bekannt, benannt und definiert sind.

Die Unterschiede zwischen festgeschriebenen und gelebten Arbeitsabläufen wurde schon 1978 von Argyris und Schön Argyris und Schön (1978) beschrieben. Mit der Unterscheidung zwischen „*espoused theories*“ (= die offiziell veröffentlichten Theorien über Arbeit) und den „*theories-in-use*“ (= die tatsächlich handlungsleitenden Theorien) wurden dieser Gegensatz auch explizit benannt. Sachs (1995) beschreibt das gleiche Phänomen und unterscheidet zwischen dem „*explicit organisational view*“ und dem „*tacit organisational view*“ auf Arbeit. Sie beschreibt damit einerseits eine explizit formulierte und statische Sicht auf Arbeit und andererseits eine informelle, im Fluss befindliche und zum Zeitpunkt der Betrachtung nirgendwo niedergeschriebene Sicht auf Arbeit. Letztere kann lediglich aus der Analyse der tatsächlichen Arbeitsabläufe gewonnen werden kann, die den Tätigkeiten zugrunde liegenden An-

1. Einführung

nahmen („theories-in-use“ Argyris und Schön (1978)) sowie deren vom Arbeitenden konkret wahrgenommene organisationale Rahmenbedingungen bleiben allerdings verborgen — die Frage nach dem „Warum?“, die die Form des konkreten Arbeitsablaufs motiviert, kann nicht unmittelbar beantwortet werden.

Wie Sachs verdeutlichen auch Wellman sowie Argyris und Schön, dass das zweitgenannte Verständnis von Arbeit nicht explizit niedergeschrieben und formal definiert ist — in seinem Wesen also „unbekannt“ ist. Wenn „Arbeit“ oder die ihr zugrunde liegenden Annahmen unbekannt sind, kann eine Veränderung ihrer selbst oder der Umgebung, in der sie durchgeführt wird, zu schwerwiegenden Problemen führen (wie z.B. von Nonaka und Takeuchi (1995), von Krogh et al. (2000) oder Gerson und Star (1986) beschrieben). Der Begriff „Veränderung“ deckt dabei nicht nur tiefgreifende organisatorische Änderungen im Unternehmen ab, sondern durchaus auch „marginale“ Änderungen wie z.B. die Einstellung neuer Mitarbeiter oder dem Einsatz eines neuen Werkzeugs Olesen und Markussen (2003). Daraus kann man schließen, dass potentiell „problematische“ Situationen häufig auftreten können.

„Problematische“ Situationen sind dabei all jene Situationen, in denen die Zielerreichung erschwert wird, weil die dazu notwendigen Schritte entweder unklar sind oder nicht operationalisiert werden können. Im Kontext der obigen Aussage bedeutet dies, dass durch eine organisationale Veränderung neue Arbeitsschritte notwendig werden bzw. die bisherigen nicht mehr funktionieren oder angemessen sind. Beim Auftreten einer derartigen Veränderung ist daher eine erneute Planung bzw. Abstimmung der zur Zielerreichung notwendigen Arbeitsschritte notwendig. Fujimura Fujimura (1987) unterscheidet in diesem Sinne zwischen zwei Formen von Arbeit — der „Produktion“ („*production*“) und der „Artikulation“ („*articulation*“), wobei letztere alle Tätigkeiten umfasst, die die Umsetzung bzw. Aufrechterhaltung der „Produktion“ ermöglichen.

„Artikulation“ ist ein integraler Bestandteil von Arbeit Strauss (1985). Mit der Komplexität der „Produktion“ steigt auch der Aufwand der dazu notwendigen „Artikulation“ an Strauss (1988). Die Komplexität steigt hier mit der Anzahl der benötigten Arbeitsschritte, den dazu benötigten Kompetenzen und der Anzahl der involvierten Personen. Nicht bekannte, falsche oder zurückgehaltene Information über die „Produktion“ erschweren die „Artikulation“ oder machen sie unmöglich Fujimura (1987). Dies hat jedoch nicht nur negative Auswirkungen auf die „Produktion“ sondern verhindert auch eine tiefergehende Beschäftigung mit der aktuellen Arbeitspraxis und eine potentielle Verbesserung derselben Argyris und Schön (1978). Erfolgreiche Artikulation ist damit nicht nur eine Voraussetzung für eine funktionierende Produktion sondern auch ein Enabler für organisationale Veränderungen im Sinne eines organisationalen Lernprozesses (z.B. Kim (1993) oder Firestone und McElroy (2003)).

Eine methodische und technische Unterstützung kann Artikulation ermöglichen oder deren erfolgreichen Ablauf erleichtern (Schmidt und Bannon (1992), Simone et al. (1999), Jørgensen (2004), Baker und Millerand (2007)).

In der Arbeit sind die methodischen und technischen Möglichkeiten zur Ermöglichung und Unterstützung von Articulation Work zu ergründen, die gewonnenen Erkenntnissen in einem Werkzeug umzusetzen und dessen Auswirkungen auf die Interaktion zur Verbesserung der Production Work zu bewerten.

I.1. Forschungsfragen

1. Wie kann Articulation Work ermöglicht und unterstützt werden?
 - a) Durch welche Aktivitäten zeigt sich Articulation Work im Arbeitsprozess?
 - b) Welche Rahmenbedingungen ermöglichen bzw. begünstigen Articulation Work?
 - c) Wie können die in 1.1. identifizierten Aktivitäten unterstützt werden?
 - d) Wie können die in 1.2. identifizierten Rahmenbedingungen und die in 1.3. identifizierten Anforderungen in einer Methodik umgesetzt werden?
2. Was muss ein Werkzeug zur Unterstützung von Articulation Work leisten?
 - a) Welche Anforderungen an ein Werkzeug ergeben sich aus der in 1.4. entwickelten Methodik?
 - b) Wie können diese Anforderungen technologisch umgesetzt werden?
3. Inwiefern unterstützt das entwickelte Werkzeug die Durchführung von Articulation Work?
 - a) Wie kann die Unterstützungsleistung bewertet werden?
 - b) Welche Auswirkungen auf die Interaktion hat die Anwendung von Methodik und Werkzeug?

I.2. Aufbau der Arbeit

Teil I.

Grundlagen

Einleitung

Dieser Teil stellt die dieser Arbeit zugrundeliegenden Konzepte und deren Auswirkungen auf die Erreichung der globalen Zielsetzung vor. Ziel dieses Teils ist es, diese Konzepte umfassend darzulegen und in der existierenden Literatur Möglichkeiten bzw. Ansatzpunkte zur Unterstützung expliziter Articulation Work zu identifizieren.

AW motivieren

Mentale Modelle motivieren

Externalisierung motivieren

Aufbau des Teils beschreiben

2. Articulation Work

In diesem Kapitel wird das Konzept „Articulation Work“ dargestellt und in den Kontext von menschlicher Arbeit gestellt. Im ersten Teil des Kapitels wird auf die historische Entwicklung des Begriffs „Articulation Work“ und die unterschiedlichen Herangehensweise zu dessen Verständnis eingegangen. Der zweite Teil des Kapitels widmet sich den Aktivitäten, die „Articulation Work“ ausmachen, den Merkmalen, an denen sich gute „Articulation Work“ zeigt, sowie den Möglichkeiten der Unterstützung von „Articulation Work“ durch organisationale und technische Maßnahmen.

2.1. Begriffsbestimmung

Das Konzept der "Articulation Work" wurde als Erklärungsmodell für eine bestimmte Art von menschlicher Arbeit Mitte der 1980er Jahre von Strauss (1985) eingeführt. Neben Strauss (1985) tragen auch die Arbeiten von Gerson und Star (1986) und Fujimura (1987) wesentlich zur Begriffsbestimmung und Konzeptbildung bei. Die vorhandene Literatur, die Bezug auf „Articulation Work“ nimmt, referenziert im Wesentlichen auf diese drei Arbeiten bzw. eine dieser drei Arbeiten. Der Kontext, in dem die Entwicklung der im folgenden vorgestellten Konzepte erfolgte, war die komplexe, von viel Interaktion an zahlreichen Schnittstellen geprägte Arbeit in Krankenhäusern (Strauss, 1985), in der Wissenschaft (Fujimura, 1987) und in Versicherungsunternehmen (Gerson und Star, 1986), die die jeweiligen Autoren in mehreren Fallstudien untersuchten.

Um in der Folge einen einheitlichen Begriffsraum aufspannen zu können, ist vorab der Begriff „Arbeit“ zu klären. Die eben genannten Autoren führen keine explizite Definition an, weshalb hier auf eine Definition zurückgegriffen wird, die im Kontext der folgenden Ausführungen zur „inneren“ Struktur von Arbeit nach „außen“ hinreichend umfassend ist¹. (Semmer und Udris, 2004) definieren vor dem Hintergrund der Organisationspsychologie „Arbeit“ wie folgt: „*Arbeit ist zielgerichtete*

¹Auf eine umfassende Literaturstudie und die Entwicklung eines darauf aufbauenden „Arbeits“-Begriffs wurde hier verzichtet, da dies über den Betrachtungsbereich und Anspruch dieser Arbeit hinausgeht

2. Articulation Work

menschliche Tätigkeit zum Zwecke der Transformation und Aneignung der Umwelt aufgrund selbst- oder fremddefinierter Aufgaben, mit gesellschaftlicher, materieller oder ideeller Bewertung, zur Realisierung oder Weiterentwicklung individueller oder kollektiver Bedürfnisse, Ansprüche und Kompetenzen.“. Arbeit ist also ein menschliches Phänomen, Träger von Arbeit sind immer Menschen. Arbeit definiert sich außerdem durch ihre Zielgerichtetetheit und findet immer in Interaktion mit der Umwelt statt. Die Ziele, auf die Arbeit ausgerichtet ist, leiten sich aus Aufgaben ab, die sich Menschen selbst setzen können oder die ihnen vorgegeben werden. Diese Aufgaben dienen der Erreichung von individuellen oder kollektiven Bedürfnissen und Ansprüchen bzw. der (Weiter-)Entwicklung von Kompetenzen. Die Bewertung der Zielerreichung muss nicht unbedingt aus materieller Perspektive erfolgen sondern kann auch ideell oder gesellschaftlich begründet sein. In dieser Arbeit wird der Begriff „Arbeit“ vor allem auch im organisationalen Kontext gesehen. Ein wesentlicher Aspekt, der hierbei zu berücksichtigen ist, ist die Arbeitsteilung, also die koordinierte Tätigkeit mehrerer Individuen um ein gemeinsames Ziel zu erreichen. Dies stellt die obige Definition nicht in Frage (Schmidt, 1994), erweitert jedoch den Betrachtungsbereich explizit auch auf Arbeit, die gemeinschaftlich durchgeführt wird².

„Articulation Work“ ist jener Anteil der gesamten durchgeführten Arbeit, der der Abstimmung mit anderen Individuen dient. Diese Abstimmung ist notwendig, um das eigentliche Arbeitsziel erreichen zu können. Arbeit wird von den oben angeführten Autoren als inhärent kooperativer Prozess gesehen, der immer auf Interaktion mit anderen Menschen basiert bzw. diese bedingt (Strauss formuliert diese Annahme in Bezugnahme auf Hughes (1971) prägnant mit der Aussage „*work rests ultimately on interaction*“). Diese Annahme erscheint insofern als zulässig, als dass selbst Arbeitsabläufe, die selbst keine Kooperation mit anderen Menschen mit sich bringen, zumindest auf den Ergebnissen anderer Arbeitsabläufe aufbauen oder als Grundlage weiterer Arbeitsabläufe dienen. Interaktion tritt also in jedem Arbeitsprozess zumindest zu Beginn und am Ende in unmittelbarer oder mittelbarer³ Form auf. Diese Annahme stützt auch Schmidt (1994), der darauf hinweist, dass individuelle Arbeit und kooperative Arbeit oft nicht klar abgrenzbar sind bzw. dynamisch ineinander übergehen⁴

² „[...] work is an individual phenomenon in so far as labor power happens to be tied to individuals and cannot be separated from the individuals. That is, a cooperative work process, is performed by individuals with individual interests and motives.“ (Schmidt, 1994, S. 353)

³ Unter „mittelbar“ ist hier Interaktion zu verstehen, die nicht im direkten Kontakt zwischen Individuen abläuft sondern lediglich indirekt durch die Ergebnisse eines Arbeitsprozesses (Materialien, Dokumente, ...) vermittelt wird.

⁴ „Cooperative work and individual work should not be conceived of as different work domains. In daily work practice, cooperative and individual activities are inextricably interwoven. [...] More than that, the boundary between individual and cooperative work is dynamic in the sense that people enter into

Jener Teil von Arbeit, der der eigentlichen Zielerreichung dient, wird im hier vorgestellten Erklärungsmodell als „Production Work“ bezeichnet (Fujimura, 1987). „Production Work“ ist komplementär zu „Articulation Work“ zu sehen und umfasst alle Aktivitäten, die der „Wertschöpfung“ im wörtlichen Sinn dienen. „Production Work“ sind also alle Tätigkeiten, die mit der Schaffung jener Werte (oder Ergebnisse) befasst sind, die durch den Arbeitsablauf erreicht werden sollen.

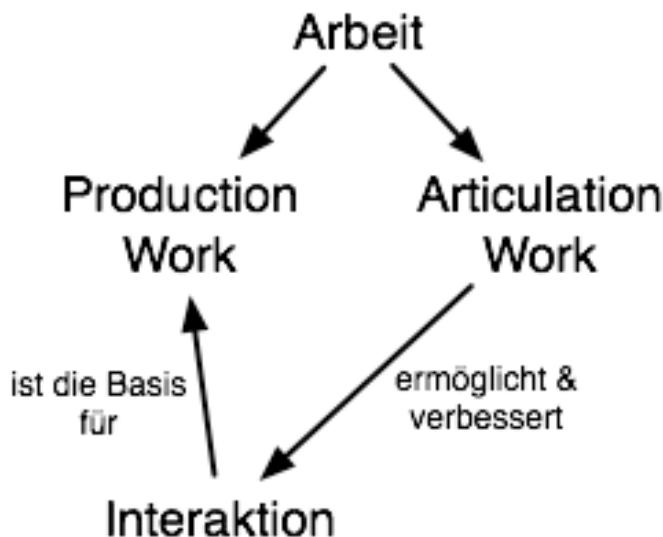


Abbildung 2.1.: Struktur von Arbeitsabläufen

Teile eines Arbeitsablaufs dienen also der Zielerreichung an sich („Production Work“). Andere Teile dienen der Abstimmung zwischen den involvierten Akteuren, um ein gemeinsames Verständnis über die jeweiligen Schnittstellen – also die Berührungspunkte zwischen den Tätigkeiten – zu entwickeln (siehe Abbildung 2.1). Diese Entwicklung eines gemeinsamen Verständnisses bzw. diese „Koordination“ ist kritisch für den Erfolg von kooperativer Arbeit (Strauss, 1993) und wird als „Articulation Work“ bezeichnet.⁵

„Articulation Work“ ist also ein Enabler für funktionierende Kommunikation und Zusammenarbeit im eigentlichen Arbeitsablauf. Zentral ist dabei vor allem die gegenseitigen Offenlegung der Annahmen aller beteiligten Personen, die den individuellen Arbeitsbeiträgen zugrunde liegen⁶.

cooperative work relations and leave them according to the requirements of the current situation and the technical and human resources at hand.“ (Schmidt, 1994, S. 352)

⁵ „Without an understanding of articulation, the gap between requirements and the actual work process in the office will remain inaccessible to analysis. That is, it will be possible to describe tasks in an idealized form but not to describe actual situations.“ (Gerson und Star, 1986)

2. Articulation Work

„Articulation Work“ ist keine Tätigkeit, die zu einem bestimmten Zeitpunkt im Arbeitsprozess durchgeführt wird und dann als abgeschlossen betrachtet werden kann⁷. Vielmehr wird „Articulation Work“ immer auch begleitend zur eigentlichen produktiven Arbeit durchgeführt und umfasst neben planenden und koordinierenden Tätigkeiten auch das Erkennen von Fehlentwicklungen bzw. von Situationen, in denen eine erneute Koordination notwendig ist⁸ (siehe Abbildung 2.2).

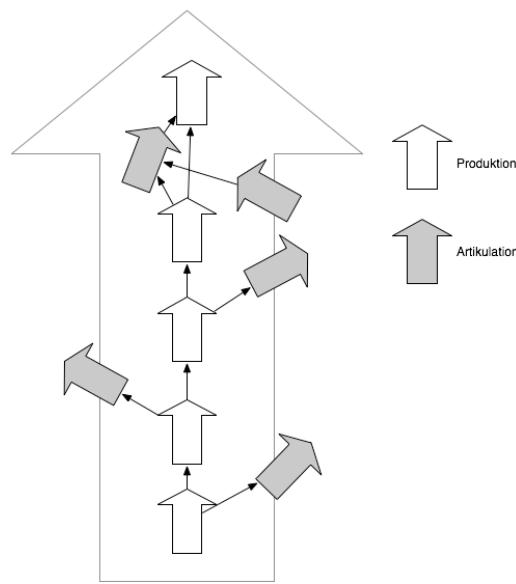


Abbildung 2.2.: Konzeptualisierung von „Arbeit“ nach (Strauss, 1985) und (Fujimura, 1987)

Der Begriff „Articulation Work“ ist im Englischen zweideutig und von Strauss auch bewusst so gewählt. Einerseits wird damit ausgedrückt, dass *Arbeit* („Work“) artikuliert wird, andererseits zeigt der Begriff, dass die *Artikulation* selbst ebenfalls Arbeit ist (also Zeit und Ressourcen in Anspruch nimmt) und auch also solche wertgeschätzt werden muss (Fujimura, 1987). Gleichzeitig kennzeichnet er auch die rekursive Natur von »Articulation Work«, die somit jederzeit selbst Gegenstand von »Articulation Work« werden kann (Star und Strauss, 1999). „Articulation Work“ ist kein klar abgegrenztes und strukturiertes Konzept – sie tritt je nach Arbeitssituation in unterschiedlichen Spielarten auf. Die Unterscheidung dieser Arten von „Articula-

⁶ „Reconciling incommensurate assumptions and procedures in the absence of enforceable standards is the essence of articulation.“(Gerson und Star, 1986, S. 266)

⁷ „The articulation work involves the pre-articulation of the tasks, their management and post-articulation.“(Raposo et al., 2004, S. 121)

⁸ „Articulation consists of all the tasks involved in assembling, scheduling, monitoring, and coordinating all of the steps necessary to complete a production task.“(Gerson und Star, 1986, S. 266)

tion Work“ ist für die Unterstützung derselben relevant und wird daher im folgenden Abschnitt genauer betrachtet.

2.2. Ausprägungen von Articulation Work

Wie bereits von Gerson und Star (1986) angeführt, argumentiert auch Strauss, dass Artikulation immer passieren muss (und passiert), wo Menschen zusammenarbeiten, um zu vermeiden, dass unbekannte Aspekte Probleme bei der Durchführung der Arbeit verursachen (Strauss, 1988). „Articulation Work“ ist kein revolutionäres Konzept, sondern fasst Tätigkeiten unter einem Begriff zusammen, die seit jeher Teil jeder Zusammenarbeit zwischen Menschen sind. Grundsätzlich geht Strauss davon aus, dass „Articulation Work“ immer abläuft, egal wie einfach oder kompliziert, wie eingespielt oder neuartig eine (Zusammen-)Arbeit ist (Strauss, 1988). Sehr wohl existieren jedoch Unterschiede in der Qualität der Arbeit, die sich auf die Form der Artikulation auswirken, die zu deren Abstimmung notwendig ist: „*A useful fundamental distinction between classes of interaction is between the routine and the problematic. Problematic interactions involve 'thought', or when more than one interactant is involved then also 'discussion'.*“ (Strauss, 1993). Dieses Zitat zeigt im Übrigen auch, dass „Interaction“ im Sinne von Strauss nicht unbedingt ein kollektives Phänomen ist, sondern auch individuell (im Bezug auf die (unbelebte) Umgebung) auftreten kann.

Je komplexer („problematic“) eine Interaktion ist, desto notwendiger wird laut Strauss eine explizite Beschäftigung mit dem Vorgang der Artikulation. Bei einfachen, eingespielten („routine“) Interaktionen bleibt die Artikulation zumeist implizit, verborgen und informell (Hampson und Junor, 2005) (entsprechend der „Sozialisation“ im aus der Domäne der Wissensgenerierung und -teilung stammenden SECI-Zyklus (Nonaka und Takeuchi, 1995)). Ein grundlegendes Problem, dass Artikulation für jeden noch so einfach erscheinend Arbeitsvorgang potentiell relevant macht, spricht Strauss mit den Worten von Hughes unmittelbar nach der Definition von „problematic interaction“ an: „*[O]ne man's routine of work is made up of the emergencies of other people*“ (Hughes, 1971) zitiert nach (Strauss, 1993).

„Articulation Work“ tritt also in zwei Qualitäten auf. Ist der Bedarf zur Abstimmung bekannt und werden Tätigkeiten zur Abdeckung dieses Bedarf bewusst durchgeführt, so spricht man von *expliziter* „Articulation Work“ (Strauss, 1988) (Fjuk et al., 1997). Die Abstimmung von Tätigkeiten, die ständig während der Zusammenarbeit unbewusst ausgeführt wird, bezeichnet man als *implizite* „Articulation Work“⁹.

⁹*The explicit articulation is thus connected to the planning and decisions regarding the salient dimensions of work – who, what, when, how – while implicit articulation is invaluable when carrying out activities in situated circumstances, in order to handle contingencies.*(Fjuk et al., 1997, S.5)

2. Articulation Work

Letztgenannte Art ist es auch, die von den Arbeitenden „automatisch“ zur Anwendung gebracht wird, sobald Änderungen in der Arbeitsumgebung oder Probleme auftreten (Strauss, 1988). Implizite „Articulation Work“ stößt aber an ihre Grenzen, wenn die Arbeitssituation als „problematisch“ (Strauss, 1988) oder „komplex“ (Schmidt, 1990, S. 23f) wahrgenommen wird. Es wird dann notwendig, dezidierte Abstimmungs-Aktivitäten anzustoßen, also explizite „Articulation Work“ durchzuführen.

Diese Abstimmungs-Aktivitäten können konkret wiederum unterschiedliche Ausprägungen annehmen (Gasser, 1986):

Fitting (bzw. „Accommodation“ (Bendifallah und Scacchi, 1987)) Tätigkeiten zur Planung bzw. Anpassung der Arbeitspraxis an gegebene bzw. veränderte Umweltbedingungen.

Augmenting (bzw. „Negotiation of additional [maintainance] activities“ (Bendifallah und Scacchi, 1987)) Planung von zusätzlichen kurz- oder mittelfristigen Tätigkeiten, um das Auftreten von erkannten Problemen zu verhindern.

Working around Entwicklung von Strategien zur Vermeidung des Auftretens von Situationen, in denen Probleme auftreten, ohne deren Ursache zu beseitigen.

In einer späteren Arbeit (Corbin und Strauss, 1993) geht Strauss auf den bislang nicht betrachteten temporalen Aspekte des Auftretens von expliziter „Articulation Work“ ein und unterscheidet dabei zwischen „Working out Original Arrangements“ (also der erstmaligen Vereinbarung der Modalitäten einer Zusammenarbeit) und „Reworking Arrangements“ (also der Veränderung von getroffenen Vereinbarungen, die durch Änderungen im Arbeitskontext nicht mehr angewandt werden können). Während der Ausgangspunkt in diesen beiden Fällen unterschiedlich ist, ist doch die Zielsetzung die gleiche – Ziel ist es, die individuellen Standpunkte und Sichtweisen („stances“) soweit abzulegen, dass eine Zusammenarbeit möglich ist. Diese Aktivitäten unterscheiden sich insofern von der laufend im Arbeitsablauf durchgeführten „Articulation Work“, als dass sie die Individuen dazu zwingen, aus dem Arbeitssystem herauszusteigen und dieses als Gesamtes zum Gegenstand der „Articulation Work“ zu machen. Dies umfasst auch die Aushandlung der Modalitäten der im Arbeitsablauf durchgeführten „Articulation Work“, wodurch deren rekursive Natur (Star und Strauss, 1999) deutlich wird. Sarini und Simone (2002a) beschäftigen sich explizit mit „rekursiver Articulation Work“ und führen als wesentliche Ziele die Entwicklung eines gemeinsamen Verständnisses über die Arbeitsdomäne („alignment of meaning“) sowie die Abstimmung des Arbeitsablaufs selbst („alignment of procedures“) an. Diesem breiten Verständnis von „Articulation Work“ schließen sich auch Baker und Millerand (2007) aufgrund empirischer Erkenntnisse an.

Die Schritte, die im Rahmen von der Erarbeitung von „arrangements“ durchgeführt werden müssen, geben Corbin und Strauss (1993) wie folgt an (und beziehen sich dabei im Wesentlichen auf „alignment of procedures“):

1. Jedes beteiligte Individuum definiert für sich, welche Aspekte der Arbeit ver einbart werden müssen und legt seine Position dazu („stance“) fest. Mögliche Aspekte betreffen die durchzuführenden Schritte, mögliche Verantwortlichkeiten und benötigte Ressourcen¹⁰.
2. Die beteiligten Individuen interpretieren die explizierten Standpunkte und Sichtweisen („stances“) der jeweils anderen.
3. Basierend auf diese Interpretation passt das Individuum seine Standpunkte und Sichtweisen an oder behält diese bei und verändert ggf. seine Verhandlungsstrategie.
4. Die Schritte 1-3 werden wiederholt, bis eine für alle Individuen akzeptable Vereinbarung erreicht ist.

Diese zeitliche Unterscheidung lässt jedoch die häufiger auftretende „Articulation Work“ im Arbeitsprozess (ohne aus dem Arbeitssystem herauszusteigen) außen vor. Während eines Arbeitsablaufs kann neben „Reworking Arrangements“ (das als Eskalationsstufe bei Vereinbarungen zu sehen ist, auf deren Basis die Arbeit nicht mehr fortgesetzt werden kann und gleichbedeutend mit den Tätigkeiten „augmenting“ und „working around“ in (Gasser, 1986) ist) auch das Lösen von problematischen Situationen im Rahmen der aktuellen Vereinbarungen („resolving contingencies“ (Gerson und Star, 1986) bzw. „fitting“ (Gasser, 1986)) im Rahmen von „Articulation Work“ durchgeführt werden.

Neben den beschriebenen Unterscheidungen führt Strauss keine weitere systematische Betrachtung von „Articulation Work“ hinsichtlich deren Ausprägungen durch. In der Literatur existieren drei weitere Ansätze zur Differenzierung zwischen unterschiedlichen Arten von Articulation Work auf Basis einer unterschiedlichen Konzeptualisierung der abzustimmen Arbeit. Fjuk et al. (1997) stellen „Articulation Work“ der Activity Theory (Leont'ev, 1978) gegenüber und unterscheiden so verschiedene Ebenen in Arbeitsabläufen, die abzustimmen sind. Hampson und Junor (2005) führen ein Raster ein, das „Articulation Work“ hinsichtlich der Art des Arbeitsprozesses unterscheidet, in dem sie zur Anwendung kommt. Færgemann et al. (2005) unterscheiden Varianten von „Articulation Work“ je nach Reichweite der zugrunde liegenden Arbeitsprozesse. Alle drei Ansätze werden in der Folge im Detail beschrieben und bezüglich ihrer Implikationen für diese Arbeit betrachtet.

¹⁰ „what needs to be done, by whom, what resources are needed, what one has to offer, what one expects from others, who has what power, and so forth“ (Corbin und Strauss, 1993, S. 76)

2.2.1. Unterscheidung nach Fjuk, Smørdal und Nurminen

Fjuk et al. (1997) betrachten Articulation Work im Kontext von CSCW¹¹ und versuchen ein konzeptionelles Framework zu entwickeln, das die Rolle von Computersystemen im Kontext individueller und kollektiver Tätigkeiten erklärt – sie entwickeln also ein Erklärungsmodell für die Funktionsweise sozio-technischer Systeme (Emery und Trist, 1960). Während die Implikationen von „Articulation Work“ für CSCW an dieser Stelle nicht näher von Belang sind ist aber das theoretische Framework, das die Autoren ihren Ausführungen zu Grunde legen von Interesse.

Fjuk et al. (1997) beziehen sich bei ihren Überlegungen auf die „Activity Theorie“ (Tätigkeits-Theorie), die maßgeblich von (Leont'ev, 1972) geprägt wurde. Die Autoren argumentieren, dass diese einen Ansatzpunkt bietet, die von Strauss als relevant erkannten aber nicht näher behandelten „externen Faktoren“, die Arbeit beeinflussen, zu berücksichtigen. Der Begriff der „externen Faktoren“ wird mit allen Einflussfaktoren beschrieben, die nicht unmittelbar Teil des Arbeitsablaufs sind sondern technologischer, organisationaler, kultureller, wirtschaftlicher oder physiologischer Natur sind.

Ohne an dieser Stelle näher auf die „Activity Theory“¹² einzugehen, seien hier die drei Kernkonzepte der Theorie erwähnt:

- Activity (Tätigkeit)
- Action (Aktion)
- Operation (Operation)

Diese drei Konzepte bilden eine Hierarchie, in denen eine „Activity“ an oberster Stelle steht. Eine „Activity“ ist eine menschliche Tätigkeit, die durch ein Motiv getrieben ist und der (vorerst) individuellen Bedürfnisbefriedigung dient. Eine „Activity“ setzt sich aus mehreren „Actions“ zusammen, die jede für sich ein aus dem Motiv heraus begründbares Ziel haben und zur Bedürfnisbefriedigung direkt oder indirekt betragen. „Actions“ setzen sich wiederum aus „Operations“ zusammen, also einzelnen, nicht mehr bewusst ausgeführten Handlungen, die durch die Bedingungen des jeweiligen Umgebungskontexts bestimmt werden. Während sie lernen, transformieren Individuen laufend „Actions“ zu „Operations“, automatisieren also deren Ausführung, sodass sich die kognitive Belastung verringert (als klassisches Beispiel kann hier das Erlernen des Autofahrens dienen).

Die „Activity Theory“ beschreibt als psychologisches Modell vorerst das Individuum und dessen Verhalten. In sozialen Systemen, die auf Interaktion basieren, stößt das Modell jedoch an die Grenzen der erklärbaren Phänomene. Engeström (1987)

¹¹Computer Supported Cooperative Work

¹²für eine allgemein verständliche Einführung unter Berücksichtigung der praktischen Implikationen siehe Dahme und Raeithel (1997) oder Nardi und Kapteinin (2006)

baut auf der klassischen „Activity Theory“ auf und erweitert diese um den Aspekt der Gemeinschaft sowie der Interaktion in dieser bzw. der Rolle von Artefakten („Objects“) in derartigen Settings. Fjuk et al. (1997) bemängeln aber in ihren Ausführungen, dass Engeström in seinen Ausführungen abstrakt bleibt und nicht den Konkretisierungsgrad der originären „Activity Theory“ erreicht, was das Zusammenspiel der unterschiedlichen Ebenen („Activity“, „Action“ und „Operation“) betrifft.

Hinsichtlich der näheren Betrachtung von Articulation Work unterscheiden Fjuk et al. (1997) in Bezugnahme auf Strauss (1993) zwei Ebenen („levels“) von „Articulation Work“, namentlich „planned“ und „situated Articulation Work“. Diese Unterscheidung korrespondiert den Autoren nach im Wesentlichen mit der Unterscheidung zwischen „expliziter“ und „impliziter Articulation Work“. „Planned“ bezieht sich hier darauf, dass die „Articulation Work“ der Koordination eines vordefinierten Arbeitsablaufs dient, also nicht zur unmittelbaren Bewältigung von aufgetretenen Problemen dient. „Situated Articulation Work“ hingegen läuft ad-hoc im Arbeitsablauf bei Bedarf (d.h. zur Beseitigung von aufgetretenen Problemen) ab. Aufgrund dieser Definitionen (die auch der unabhängig davon getroffenen Unterscheidung zwischen „ad-hoc alignment“ und „coordination of predefined work“ in (Schmidt und Simone, 2000) entspricht), ist eine Gleichsetzung dieser Unterscheidung mit dem Unterschied zwischen impliziter und expliziter Articulation Work im Sinne von Strauss (1993) fragwürdig. Die Autoren relativieren die strikte Entsprechung auch selbst mit späteren Aussagen in der Arbeit, in der auf „situated Articulation Work“ Bezug genommen wird, die aber ob der herausfordernden Natur des Arbeitsablaufs „expliziter“ abzulaufen habe (Fjuk et al., 1997, S. 15). Die Unterscheidung zwischen „situated“ und „planned Articulation Work“ wird deshalb hier als eigenständig und orthogonal zu impliziter und expliziter „Articulation Work“ betrachtet.

Unter Einbeziehung der „Activity Theory“ und basierend auf der Unterscheidung zwischen „Activity“, „Action“ und „Operation“ führen Fjuk et al. (1997) außerdem zwei unterschiedliche Arten von „Articulation Work“ ein, die sich in ihren Bezugspunkten unterschieden und jeweils für den Fall individueller und kollektiver Tätigkeiten bzw. Aktionen betrachtet werden.

Articulation of action within individual activity Die Artikulation von Aktionen innerhalb einer Tätigkeit entspricht einer bewussten Planung eines Vorgehens zur Erreichung von definierten Zielen. Diese Form von „Articulation Work“ ist per Definition explizit. Sie umfasst lediglich Planungsaktivitäten eines Individuums und umfasst die Klärung der Fragen „wer“ (in diesem Zusammenhang das Individuum selbst oder andere) „was“ (im Sinne des zu erreichenden Ziels) „wo“ (im Sinne des örtlichen, zeitlichen oder organisationalen Kontexts) „wie“ (im Sinne der Operationalisierung der Aktionen zur Zielerreichung) arbeitet.

2. Articulation Work

Articulation of operation within action in individual activities Die Auswahl und Ausführung von Operationen im Kontext einer Aktion erfolgt zu meist nicht bewusst basierend auf Erfahrungswissen. Tatsächlich kann die Auswahl von adäquaten Operationen als ein permanenter Fluss von mit der produktiven Arbeit verwobenen „Articulation Work“-Vorgängen gesehen werden, der implizit auch in individuellen Arbeitssituationen abläuft. In diesem Zusammenhang ist es wichtig zu erwähnen, dass Operationen in „problematischen“ Situationen (im Sinne von Strauss) zu Aktionen werden können, die nicht mehr unbewusst und automatisiert ablaufen können. Mit dieser Transformation wird auch die „Articulation Work“ explizit und muss dass individuelle Vorgehen der geänderten Situation anpassen.

Articulation of individual action within collective activity Die Artikulation von Aktionen innerhalb einer kollektiven Tätigkeit geht in den Gegenständen der Artikulation über die im individuellen Fall zu berücksichtigenden Planungsaspekte („wer“, „was“, „wann/wo“, „wie“) hinaus. Zusätzlich müssen um Zuge der Artikulation die Regeln der Kommunikation und Arbeitsteilung zwischen den am Arbeitsprozess Beteiligten artikuliert werden. Die Artikulation umfasst hier auch die gegenseitige Offenlegung und Kenntnisnahme der individuellen „kognitiven Strukturen“ und existierender Annahmen über den Arbeitsablauf.

Articulation of individual operation within action in collective activity Im Gegensatz zur individuellen Artikulation von Operationen im Kontext von Aktionen ist diese im kollektiven Fall seltener implizit abzuwickeln. Unterschiedliche Auffassungen über Herangehensweisen oder Missverständnisse bedürfen zum Teil einer expliziten Klärung um die Zielerreichung zu gewährleisten. Operationen werden hier damit oft auf die Ebene von Aktionen gehoben und bewusst ausgehandelt.

Articulation of collective action in collective activity Die Kategorie der kollektiven Aktion wird von (Fjuk et al., 1997) nicht im Detail behandelt, da die „Activity Theory“ selbst diese nicht behandelt und auch keinerlei anderen diesbezüglich verwendbaren Forschungsergebnisse verwendbar wären. Jede Tätigkeit involviert auch kollektive Aktionen wie Aushandlungen, Konsensfindung oder gemeinsame Problemlösung. Bei der Artikulation von kollektiven Aktionen müssen alle beteiligten Individuen ihre Perspektive, ihr Wissen und ihre Überlegungen einbringen um die gemeinschaftliche Entwicklung voranzutreiben. Fjuk et al. (1997) treffen hier keine Aussagen hinsichtlich der Implikationen für „Articulation Work“.

Articulation of operations within collective actions in collective activity
Bei Zusammenarbeit auf Aktionsebene kann es durch die per Definition nicht

bewusst geplante Durchführung der individuellen Operationen zu Zielerreichung der kollektiven Aktion zu konfliktionären Situationen kommen. Vor allem, wenn die individuellen Vorstellungen des Arbeitsablaufs divergieren („weak common conceptual structures“), kann es notwendig sein, explizite „Articulation Work“ anzustoßen, um diese Vorstellungen offenzulegen und abzugleichen.

Innerhalb eines Arbeitsablaufs können auch mehrere der hier beschriebenen Kategorien auftreten. Teile von Arbeitsabläufen können durch Änderungen im Arbeitskontext die Kategorie wechseln und somit mehr oder weniger explizite „Articulation Work“ notwendig machen. Durch die Unterscheidung zwischen kollektiver Tätigkeit und Aktion wird es möglich „Articulation Work“ je nach Enge der Interaktion und den damit auftretenden unterschiedlichen Artikulationsbedürfnissen entsprechend auszulegen.

2.2.2. Unterscheidung nach Hampson und Junor

(Hampson und Junor, 2005) verwenden „Articulation Work“ als Framework zur Erklärung von „interactive customer service“, also dem jenen Kundenbeziehungen, bei denen die Interaktion zwischen Anbieter und Kunden im Vordergrund steht. Im Rahmen dieser Arbeit zeigen die Autoren auch die historische Entwicklung des Begriffs „Articulation Work“ auf und entwickeln einen Raster zur Einordnung unterschiedlicher Ausprägungen von Arbeit, die wiederum unterschiedliche Arten von „Articulation Work“ bedingen. Dieses Raster ist hier von Interesse.

Bezugnehmend auf (Strauss, 1993) unterschieden die Autoren einerseits zwischen Arbeitsabläufen, die *routine* sind, und solchen, die *non-routine* sind. Außerdem kann zwischen Arbeitsabläufen unterschieden werden, die *visible* oder *invisible* sind (Star und Strauss, 1999). Während *visible work* all jene Arbeitsabläufe umfasst, die als solche wahrgenommen werden, bezieht sich *invisible work* auf alle Arbeitsabläufe, die stattfinden aber nicht „offiziell“ wahrgenommen werden (also etwa nicht in einem Prozessmodell aufscheinen). Daraus ergeben sich vier zu unterscheidende Settings, in denen „Articulation Work“ stattfindet und die sich sowohl in der konkret als „Articulation Work“ ausgeführten Tätigkeit als auch in der möglichen methodischen und/oder technischen Unterstützung unterscheiden.

Visible routine work beschreibt jene Arbeitsabläufe, die von klassischen Management-Ansätzen erfasst werden, formalisiert werden können und in Unternehmen oft normiert vorgegeben sind (etwa in Form von Prozessmodellen oder durch die Vorgaben eines Workflow-Management-Systems). „Articulation Work“ findet hier zu definierten Zeitpunkten und explizit ausgelöst statt, um die normier-

2. Articulation Work

ten Abläufe zu definieren bzw. diese an veränderte Rahmenbedingungen anzupassen.

Visible non-routine work beschreibt Arbeitsabläufe in Umgebungen, die so dynamisch sind, dass normierte Abläufe aufgrund der raschen, nicht absehbaren Veränderungen der Anforderungen nicht sinnvoll einsetzbar sind. „Articulation Work“ tritt hier regelmäßig implizit und explizit auf, da jede Veränderung eine – je nach Ausmaß der Veränderung implizite oder explizite – Neuabstimmung der Zusammenarbeit nach innen und außen benötigt.

Invisible routine work umfasst all jene Arbeitsabläufe in Unternehmen, die zwar etabliert sind, von den traditionellen Steuer- und Kontroll-Werkzeugen im Unternehmen jedoch nicht erfasst werden. Sie sind formal nicht normiert, treten jedoch so regelmäßig auf, dass sich eine routinemäßige Herangehensweise herausbildet. Articulation Work läuft hier bei Veränderungen der Rahmenbedingungen zumeist implizit ab und sorgt dafür, dass die Interaktion zwischen den Beteiligten weiter funktioniert. Explizite „Articulation Work“ unter Einbeziehung der betroffenen Personen kann hier dafür sorgen, Arbeitsabläufe dieser Kategorie in den Bereich der „visible routine work“ überzuführen.

Invisible non-routine work umfasst jene Arbeitsabläufe, die zur Behandlung von unvorhergesehenen Anforderungen durchgeführt werden und die nach außen hin nicht sichtbar wird. Typisch treten derartige Situationen bei Ausnahmefällen in etablierten Arbeitsabläufen auf, bei denen die Tätigkeiten zu Wiederherstellung einer „regelkonformen“ Situation oft nicht durch Steuer- und Kontrollelemente erfasst werden und durch die Einzigartigkeit der Ausnahme oder des Kontexts, in dem diese auftritt, keine etablierten Handlungsmuster existieren. „Articulation Work“ ist hier ad-hoc notwendig, um adäquat auf die Anforderungen der Umwelt reagieren zu können. Sowohl explizite und implizite „Articulation Work“ kann hier zu Anwendung kommen, wobei als Entscheidungskriterien zwischen diesen beiden Ausprägungen die wahrgenommene Komplexität der Situation sowie die zur Lösung zur Verfügung stehende Zeit zu berücksichtigen sind.

In unterschiedlichen Arbeitssituationen können diese vier Kategorien auch kombiniert auftreten. Auch hier können manche Arbeitsabläufe durch erfolgreich durchgeführte „Articulation Work“ in eine andere Kategorie verschoben werden, wo der Bedarf an laufender ad-hoc Abstimmung geringer oder nicht vorhanden ist. Andere Arbeitsabläufe sind ihrer Natur nach nicht strukturierbar und formalisierbar, so dass „Articulation Work“ ein inhärenter Bestandteil des Ablaufs ist und trotz wiederholter Durchführung auch bleibt.

2.2.3. Unterscheidung nach Færgemann et al.

(Færgemann et al., 2005) sprechen einen Aspekt von „Articulation Work“ an, der von anderen Autoren in dieser Form nicht erwähnt wurde. Das strukturierende Merkmal ist in diesem Fall der „Reichweite“ der Arbeit, die zu koordinieren ist. Die „Reichweite“ (grob unterteilt in „lokal“ und „global“) beschreibt, ob die kooperierende Instanzen (Individuen oder Organisationseinheiten) miteinander vertraut sind, im täglichen Arbeitsverlauf ständig kooperieren und auch Zugriff auf die gleichen Informationsquellen haben und diese identisch interpretieren. Ist dies nicht der Fall, liegt ein „globales“ Arbeitssetting vor, in dem „Articulation Work“ anders als in lokalen Settings unterstützt werden muss.

Die Autoren führen dementsprechend vier unterschiedliche „Reichweiten“ von „Articulation Work“ an:

internal beschreibt „Articulation Work“, die zwischen ständig eng zusammenarbeitenden Individuen in einem etablierten Arbeitskontext durchgeführt wird.

semi-internal beschreibt die „Articulation Work“, die zwischen eng kooperierenden, aber organisational getrennten Einheiten auftritt.

semi-external bezeichnet die „Articulation Work“, die zwischen sporadisch interagierenden, organisational getrennten Einheiten auftritt, die jedoch noch unter einem gemeinsamen konzeptionellen Dach (also z.B. innerhalb einer bestimmten Arbeitsdomäne) arbeiten.

external bezeichnet jene „Articulation Work“, die über organisationale Grenzen hinweg durchgeführt wird, und in der auch kein gemeinsames Domänenwissen mehr vorausgesetzt werden kann.

Je weiter die Reichweite der „Articulation Work“ gefasst ist, desto expliziter und formalisierter muss diese den Autoren zufolge auch durchgeführt werden¹³

2.2.4. Zusammenfassung

In diesem Abschnitt wurden vier Arbeiten näher vorgestellt, die sich der Strukturierung des Konzepts „Articulation Work“ widmen. Die grundlegende Strukturierung bietet bereits Strauss (1985) (bzw. Strauss (1988) und Strauss (1993)). Die drei übrigen Arbeiten bauen auf Strauss auf und vertiefen das Verständnis von „Articulation Work“ weiter, in dem sie vor allem den im Zuge von „Articulation Work“ behandelten Gegenstand weiter detaillieren und strukturieren. Die drei Arbeiten gehen

¹³ „local articulation work often is based on immediate access and visibility [...] articulation work across unit boundaries is [...] much more demanding and is more dependent on a high degree of formalization in the interaction and coordination undertaken.“ (Færgemann et al., 2005, S. 178f)

2. Articulation Work

hierbei unterschiedliche Wege. Fjuk et al. (1997) setzen „Articulation Work“ in Beziehung zur aus der Psychologie stammenden „Activity Theory“ während Hampson und Junor (2005) und Færgemann et al. (2005) im Kontext der Soziologie bleiben und neben den Arbeiten von Strauss z.B. auch auf (Star und Strauss, 1999) aufbauen.

Fasst man die Konzepte zur Strukturierung von „Articulation Work“ aus allen hier besprochenen Arbeiten zusammen, so ergibt sich folgender Überblick:

- Ausprägung der „Articulation Work“
 - implizit vs. explizit¹⁴
- Ziel der „Articulation Work“
 - situated vs. planned¹⁵ bzw.
ad-hoc alignment vs. coordination of predefined work¹⁶
 - working out original arrangements vs. reworking arrangements¹⁷ vs. resolving contingencies¹⁸
- Reichweite der „Articulation Work“
 - internal vs. semi-internal vs. semi-external vs. external¹⁹
- Gegenstand der „Articulation Work“
 - alignment of procedures vs. alignment of meanings²⁰
 - routine vs. non-routine work²¹ bzw.
routine vs. problematic interaction (mit der belebten oder unbelebten Umwelt)²²
 - visible vs. invisible work²³
 - individual activity vs. collective activity vs. collective action²⁴
- Abstraktionsgrad des Gegenstandes der „Articulation Work“
 - activity-action vs. action-operation²⁵

¹⁴in (Strauss, 1993)

¹⁵in (Fjuk et al., 1997)

¹⁶in (Schmidt und Simone, 2000)

¹⁷in (Corbin und Strauss, 1993)

¹⁸z.B. in (Gerson und Star, 1986)

¹⁹in (Færgemann et al., 2005)

²⁰in (Sarini und Simone, 2002a)

²¹in (Hampson und Junor, 2005)

²²in (Strauss, 1993)

²³u.a. in (Suchman, 1995), (Suchman, 1999), (Star und Strauss, 1999), (Hampson und Junor, 2005)

²⁴in (Fjuk et al., 1997)

²⁵in (Fjuk et al., 1997)

Bezüglich des Ziels von „Articulation Work“ sind im Wesentlichen zwei Gegen-satzpaare zu identifizieren. „Situated Articulation Work“ wird während des Arbeits-ablaufs beim Auftreten von unvorhergesehenen Problemen durchgeführt und dient der ad-hoc-Abstimmung der Beteiligten. Obwohl diese in den meisten Fällen implizit abläuft, sind doch Fälle vorstellbar, in denen eine explizite, d.h. bewusst durchgeführte, „Articulation Work“ sinnvoll bzw. notwendig ist (siehe weiter unten – Ge-genstand der „Articulation Work“). „Planned Articulation Work“ dient der Koordi-nation von vordefinierten Arbeitsabläufen und kann – je nach Arbeitskontext – implizit oder explizit auftreten. Ein Großteil der Autoren, die den Begriff „Arti-culation Work“ prägen (u.a. (Strauss, 1985), (Gerson und Star, 1986) und z.T. auch (Schmidt und Bannon, 1992)), schränken diese auf deren Durchführung zur Lösung von im Arbeitsprozess aufgetretenen Probleme (also „situated“) ein. Die Durchfüh-ru ng von „Articulation Work“ zur Koordination von Arbeitsabläufen wird nur selten und erst in späteren Arbeiten explizit angesprochen (etwa bei (Grinter, 1996) oder (Fjuk et al., 1997)).

Die Unterscheidung zwischen „Working out arrangements“, „reworking arrange-ments“ und „resolving contingencies“ ist eine weitere Kategorisierung des Kontextes von „Articulation Work“, die sich auf den Zeitpunkt der Durchführung bzw. deren Zielsetzung bezieht. Die beiden erstgenannten Ausprägungen sind dabei im Regel-fall explizit, da sie eine bewusste Beschäftigung mit dem Arbeitsablauf bedingen, und steigen aus dem Arbeitssystem heraus. „Resolving contingencies“ wird im aktu-ellen Arbeitskontext durchgeführt und kann je nach wahrgenommener Komplexi-tät des Problems implizit oder explizit auftreten. „Reworking arrangements“ ist wie „resolving contingencies“ immer „situated“, da es immer aufgrund einer im Arbeits-prozess auftretenden problematischen Situation ausgelöst wird, die die Durchfüh-ru ng der bis dahin geltenden Modalitäten der Zusammenarbeit unmöglich macht. „Working out arrangements“, also die initiale Festlegung der Modalitäten einer Zu-sammenarbeit, kann nicht in die Unterscheidung zwischen „situated“ und „planned“ eingeordnet werden, da die „Articulation Work“ in diesem Fall weder auf aufgetre-tenen Problemen beruht noch die Koordination eines bereits bestehenden Arbeits-ablaufs zum Ziel hat.

Die Reichweite von „Articulation Work“ ist orthogonal zu den bereits genannten Unterscheidungen zu nennen, da diese alle in einer der vier Reichweiten-Kategorien auftreten können. Tendenziell ist „Articulation Work“ mit größerer Reichweite (also „semi-external“ oder „external“) eher explizit, da der Abstimmungsbedarf im Allge-meinen größer ist.

Hinsichtlich des Gegenstandes von „Articulation Work“ sind fünf Unterschei-dungskategorien zu identifizieren. Die Ausprägungen der jeweiligen Kategorien wei-sen zum Teil auf die Art der durchzuführenden Articulation Work hin.

2. Articulation Work

Die Kategorie „routine vs. non-routine work“ bezieht sich darauf, ob der fragliche Arbeitsablauf für die beteiligten Personen alltäglich ist und unter bekannten Rahmenbedingungen stattfindet oder nicht. Je stärker der „non-routine“-Anteil in einem Arbeitsablauf zum Tragen kommt, desto expliziter muss im Allgemeinen die „Articulation Work“ sein – bei Routine-Arbeit ist der Bedarf an Articulation gering und beschränkt sich auf implizit durchführbare Detailabstimmungen zwischen den Beteiligten.

Obwohl vordergründig unterschiedlich, bezieht sich die nächste Kategorie „routine vs. problematic interaction“ auf den gleichen Sachverhalt. Strauss (1993), von dem diese Unterscheidung stammt, bezeichnet Interaktion als die Grundlage von Arbeitsabläufen und als wesentlichen Bestandteil derselben. Der „routine“-Begriff kann deshalb mit jenem der zuvor beschriebenen Unterscheidung gleichgesetzt werden. Der Begriff der „problematic interaction“ beschreibt insofern das gleiche Phänomen wie der der „non-routine work“ als dass er sich ebenfalls auf die erhöhte kognitive Belastung der beteiligten Personen bei der Zielerreichung bezieht. Dementsprechend impliziert „problematic interaction“ eine meist explizite „Articulation Work“, während „routine interaction“ meist durch implizite „Articulation Work“ produktiv gehalten werden kann.

Die Unterscheidung zwischen „visible“ und „invisible work“ bezieht sich auf die Kenntnisnahme eines Arbeitsablaufs in seinem Durchführungskontext und dessen Wahrnehmung durch andere – vor allem auch übergeordnete – organisationale Instanzen. Während „visible work“ definierten Aufgaben dient, formalisiert werden kann und durch Steuer- und Kontrollinstrumente oder organisationale Unterstützungsgeräte erfasst werden kann, bleibt „invisible work“ im organisationalen Kontext verborgen und ist nur für die handelnden Individuen sichtbar (und wird dementsprechend auch organisational nicht unterstützt und wertgeschätzt). Für „Articulation Work“ hat dies per se keine unmittelbaren Auswirkungen, außer dass „visible work“ immer ein Ergebnis expliziter „Articulation Work“ ist. Dies bedeutet gleichzeitig, dass explizite „Articulation Work“ (unter Einbeziehung sowohl der unmittelbar am Arbeitsablauf beteiligten Personen als auch der „übergeordneten“ Instanzen) dazu beitragen kann, „invisible work“ zu „visible work“ zu machen (siehe dazu auch (Fujimura, 1987)). „Articulation Work“ ist somit ein Mittel, einen Abgleich zwischen dem offiziellen (organisational festgeschriebenen) Verständnis eines Arbeitsablaufs und dem tatsächlichen Ablauf, wie er in der Praxis ausgeführt wird, durchzuführen. „Articulation Work“ kann damit eine Realisierung eines organisationalen Lernschritts sein, der im Sinne von Argyris und Schön (1978) die „espoused theories“ (die offiziell veröffentlichten Theorien über Arbeit) mit den „theories-in-use“ (die tatsächlich handlungsleitenden Theorien) abgleicht bzw. im Sinne von Sachs (1995) einen „tacit organisational view“ in einen „explicit organisational view“ überführen (siehe dazu auch die Ausführungen in Kapitel XY (Einführung)).

Die Enge der notwendigen Kooperation bei der Durchführung eines Arbeitsablaufs (festgemacht an den Handlungs-Kategorien der „Activity Theorie“) ist Gegenstand der letzten Kategorie. „Individual activity“ beschreibt Arbeitsabläufe, die im Wesentlichen von einem Individuum ausgeführt werden und lediglich an den Schnittstellen zu Beginn und am Ende Interaktion benötigen. „Collective Activity“ beschreibt Arbeitsabläufe, in denen mehrere Individuen klar abgegrenzte Teile der Arbeit übernehmen und Interaktion an festgelegten Schnittstellen bzw. zu festgelegten Zeitpunkten stattfindet. Dies entspricht im Wesentlichen der klassischen Arbeitsteilung in Unternehmen. „Collective Action“ beschreibt tatsächlich kooperative Arbeit im engeren Sinn, deren Durchführung nur durch enge Interaktion mehrere Individuen auch in Detailaspekten notwendig ist. Je enger die Kooperation, desto notwendiger wird „Articulation Work“, wobei diese in allen Fällen sowohl in ihrer impliziten als auch expliziten Ausprägung zum Einsatz kommen kann.

Zur Identifikation der im Einzelfall sinnvollen Variante von „Articulation Work“ (implizit oder explizit) ist die Berücksichtigung der letztgenannten Unterscheidung hinsichtlich des Abstraktionsgrades der „Articulation Work“ notwendig. Beschäftigt sich „Articulation Work“ mit der abstrakteren Ebene zwischen „activity“ und „action“, steht vorrangig die Betrachtungsdimension „Was?“ (also die Ziele und das generelle Vorgehen) im Zentrum. Bei Arbeitsabläufen, die „non-routine“ sind, kommt dabei eher explizite Articulation Work zum Einsatz. Bei „routine“-Arbeitsabläufen ist „Articulation Work“ auf dieser abstrakten Ebene implizit nicht und explizit nur dann notwendig, wenn der Ablauf selbst nicht mehr anwendbar ist (also „problematic“ bzw. „non-routine“ wird) und hinterfragt werden soll („reworking arrangements“). Auf der konkreten Ebene zwischen „action“ und „operation“ (also der Frage nach dem „Wie?“) kommt in individuell abgehandelten Arbeitsabläufen vorrangig implizite „Articulation Work“ zum Einsatz. Treten unvorhergesehene Probleme auf oder erscheinen etablierte Operationen nicht mehr adäquat, kommt zur Klärung wieder explizite Articulation Work zum Einsatz. In kollektiven Arbeitsprozessen ist die Enge der Interaktion entscheidend. Bei klar separierbaren Arbeitsanteilen (also bei Interaktion auf „activitiy“-Ebene) bleibt die Entscheidung zur konkreten Umsetzung beim Individuum und die „Articulation Work“ im Normalfall implizit (Ausnahmen siehe Ausführungen zur individuellen Arbeitsabläufen). Bei Arbeitsabläufen mit enger Interaktion auf Aktionsebene muss diese im Normalfall im Vorhinein („working out arrangements“) durch explizite „Articulation Work“ ausgehandelt werden. Während des Arbeitsablaufs kann wiederum implizite „Articulation Work“ zurückgegriffen werden, wobei auf Grund der Anzahl der beteiligten Personen die Wahrscheinlichkeit steigt, dass ein beteiligtes Individuum die Interaktion als „problematic“ empfindet und dies wiederum explizite „Articulation Work“ notwendig macht.

2. Articulation Work

Zusammenfassend können die hier beschriebenen Zusammenhänge wie in Abbildung 2.3 dargestellt beschrieben werden. Diese Darstellung stellt den Versuch einer Strukturierung des Konzepts „Articulation Work“ dar, um deren mögliche Ausprägungen in unterschiedlichen Arbeitskontexten zu zeigen. Sie ist keine Beschreibung der möglichen Abläufe eines „Articulation Work“-Prozesses. In der Abbildung sind die wesentlichen Inhalte der oben behandelten Einteilungen abgebildet.

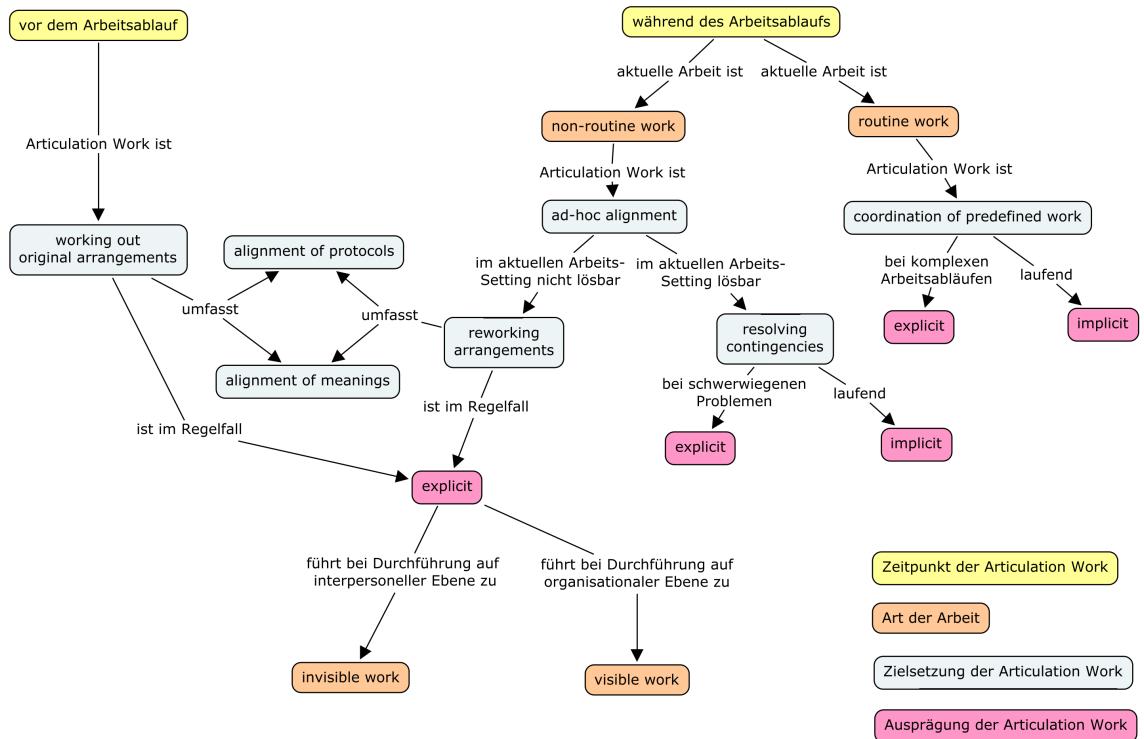


Abbildung 2.3.: Articulation Work im Durchführungskontext

Die erste Unterscheidung, die am oberen Rand der Grafik dargestellt ist, betrifft den Zeitpunkt der „Articulation Work“. Während von den meisten Autoren lediglich jene Phasen eines Arbeitsprozesses zu „Articulation Work“ gezählt werden, die während der Durchführung der „Production Work“ auftreten, bezeichnet jedoch Strauss selbst in (Corbin und Strauss, 1993) auch den Vorlauf eines Arbeitsprozesses (also die Konzeptions- und Planungsphase) als „Articulation Work“ (auch die ursprüngliche Definition von Stauss schließt diese Phase nicht aus). Nach Corbin und Strauss (1993) ist „Articulation Work“ in dieser Phase immer explizit (entsprechend dem in der Arbeit angegebenen Schema). Obwohl grundsätzlich nichts gegen eine implizite Durchführung dieser Phase spräche (bei „einfachen“ Arbeitsaufgaben, die von etablierten (Gruppen von) Individuen übernommen werden), wird diese Variante von den Autoren nicht erwähnt und deshalb in der Abbildung nicht dargestellt.

Je nach Kontext, in dem die „Articulation Work“ durchgeführt wird (organisational, d.h. in offiziellem Rahmen oder informell ausschließlich in der lokalen Arbeitsumgebung) führt sie zu „visible“ oder „invisible work“.

„Articulation Work“, die im Zuge des Arbeitsablaufs ausgeführt wird, kann zwei Zwecken dienen. Einerseits kann sie im Falle etablierter Arbeitsabläufe („routine“) der Koordination der ohnehin definierten Abläufe dienen (die nicht notwendigerweise „visible“ sein müssen). Dies wird im Regelfall implizit ablaufen, lediglich bei komplexen Arbeitsabläufen oder z.B. in verteilten Umgebungen (vgl. (Carstensen und Schmidt, 1999)) wird eine explizite Beschäftigung mit der Koordination notwendig sein. Andererseits kann „Articulation Work“ auch in Situationen („ad-hoc“) zur Anwendung kommen, in denen keine etablierten Arbeitsabläufe existieren oder diese aufgrund von Veränderung in der Arbeitsumgebung nicht adäquat sind. Dieser Fall beschreibt „Articulation Work“ im engeren Sinne, die der Wiederherstellung der Arbeitsfähigkeit in Situationen dient, in denen die Ausführung von „Production Work“ (aus welchen Gründen auch immer) nicht mehr möglich ist.

Bei der „ad-hoc“-Anwendung von „Articulation Work“ sind unterschiedliche Eskalationsstufen zu unterscheiden. In vielen Situationen, die von einzelnen beteiligten Individuen als „problematisch“ oder „non-routine“ wahrgenommen werden, ist die Auflösung dieser Bedenken durch einfache implizite „Articulation Work“, also ohne bewusste Beschäftigung im Rahmen der üblichen sozialen Interaktion möglich. Wird die aktuelle Arbeit als „problematischer“ wahrgenommen, muss eine explizite (d.h. bewusste) Beschäftigung mit der Situation erfolgen, um die „Production Work“ wieder in einen operativen Zustand zu versetzen oder verbleibende Unklarheiten implizit beseitigen zu können.

Es können jedoch Situation auftreten, in denen die aktuelle Arbeitspraxis nicht mehr soweit angepasst werden kann, dass die „Production Work“ wieder aufgenommen werden kann. Dies ist vor allem in Situationen der Fall, in denen es zu massiven Veränderungen des Arbeitskontexts gekommen ist (etwa durch veränderte Vorschriften, neue Werkzeuge oder Personaländerungen). In diesen Fällen muss die Arbeitspraxis selbst hinterfragt werden, was nach (Corbin und Strauss, 1993) im Wesentlichen einer Neuplanung unter Berücksichtigung der bislang gemachten Erfahrungen dient. Diese Überarbeitung ist immer explizit und kann je nach Durchführungskontext wiederum zu „visible“ oder „invisible work“ führen. Sie umfasst je nach Art des aufgetretenen Problems „alignment of meaning“ (also die Entwicklung einer gemeinsamen Sicht aller beteiligten Individuen auf die Arbeitsdomäne) und / oder „alignment of procedures“ (also die Abstimmung der Modalitäten der Zusammenarbeit).

An dieser Stelle ist es wichtig, nochmals darauf hinzuweisen, dass die getroffene Unterscheidung keine strikte, präskriptive Kategorisierung von „Articulation Work“

2. Articulation Work

darstellt. Wie auch von Schmidt und Simone (2000) angemerkt²⁶, sind die Grenzen zwischen den unterschiedlichen Ausprägungen von „Articulation Work“ fließend. Auch die Kontexte, die zur Durchführung einer bestimmten Ausprägung führen, sind dynamisch und können sich während der Durchführung der „Articulation Work“ selbst ändern.

In der Darstellung nicht enthalten sind die Unterscheidungen, die nach (Fjuk et al., 1997) auf die „Activity Theory“ zurückzuführen sind. Es sind dies die Unterscheidung nach Kooperations-Art der Arbeit („individuell“ vs. „lose kooperativ“ vs. „eng kooperativ“) und dem Abstraktionsgrad der „Articulation Work“ (abstrakt zwischen „activity“ und „action“ vs. konkret zwischen „action“ und „operation“). Hintergrund der Entscheidung, diese Unterscheidungen in der Abbildung zu vernachlässigen, ist deren fehlende Wirkung auf die Ausprägung der durchgeführten „Articulation Work“ (implizit oder explizit). In allen Kooperations-Arten ist die Durchführung von sowohl impliziter als auch expliziter „Articulation Work“ auf beiden Abstraktionsstufen denkbar. Es ändert sich lediglich die konkrete Ausgestaltung der „Articulation Work“ bzw. die Notwendigkeit einer eventuellen Unterstützung.

Außerdem fehlt in der Darstellung die Unterscheidung anhand der Reichweite von „Articulation Work“ nach (Færgemann et al., 2005). Diese hat durchaus Einfluss auf die Ausprägung der durchgeführten „Articulation Work“, steht aber vollständig orthogonal zu den abgebildeten Unterscheidungen. Die Komplexität der Darstellung wäre deshalb durch die Integration dieser Unterscheidung auf eine Maß angestiegen, die eine sinnvolle Interpretation der Grafik nicht mehr zugelassen hätte. Die Reichweite ist deshalb implizit in der Unterscheidung nach „routine“ und „non-routine work“ codiert, da domänenübergreifende Arbeitssituationen tendenziell eher „non-routine“ bzw. „problematic“ sind und dadurch eher zu expliziterer „external Articulation Work“ führen.

Auf Basis dieser dargestellten Struktur werden die im folgenden Abschnitt beschriebenen Arbeitsaspekte in unterschiedlichen Ausprägungen von „Articulation Work“ abgestimmt bzw. koordiniert. Jede dieser Ausprägungen muss in den unterschiedlichen Kontexten, in denen sie auftreten können, verschieden stark und mit unterschiedlichen Mitteln unterstützt werden. Abschnitt 2.4 widmet sich auf Basis einer Literaturstudie dem State-of-the-Art dieser Unterstützung aus organisationaler, methodischer und technischer Sicht.

²⁶ „As already indicated, the general modalities of articulation work which we for the sake of clarity discussed separately — ad hoc alignment and improvisation on the basis of mutual awareness versus action constrained by coordinative artifacts and protocols — are not ‘natural kinds’ in the Aristotelian sense: they do not exist as distinct domains of action, they are analytical distinctions.“ (Schmidt und Simone, 2000, S. 7)

2.3. Abzustimmende Arbeitsaspekte

Abbildung 2.4 zeigt eine Übersicht über Aspekte, die im Rahmen von „Articulation Work“ Gegenstand der Abstimmung zwischen den beteiligten Individuen sein können (entnommen aus (Schmidt und Simone, 1996)). In Abbildung 2.5 sind diese Aspekte nochmals hinsichtlich der Zusammenhänge zwischen ihnen diagrammatisch dargestellt.

Obwohl diese Übersicht aus Beobachtungen induktiv abgeleitet ist und keinen Anspruch auf Vollständigkeit erhebt, zeigt sie doch eine Anzahl von konkreten Ausprägungen der von Strauss angeführten Dimensionen von Arbeit, die im Rahmen von „Articulation Work“ abzustimmen sind („Wer?“, „Was?“, „Wann/Wo?“ und „Wie?“). Mit der Unterscheidung zwischen den nominalen Aspekten von Arbeit und der konkreten Manifestation eines Arbeitsablaufs im organisationalen Kontext stellen die Autoren einen in dieser Form bislang nicht explizit angesprochenen Unterschied bei der Durchführung von „Articulation Work“ dar.

Zusätzlich zu den von Strauss angeführten Dimensionen nimmt die Frage nach dem „Womit?“ – also den Arbeitsmitteln, den Ressourcen bzw. dem Arbeitsumfeld an sich – sowohl auf konzeptioneller als auch auf konkreter Ebene eine großen Stellenwert ein. Die Behandlung dieses Aspektes dient vor allem der Bildung einer gemeinsamen Sichtweise auf den Arbeitskontext, in dem der betreffende Arbeitslauf abgehandelt wird.

Von Interesse sind auch die den einzelnen Aspekten zugeordneten Prädikate, die eine Aussage über die konkret durchzuführenden Aktivitäten zulassen. Diese Aktivitäten sind sowohl dem Bereich der „Production Work“ (z.B. „do“, „use“) als auch der „Articulation Work“ zuzuordnen (z.B. „assign to“, „allocate“). Zum Teil sind auch Beziehungen zwischen den Aspekten angeführt, die eine notwendige Abstimmung über die Grenzen der einzelnen Aspekte hinweg anzeigen (z.B. „Task realized by Activity“ oder „Role responsible for Task“).

2.4. Unterstützung von Articulation Work

Nach den ersten Arbeiten von Strauss zum Thema „Articulation Work“ wurde das Konzept rasch als Erklärungsmodell für die Vorgänge im Zuge kooperativer Arbeit aufgenommen. Bereits 1986 verweisen Gerson und Star auf die Notwendigkeit einer expliziten Unterstützung von „Articulation Work“²⁷. Anhand der historischen

²⁷ „Methods for analyzing due process means, in this perspective, explicit procedures for evaluating and reconciling incompatibilities among different bodies of tacit local knowledge.“(Gerson und Star, 1986, S. 266)

2. Articulation Work

Nominal		Actual	
Elemental categories of articulation work	Elemental predicates	Elemental categories of articulation work	Elemental predicates
<i>Articulation work with respect to the cooperative work arrangement</i>			
Role	assign to [Committed actor]; responsible for [Task, Resource]	Committed-actor	assume , accept, reject [Role]; initiate [Activity];
Task	point out, express; divide, relate; allocate, volunteer; accept, reject; order, countermand; accomplish, assess; approve, disapprove; realized by [Activity]; to be aligned with [Task]	Activity/Action	[Committed actor] initiate; [Actor-in-action] undertake, do, accomplish; realize [Task]; [Actor-in-action] makes publicly perceptible, monitors, is aware of, explains, questions; aligned with [Activity]
Personnel	locate, allocate, reserve;	Actor-in-action	does [Activity];
<i>Articulation work with respect to the field of work</i>			
Conceptual structures	categorize: define, relate, exemplify relations between categories pertaining to [Field of Work];	State of field of work	classify aspect of [State of field of work]; monitor, direct attention to, make sense of, act on aspect of [State of field of work];
Informational resource	locate, obtain access to, block access to;	Informational resources-in-use	show, hide content of; publicize, conceal existence of;
Material resource	locate, procure; allocate, reserve to [Task];	Material resources-in- use	deploy, consume; transform;
Technical resource	locate, procure; allocate, reserve to [Task];	Technical resources- in-use	deploy; use;
Infrastructural resource	reserve;	Infrastructural resources-in-use	use;

Abbildung 2.4.: Abzustimmende Arbeitsaspekte (entnommen aus (Schmidt und Simone, 1996))

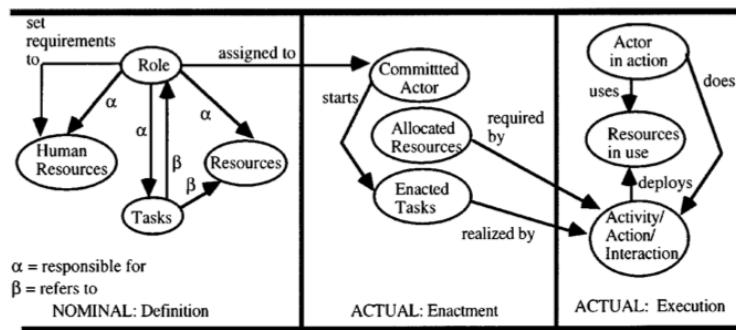


Abbildung 2.5.: Zusammenhänge zwischen Arbeitsaspekten (entnommen aus (Davitini und Simone, 2000))

Entwicklung von Mitte der 1980er-Jahre bis Ende des ersten Jahrzehntes des neuen Jahrtausends werden in den folgenden Abschnitten Maßnahmen zur Unterstützung beschrieben und in den jeweiligen Anwendungskontext gesetzt. Hierbei werden alle Arbeiten berücksichtigt, die sich direkt auf den von Strauss geprägten „Articulation Work“-Begriff beziehen. In der Literatursuche wurden dazu Datenbanken aus den Bereichen Informatik, Psychologie, Soziologie, den Wirtschaftswissenschaften sowie der Organisationslehre durchsucht (für eine detaillierte Auflistung siehe Abschnitt A.1). Nach der initialen Suche wurde jeweils auch die in den gefundenen Arbeiten referenzierte Sekundärliteratur aufgearbeitet. Des Weiteren wurden mit Hilfe von rückwärts verlinkenden Datenbanken (wo vorhanden) Publikationen erfasst, die die bislang gefundenen Arbeiten referenzieren und diese hinsichtlich ihrer Relevanz überprüft. Insgesamt ergab sich so eine Sammlung von 70 Publikationen (inklusive der grundlegenden Arbeiten, die bereits oben beschrieben wurden). Von diesen 70 Publikationen wurden 13 Arbeiten einer näheren Betrachtung unterzogen, da sie Aussagen zur Unterstützung von „Articulation Work“ treffen. Die übrigen Publikationen geben hierzu keine Aussage ab oder sind im Kontext größerer Forschungsvorhaben entstanden, die bereits durch eine repräsentative Arbeit in der detaillierten Betrachtung enthalten sind. Sämtliche Arbeiten sind in Anhang A hinsichtlich ihres Inhalts beschrieben und ihrer Relevanz für die Unterstützung von „Articulation Work“ beurteilt. In Anhang A ist außerdem eine Übersichtsgrafik zu finden, die das gesamte Feld an Publikationen zum Thema „Articulation Work“ visualisiert und die Zusammenhänge zwischen den einzelnen Arbeiten und Forschungsvorhaben aufzeigt.

2.4.1. Vorgehen zur detaillierten Betrachtung

2. Articulation Work

Zur strukturierten Betrachtung der Unterstützung von „Articulation Work“ wird ein einheitlicher Raster angewandt, anhand dessen die aus unterschiedlichen Forschungsgebieten stammenden und in unterschiedlichen Anwendungsdomänen angewandten Arbeiten einander gegenüber gestellt werden können. Neben den eigentlichen Unterstützungsmaßnahmen ist zur Bewertung derselben auch Kontextinformation notwendig, die die unterschiedlichen Ansätze offenlegt. Folgende Merkmale bzw. Inhalte einer Arbeit werden dazu betrachtet:

Kontext Forschungsgebiet aus dem das Konstrukt „Articulation Work“ betrachtet wird bzw. in dessen Kontext es zur Anwendung gebracht wird und / oder abstraktes oder konkretes Problemfeld, in dem „Articulation Work“ als Analysedimension oder zur Ableitung von Maßnahmen angewandt wird.

Unterstützung Konkrete oder abstrakte Maßnahmen oder Werkzeuge, die zur Unterstützung von „Articulation Work“ vorgeschlagen und/oder umgesetzt werden. Ggf. unterschieden in

- organisationale Unterstützung
- methodische Unterstützung
- technische Unterstützung

Auswirkungen Tatsächliche oder vermutete Auswirkungen der Unterstützung auf die durchgeführte „Articulation Work“.

Bewertung Zusammenfassung des Beitrags der Arbeit zur Unterstützung von „Articulation Work“ und Bewertung der Relevanz für die vorliegende Arbeit.

Die als relevant betrachteten Publikationen sind methodisch unterschiedlich ausgerichtet. Ein großer Anteil beschreibt rein empirisch-deskriptiv ein beobachtetes Phänomen und zieht Schlüsse hinsichtlich möglicher bzw. notwendiger Ausprägungen von „Articulation Work“ in bestimmten Anwendungsdomänen. Ein anderer Teil fokussiert auf die organisationale und/oder technische Unterstützung von „Articulation Work“, zum Teil ohne auf eigene empirische Ergebnisse aufzubauen oder diese zu erheben. Aus diesem Grund kann das oben angegebene Raster nicht immer vollständig gefüllt werden. Wo hinsichtlich einer bestimmten Dimension keine Information vorhanden ist, wird explizit im Text darauf hingewiesen. Wo mehrere Publikation eines Autors oder einer Gruppe zum gleichen Forschungsgegenstand existieren, wurden die als am relevantesten beurteilten Arbeiten (i.A. jene, die das jeweilige Forschungsvorhaben abschließend beschreiben) herausgegriffen und exemplarisch detailliert ausgearbeitet.

2.4.2. Modeling Articulation Work in Software Engineering Processes

Die erste Publikation, die sich mit der organisationalen Unterstützung von „Articulation Work“ in Form der expliziten Berücksichtigung von „Articulation Work“ in formalisierten Ablaufmodellen beschäftigt, ist die Arbeit von Mi und Scacchi (1991).

Kontext

Mi und Scacchi (1991) betrachten „Articulation Work“ im Kontext der Softwareentwicklung und argumentieren für deren explizite Berücksichtigung in Software Engineering Prozessen. In einer Literaturstudie zeigen sie, dass (die zum Zeitpunkt der Erstellung) verfügbaren Software-Engineering-Prozess-Modellierungs-Techniken die Einbindung von „Articulation Work“ in die Vorgehensmodelle nicht ermöglicht²⁸. Die Autoren selbst haben ihren Hintergrund ebenfalls in der Domäne des Software Engineering und wählen ihren Zugang zu Thematik dementsprechend, indem sie eine formale Abbildung des „Articulation Work“-Prozesses und eine Unterstützung durch regelbasierte Heuristiken zur Lösungsfindung vorschlagen.

Unterstützung

Die Autoren formalisieren im ersten Schritt den Ablauf von „Articulation Work“ im Kontext von Softwareengineering (siehe Abbildung 2.6). Die einzelnen Schritte leiten sie aus drei empirischen Studien ab, die sowohl hinsichtlich ihres Inhalts als auch ihrer Durchführung nicht näher beschrieben werden.

Die Autoren beziehen sich also offensichtlich auf explizite „Articulation Work“, die „ad-hoc“ – beim Auftreten eines Problems im Software Engineering Prozess – ausgelöst wird.

Zur Durchführung dieses Prozesses schlagen die Autoren einen Satz von regelbasierten Heuristiken vor, aus denen die betroffenen Individuen (hier: „agents“) auswählen können. Diese Heuristiken beschreiben mögliche Tätigkeiten im Zuge der „Articulation Work“ (ausdefiniert durch ECA²⁹-Regelsätze). Dabei geben die Autoren Heuristiken zur Problemlösung („problem-solving heuristics“, die der direkten Behebung der aufgetretenen Probleme dienen) und Heuristiken zur Auswahl geeigneter Lösungen („selection heuristics“) an.

²⁸ After we compare these findings with the process modeling techniques, it becomes apparent that current software process modeling techniques do not directly address articulation work. (Mi und Scacchi, 1991, S. 192)

²⁹ Event-Condition-Action

2. Articulation Work

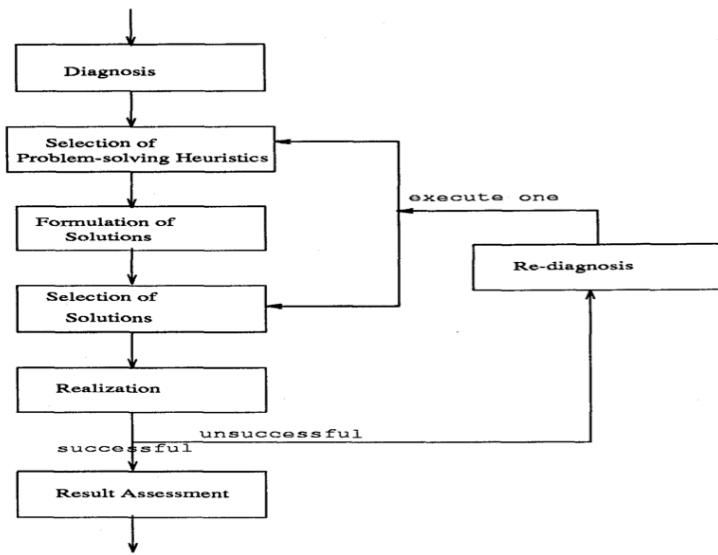


Abbildung 2.6.: Artikulations-Prozess (entnommen aus (Mi und Scacchi, 1991))

Auswirkungen

Das vorgeschlagene System wurde auf konzeptioneller Ebene entwickelt und nicht praktisch umgesetzt. Insofern existieren keinerlei reale Erfahrungen mit dem Ansatz. Anhand eines hypothetischen Beispiels demonstrieren die Autoren jedoch die Anwendung des Modells und der Heuristiken.

Der Vorteil liegt im Wesentlichen darin, dass der „Articulation Work“-Prozess durch seine Formalisierung bekannt ist („visible“ in Sinne der obigen Ausführungen) und dementsprechend auch offiziell auftreten „darf“. Durch den vorgegebenen Satz an Heuristiken sind außerdem die Alternativen zur Problembehandlung und deren Durchführung bekannt. Die Autoren geben diese Heuristiken für den Bereich des Software-Engineering an, betonen aber deren exemplarischen Charakter – die Heuristiken und vor allem deren konkrete Umsetzung (durch die Spezifikation von ECA-Regeln) müssen an die jeweilige Arbeits-Domäne angepasst werden.

Bewertung

Das vorgeschlagene Prozess-Modell von „Articulation Work“ bildet den Ablauf auf so abstrakter Ebene ab, dass es für „ad-hoc explicit Articulation Work“ zur Lösung unmittelbar auftretender Probleme allgemein (d.h. unabhängig von der Anwendungsdomäne) anwendbar erscheint und auch in mit den von Corbin und Strauss

(1993) genannten Schritten bei der Durchführung expliziter Articulation Work in Einklang gebracht werden kann.

Die Angabe von (exemplarischen) Heuristiken zur Durchführung des Artikulations-Prozesses erscheint insofern sinnvoll, als dass diese domänen- und organisations-spezifische Lösungsstrategien auch für unerfahrene Teilnehmer zugänglich machen können.

In Bezug auf die Generalisierbarkeit des Ansatzes problematisch zu sehen ist jedoch die Verwendung von ECA-Regeln zur Spezifikation der Durchführung der einzelnen Heuristiken. Die Angabe derartiger Regeln erscheint nicht in allen Anwendungsbereichen in einem sinnvollen Detaillierungsgrad möglich zu sein. Vor allem soziale Prozesse in kooperativen Umgebungen, auf die die Autoren der ursprünglichen Literatur zum Thema „Articulation Work“ stark Bezug nehmen, können in diesen Regeln nicht sinnvoll (im Sinne einer Durchführungsvorschrift) abgebildet werden.

Kontext	Unterstützung von Software Engineering
Art von AW	ad-hoc explizit
Unterstützung	<i>organisational</i> durch Formalisierung und a-priori-Spezifikation des Articulation-Prozesses („Was?“) und dessen Ausgestaltung („Wie?“)
Auswirkungen	Definiertes Vorgehen, wie „Articulation Work“ durchgeführt werden muss

2.4.3. Taking CSCW seriously: Supporting Articulation Work

Schmidt und Bannon (1992) begründen mit dieser Arbeit eine Entwicklungsrichtung der CSCW, die neben der Unterstützung der eigentlichen produktiven Arbeit auch auf die Unterstützung von „Articulation Work“ fokussiert. Sie beschreiben damit erstmals Anforderungen an die und Möglichkeiten der technische Unterstützung von „Articulation Work“.

Kontext

Schmidt und Bannon (1992) widmen sich in ihren Ausführungen der kooperativen Arbeit und der Unterstützung derselben durch Computersysteme. Sie argumentieren, das zum Zeitpunkt der Erstellung der Arbeit CSCW auf einer sehr formalisierten, strukturierten Sichtweise von kooperativer Arbeit aufbaut (z.B. im Sinne einer präskriptiven Workflow-Unterstützung) und Aspekte wie individuelle Arbeitsweise

2. Articulation Work

und Kommunikation unter den Beteiligten vernachlässigt. Dieser Sichtweise setzen sie einen Ansatz entgegen, der auch die Unterstützung von „Articulation Work“ berücksichtigt und damit den handelnden Individuen selbst die Kontrolle über die Interaktion übergibt³⁰

Die Autoren selbst arbeiten vor ihrem Hintergrund aus der Informatik, der Soziologie und den Arbeitswissenschaften. Sie stellen die Technologie nicht ins Zentrum ihrer Überlegungen, sondern betonen deren unterstützende Funktion für Individuen, die in Organisationen arbeiten. Die Untersuchungsdomäne ist dabei explizit nicht eingeschränkt, kooperative Arbeit wird in allen ihren Ausprägungen und in beliebigen Kontexten betrachtet³¹.

Unterstützung

Bei ihren Ausführungen zur unterstützenden Wirkung von CSCW bei „Articulation Work“ konzentrieren sich die Autoren auf zwei Aspekte von CSCW und zeigen, wie diese ausgestaltet sein müssen, um „Articulation Work“ zuzulassen bzw. zu unterstützen.

Der erste Aspekt, auf den eingegangen wird, ist die Unterstützung von Workflows durch Computersysteme. Um hier „Articulation Work“ zu berücksichtigen, ist die Unterstützung der Selbst-Organisation der kooperierenden Personen ein zentraler Punkt³². In diesem Sinne argumentieren die Autoren gegen einen fixen, unveränderlich vorgegebenen Workflow mit a-priori definierten Zuständigkeiten, sondern schlagen die Ermöglichung bzw. Unterstützung von Aushandlungsprozessen vor, in denen der Arbeitsablauf entsprechend dem aktuellen Arbeitskontext angepasst und kooperativ abgearbeitet werden kann. Anhand der angeführten Beispiele wird deutlich, dass dabei eher von „Articulation Work“ zur Koordinierung etablierter Arbeitsprozesse ausgegangen wird, die vorgeschlagene Unterstützungsfunctionalität ist jedoch auch für „ad-hoc Articulation Work“ sinnvoll anzuwenden (in Klammer jeweils die Zuordnung zur Art von „Artikulation Work“):

- Offenlegung des der Workflowunterstützung zugrunde liegenden Modells und Unterstützung der Interpretation und Reflexion desselben („ad-hoc implicit and explicit Articulation Work“)

³⁰ „Thus, by entering into cooperative work relations, the participants must engage in activities that are, in a sense, extraneous to the activities that contribute directly to fashioning the product or service and meeting requirements.“(Schmidt und Bannon, 1992, S. 8)

³¹ „In sum, we certainly want CSCW to address the aspects of computer support for cooperative work wherever they occur.“(Schmidt und Bannon, 1992, S. 11)

³² „Computer support of cooperative work should aim at supporting self-organization of cooperative ensembles as opposed to disrupting cooperative work by computerizing formal procedures.“(Schmidt und Bannon, 1992, S. 17)

- Unterstützung der Anpassung des Modells an den aktuellen Arbeitskontext („explicit resolving contingencies“)
- Ermöglichung der flexiblen Anwendung bzw. dynamischen Veränderung des Modells während der Ausführung des Arbeitsablaufs („implicit and explicit coordination of predefined work“)
- Unterstützung zur Veränderung bzw. Neuerstellung von Modellen, um die Arbeit an veränderte organisationale Rahmenbedingungen anzupassen („rewriting arrangements“)
- Dokumentation und Kommunikation aller Veränderungen des Modells oder Abweichungen während der Ausführung (in allen Fällen)
- Unterstützung der Aushandlung eines gemeinsamen Verständnisses des Modells („ad-hoc explicit Articulation Work“)

Der zweite Aspekt, auf den näher eingegangen wird, ist jener der Kooperation mittels geteilter Informationsräume, also Ablagesysteme für Informationen, auf die mehrere Personen Zugriff haben. Als zentral wird hier die Unterstützung der Entwicklung einer einheitlichen Interpretation der verfügbaren Information gesehen³³. Die Autoren führen hier drei Funktionen an, die ein CSCW-System unterstützen muss:

- Unterstützung der Identifikation des Erstellers die Information
- Offenlegung der Kontexts der Information (im Sinne einer Relation zur Ursprungs- bzw. Anwendungsdomäne)
- Kontrolle über den Zugriff auf Information durch den Ersteller

Die genannten Aspekte zielen allesamt nicht auf die Unterstützung direkter Interaktion ab sondern ermöglichen im Wesentlichen durch die Erfassung von Metadaten eine Beurteilung von geteilt verfügbarer Information durch Individuen während diese damit arbeiten. Die „Articulation Work“, die hier unterstützt wird, ist durch die semi-automatisierte Erfassbarkeit bzw. die automatisierte Anzeige zumeist implizit.

Auswirkungen

Aufgrund der konzeptionellen Natur der Arbeit stehen keine Angaben zu etwaigen Auswirkungen der Maßnahmen zur Verfügung.

³³ „At the level of the objects themselves, shareability may not be a problem, but in terms of their interpretation, the actors must attempt to jointly construct a common information space which goes beyond their individual personal information spaces.“ (Schmidt und Bannon, 1992, S. 21)

Bewertung

Schmidt und Bannon (1992) sind die ersten, die sich mit der technischen Unterstützung von „Articulation Work“ in beliebigen Anwendungsdomänen und in all ihren Ausprägungen beschäftigen. Sie geben dabei Anforderungen („Was?“) an diese technische Unterstützung an, gehen jedoch nicht auf deren Umsetzung („Wie?“) ein.

Auf Anforderungsebene wird jedoch umfassend hinsichtlich unterschiedlicher Artikulationsbedürfnisse argumentiert, so dass die Arbeit als Grundlage zur Konzeption einer konkreten technischen Unterstützung von „Articulation Work“ herangezogen werden kann.

Kontext	Umsetzung von CSCW-Systemen (Workflow-Support, Shared Information Spaces, Cooperative Tools)
Art von AW	alle Arten und Ausprägungen von Articulation Work (Workflow Planung und Unterstützung), implizit (Information Sharing)
Unterstützung	<i>technisch</i> durch Maßnahmen, die sowohl die selbstgesteuerte Anpassung von Arbeitsabläufen an den aktuellen Arbeitskontext als auch die Entwicklung eines gemeinsamen Verständnis über den Arbeitsablauf und die verwendeten Arbeitsartefakte ermöglicht
Auswirkungen	—

2.4.4. Supporting articulation work using software configuration management systems

Grinter (1996) führt in dieser Arbeit die Ideen von Bendifallah und Scacchi (1987) und Schmidt und Bannon (1992) weiter und zeigt, wie die Softwareentwicklung als kooperativer Prozess durch computerbasierte Werkzeuge unterstützt werden kann.

Kontext

Die Autorin betrachtet die Rolle von „Articulation Work“ im Kontext der Softwareentwicklung und zeigt anhand zweier qualitativer empirischer Studien die Auswirkungen eines computerbasierten Configuration Management Systems bei der kooperativen Erstellung von Software.

Grinter beschäftigt sich mit zwei Arten von Aufgaben, die im Rahmen eines Software-Entwicklungs-Prozesses im Rahmen von „Articulation Work“ zu bewältigen sind. Einerseits ist die tägliche Arbeit abzustimmen, so dass sichergestellt ist, dass die gera-

de erstellte Software korrekt funktioniert. Andererseits muss sichergestellt werden, dass auch das gesamte Produkt als Einheit funktioniert. Während die erstere Herausforderung durch das in den Fallstudien eingesetzte Configuration Management System unterstützt wurde (im Rahmen von „resolving contingencies“, die sowohl implizit als auch explizit war), waren im zweiten Fall organisationale Maßnahmen wie die Bildung von Koordinations-Kommitees (also „explicit coordination of pre-defined work“) notwendig.

Unterstützung

In der Arbeit detailliert beschrieben sind die Eigenschaften und Auswirkungen des Configuration Management Systems, so dass hier nur auf dessen Unterstützungsleistung berücksichtigt werden kann. Konkret muss hier weiter auf jene Fälle von „resolving contingencies“ eingeschränkt werden, die von Problemen in Arbeitsabläufen ausgelöst werden, die auf konfliktionierende (Teil-)Ergebnisse der produktiven Arbeit zurückzuführen sind (hier: simultan editierter Source-Code, der nicht automatisiert zusammengeführt werden kann)

Während der Entwicklung von Software wird durch das Configuration Management System dargestellt ob bzw. durch wen ein bestimmter Teil des Source Codes zur Zeit bearbeitet wird. So kann die simultane und potentiell zu Konflikten führende Bearbeitung durch andere Personen vermieden werden. Diese Information muss weder explizit ins System eingepflegt noch abgerufen werden und ist deshalb als Unterstützung von impliziter „Articulation Work“ zu klassifizieren.

Sind bereits Konflikte aufgetreten, unterstützt das System die Auflösung derselben durch eine adäquate Darstellung der betroffenen Teile des Source Codes, so dass den zusammenarbeitenden Individuen eine Grundlage zur Erörterung und Auflösung des Konflikts zur Verfügung steht. Diese Darstellung muss „geeignet“ sein – welche Arten von Zusatzinformation und welche Form der Darstellung dies gewährleistet, ist domänenabhängig. Im konkreten Fall wird die zusätzliche Angabe einer Begründung für Änderungen in Source Code Teilen empfohlen, um bei der Auflösung der Konflikte fundiert argumentieren zu können.

Auswirkungen

Den Software Entwicklern wird die Freiheit gelassen, Konflikte durch sequentielle Bearbeitung zur vermeiden oder diese (z.B. aus Zeitdruck) bewusst in Kauf zu nehmen und deren Auflösung ggf. in einem separaten Schritt durchzuführen.

2. Articulation Work

Bewertung

Die Arbeit von Grinter (1996) geht auf einen Spezialfall von „resolving contingencies“ ein und zeigt, welche Unterstützungsleistung ein Software-Werkzeug bei der Ausgestaltung eines Arbeitsprozesses leisten kann, in dem sich die produktive Arbeit stark an Artefakten (hier: Source-Code) materialisiert. In derartigen Arbeits-Umgebungen erscheinen die vorgeschlagenen bzw. beschriebenen technischen Maßnahmen sinnvoll und führten in den angeführten Fällen offenbar zum Erfolg.

Kontext	Unterstützung von Software Entwicklung
Art von AW	„resolving contingencies“ implizit und explizit (zur Vermeidung bzw. zur Beseitigung von konfliktionierenden Arbeitsergebnissen)
Unterstützung	<i>resolving contingencies implizit:</i> Visualisierung aktuell durch andere Personen bearbeitete bzw. verwendete Arbeitsgegenstände; <i>resolving contingencies explizit:</i> adäquate Darstellung der konfliktionierenden Ergebnisse (Darstellung domänenabhängig)
Auswirkungen	Wahlfreiheit der interagierenden Individuen bei der Ausgestaltung ihres Arbeitsablaufs.

2.4.5. Coordination Mechanisms: Towards a Conceptual Foundation of CSCW Systems Design

Schmidt und Simone (1996) entwickeln in ihrer Arbeit ein generisches Vorgehen zur Konzeption von technischer Unterstützung von „Articulation Work“. Aufbauend auf früheren Arbeiten der Autoren (z.B. (Schmidt, 1990) und (Schmidt und Bannon, 1992)) formulieren die Autoren eine Spezifikationnotation für CSCW-Systeme, die auf der Unterstützung von „Articulation Work“ aufbauen.

Kontext

Die Arbeit führt den in früheren Arbeiten der Autoren bereits propagierten Ansatz der Konzeption von CSCW-Systemen zur Unterstützung von „Articulation Work“ fort. Die Herangehensweise ist stark technisch orientiert, begründet sich jedoch immer aus dem jeweiligen Anwendungskontext oder theoretischen Überlegungen aus den Arbeitswissenschaften.

Unterstützung

Die Autoren legen einen klaren Fokus auf die technische Unterstützung von „Articulation Work“ in der Form von CSCW-Systemen, für die sie eine Spezifikationsnotation entwickeln. Bevor sie jedoch im Detail auf deren Entwicklung eingehen, führen sie – als Grundlage für die weitere Entwicklung und als historischen Kontext – etablierte organisationale Maßnahmen zur Unterstützung von „Articulation Work“ an.

Bei den angeführten organisationalen Maßnahmen führen die Autoren die Nützlichkeit von Artefakten, die für ihren Anwendungsbereich die notwendige Durchführung von „Articulation Work“ formalisieren und definieren und damit die Komplexität der alltäglich Interaktion reduzieren³⁴. Beispiele für derartige Artefakte sind etwa Checklisten, Formulare zur Meldung von Fehlern o.ä.. Die Artefakte sind dabei immer Teil eines „Koordinationsmechanismus“, der den Umgang mit einer aufgetretenen Situation oder einer Aufgabe beschreibt. Neben dem Artefakt enthält dieser Mechanismus auch ein „Koordinationsprotokoll“, dass den Ablauf bzw. das Vorgehen und den Umgang mit dem Artefakt beschreibt. Diese Protokolle sind dabei nicht als strikt vorgegebenen Ablaufmuster zu verstehen sondern sollen Individuen ermöglichen, in Standardfällen ohne Planungsaufwand ein fixes Vorgehen zur Verfügung zu haben und bei Bedarf dieses an den jeweiligen Fall anpassen zu können (in Sinne von „situated action“ (Suchman, 1987))³⁵. Die Rolle des Artefakts ist dabei die „Vergegenständlichung“ des Protokolls und damit die Erhöhung der Zugänglichkeit desselben. Außerdem repräsentiert das Artefakt während der Durchführung der „Articulation Work“ deren aktuellen Status und Fortschritt. Aus all diesen Ausführungen kann abgeleitet werden, dass sich die Autoren immer auf explizite „Articulation Work“ im Arbeitsablauf beziehen.

Der eigentliche Schwerpunkt der Arbeit ist jedoch die technische Unterstützung von „Articulation Work“. Aufbauend auf den zuvor ausgeführten organisationalen Maßnahmen wird gezeigt, wie diese durch technische Mittel umgesetzt und unterstützt bzw. verbessert werden können. Die Grundidee besteht darin, sowohl die Artefakte als auch Teile der Protokolle in den Rechner zu transferieren und durch die digitale Repräsentation die Flexibilität zu gewinnen, die bei „coordination of pre-

³⁴, „Faced with a high degree of complexity of articulation work, cooperating actors typically use a special category of artifacts which, in the context of a set of conventions and procedures, stipulate and mediate articulation work and thereby are instrumental in reducing its complexity and in alleviating the need for ad hoc communication.“ (Schmidt und Simone, 1996, S. 159)

³⁵, „As a generalization, we find that a protocol stipulates the articulation of distributed activities by conveying affordances and constraints to the individual actor which the actor, as a competent member of the particular ensemble, can apply without further contemplation and deliberation unless he or she, again as a competent member, has accountable reasons not to do so.“ (Schmidt und Simone, 1996, S. 173)

2. Articulation Work

defined work“ im konkreten Arbeitsablauf notwendig ist. Die grundlegenden Anforderungen an ein technisches System, das „Articulation Work“ unterstützt, sind deshalb auch die *Formbarkeit* der Koordinationsmechanismen („malleability“, also die Anpassbarkeit an den aktuellen Arbeitskontext), sowie die *Verknüpfbarkeit* der Koordinationsmechanismen untereinander („linkability“) um die in komplexen Arbeitssituationen notwendige Durchführung von mehreren voneinander abhängigen Koordinationsmechanismen zu unterstützen.

Diese Forderungen werden im Rahmen des „Ariadne“-Ansatzes umgesetzt, der in (Schmidt und Simone, 1996) konzeptuell beschrieben ist. Die tatsächliche Implementierung von „Ariadne“ wird unter anderem in (Divitini und Simone, 2000) und (Sarini, 2003) beschrieben und ist Gegenstand von Abschnitt XY.

Auswirkungen

Da keine konkrete Anwendung des Systems beschrieben wird (bzw. dessen Implementierung zum Zeitpunkt der Erstellung der Arbeit nicht abgeschlossen war), können hier keine praktischen Auswirkungen des Ansatzes angegeben werden.

Generell ist zu erwarten, dass durch die Einführung von Koordinationsmechanismen, der Aufwand für „coordination of predefined work“ sinkt bzw. deren Komplexität reduziert wird. Durch die technische Unterstützung ist es möglich, die notwendige Flexibilität zu wahren, um die spezifizierten Koordinationsmechanismen an den aktuelle Arbeitskontext anzupassen sowie den gesamten Verlauf eines „Articulation Work“-Prozesses durch die Verknüpfbarkeit mehrerer Koordinationsmechanismen und dabei der Übernahme der jeweils relevanten Kontextinformation nahtlos zu unterstützen.

Bewertung

Schmidt und Simone (1996) zeigen umfassend die Unterstützung von expliziter „Articulation Work“ unmittelbar in Arbeitsabläufen – die vorgeschlagenen Konzepte können jedoch ebenso auf „Articulation Work“ im Vorfeld von Arbeitsabläufen zu deren Konzeption angewandt werden. Die vorgestellte technische Unterstützung setzt die organisationalen Maßnahmen um und ermöglicht die Realisierung von flexiblen, an den jeweiligen Kontext angepassten bzw. anpassbaren Werkzeugen.

Vor allem die Konzeptualisierung von Koordinationsmechanismen in Protokolle und Artefakte ist ein wesentlicher Beitrag zum Verständnis der Koordination von Individuen in Arbeitsabläufen und ein Ansatzpunkt zur Reduktion der Komplexität von Abstimmungsprozessen bzw. deren Vermeidung.

Die formulierten Anforderungen an ein technisches System („malleability“ und „linkability“) erscheinen vor allem für den unmittelbaren Einsatz in Arbeitsprozessen als wesentliche Erfolgskriterien für die praktische Einsetzbarkeit eines derartigen Systems.

Kontext	Umsetzung von CSCW-Systemen
Art von AW	explizit, im Regelfall im Arbeitsablauf
Unterstützung	<i>organisational</i> durch Koordinationsmechanismen, die aus einem vorgeschlagenen Vorgehen („Protokoll“) und einem zugehörigen „Artefakt“ (als Repräsentant des Protokolls und zur Dokumentation) bestehen; <i>technisch</i> durch Umsetzung der Koordinationsmechanismen im Rechner mit dem Ziel, die geforderte Flexibilität und Verknüpfbarkeit derselben zu gewährleisten
Auswirkungen	Reduktion der Komplexität der „Articulation Work“, Flexibilität bei der Durchführung von „expliziter Articulation Work“

2.4.6. Taking Articulation Work Seriously: An Activity Theoretical Approach

Neben den in Abschnitt 2.2.1 bereits beschriebenen konzeptionellen Arbeiten zu „Articulation Work“ enthält die Arbeit von Fjuk et al. (1997) auch Ansätze zur Unterstützung von „Articulation Work“.

Kontext

Fjuk et al. (1997) beschäftigen sich mit der technischen Unterstützung von „Articulation Work“ mittels CSCW-Systemen. Ihre Ansätze leiten sie aus der „Activity Theory“ ab, die ihre Wurzeln in der Psychologie (Leont'ev, 1972) bzw. in ihren erweiterten Variante in den Arbeitswissenschaften (Engeström, 2000) hat.

Unterstützung

Entsprechend den in Abschnitt 2.2.1 bereits beschrieben unterschiedlichen Arten von Arbeit, in denen „Articulation Work“ auftreten kann (individuelle Arbeit, lose und eng gekoppelte kooperative Arbeit), unterscheiden die Autoren auch bei der Betrachtung der Unterstützung von „Articulation Work“ nach diesen drei Gruppen.

2. Articulation Work

Im Falle individueller Arbeit kann „Articulation Work“ auf abstrakter Ebene („action within activity“) durch die automationsgestützte Verwaltung von Arbeitsorganisationswerkzeugen wie Kalendern und Todo-Listen unterstützt werden. Die „Articulation Work“ auf konkreter Ebene („operation within action“) kann – im Falle der computergestützten Abwicklung der zugehörigen „Production Work“ insofern unterstützt werden, als das die Benutzungsschnittstelle so an die Arbeitsdomäne angepasst ist, dass die Notwendigkeit der Beschäftigung mit den technischen Eigenschaften und der Bedienung des Systems geringer wird.

Bei lose gekoppelter kooperativer Arbeit zielt die geforderte Unterstützung von „Articulation Work“ auf die Organisation der Arbeitsteilung ab. Dabei muss unter anderem die Zuordnung von organisationalen Rollen zu beteiligten Individuen, die Erteilung von Zugriffsrechten sowie die Festlegung von Schnittstellen zwischen den Beteiligten berücksichtigt werden. Während des Arbeitsprozesses ist die Verfügbarkeit von expliziten Kommunikationskanälen und Mitteln zur Teilung von Daten notwendig.

Bei eng gekoppelter kooperativer Arbeit steigen die Anforderungen an Werkzeuge zur Unterstützung von „Articulation Work“ insofern noch weiter, als dass hier zusätzlich die Aushandlung der Durchführung von konkreten Arbeitsschritten sowie die Abstimmung untereinander in synchroner Zusammenarbeit unterstützt werden muss.

Auswirkungen

Die Arbeit beschreibt die Unterstützung von „Articulation Work“ auf rein konzeptioneller Ebene und führt keinerlei Auswirkungen derselben auf den Arbeitsprozess an.

Bewertung

Die Vorschläge der Autoren stehen weitgehend in Einklang mit den Ausführungen von (Schmidt und Simone, 1996) (siehe oben). Dies ist insofern bemerkenswert, als dass die Ableitung der Maßnahmen auf Basis der „Activity Theory“ erfolgt, auf die in der anderen Arbeit nicht Bezug genommen wurde.

Fjuk et al. formulieren Anforderungen an die technische Unterstützung von „Articulation Work“ (in allen Kontexten, vorrangig explizit, in Teilespekten auch implizit) und führen exemplarisch Werkzeuge an, mit denen diese erfüllt werden können. Sie geben jedoch (im Gegensatz zu (Schmidt und Simone, 1996)) kein Konzept an, wie die Unterstützung von „Articulation Work“ allgemein (d.h. unabhängig von jeweiligen Anwendungsfällen) realisiert werden kann.

Kontext	Umsetzung von CSCW-Systemen auf Basis der Activity Theory
Art von AW	alle
Unterstützung	durch computerunterstützte Werkzeuge zur Arbeitsorganisation, Kommunikation und Aushandlung von Zusammenarbeit
Auswirkungen	keine angeführt

2.4.7. TeamSpace: an environment for team articulation work and virtual meetings

(Fuchs et al., 2001) beschreiben ein technisches System, dass die Zusammenarbeit in Gruppen durch Unterstützung von „Articulation Work“ erleichtern soll.

Kontext

Die Autoren beschreiben die technische Unterstützung von „Articulation Work“ in (verteilten) Gruppen mittels CSCW-Technologie. Ihr Anwendungsbereich ist die Softwareentwicklung, in deren Kontext die Teams von Programmierern, die geographisch verteilt sind, bei deren Zusammenarbeit unterstützen. Im Rahmen dieses Anwendungsbereichs wurde auch eine Studie zur Erhebung der Anforderungen an die technische Unterstützung durchgeführt.

Unterstützung

(Fuchs et al., 2001) präsentieren ein konkret umgesetztes System, das eine Reihe von Werkzeugen zur Unterstützung von „Articulation Work“ bietet:

- „Task structured workspace“ Werkzeug zum Aufgabenmanagement und zur geteilten Bearbeitung von Informations-Objekten. Dient außerdem als Container für die übrigen Werkzeuge.
- „Place-based adaptation to work modes“ Die Autoren unterscheiden unterschiedliche zu unterstützende „work modes“ („work“, „meeting“, „social“) und unterstützen diese unterschiedlich, indem die Benutzungsschnittstelle an den jeweiligen Modus und dessen Artikulations-Anforderungen adaptiert wird.
- „Synchronous and asynchronous communication and awareness“ Werkzeug zur Planung und Durchführung von virtuellen Meetings (inkl. Audio- und Video-Übertragung) sowie Anzeige des aktuellen Tätigkeits- und Verfügbarkeitsstatus von Mitarbeitern.

2. Articulation Work

Auswirkungen

Das System wurde zum Zeitpunkt der Erstellung des Artikels nicht operativ eingesetzt. Die Autoren treffen auch keine expliziten Aussagen zu den erwarteten Effekten des Systems.

Bewertung

Die Autoren gehen sehr konkret auf einzelne Werkzeuge zur Unterstützung von „Articulation Work“ ein (ohne diese im Detail abzugrenzen und die zu unterstützenden Aspekte zu motivieren). Sie beziehen sich auf (Schmidt und Bannon, 1992) und unterstützen auch die dort genannten wesentlichen Aspekte von „Articulation Work“. Die konzeptionelle Aussagekraft ist ob der Fokussierung auf eine konkrete Implementierung eher gering einzuschätzen.

Domäne	Umsetzung von CSCW-Systemen zur Unterstützung von Software-Entwicklung
Art von AW	implizite und explizite „coordination of predefined work“, explizites „ad-hoc alignment“
Unterstützung	<i>technisch</i> Werkzeuge zur aufgaben- und kontextorientierten Anpassung der Benutzungsschnittstelle, Werkzeuge zur Kommunikation und zur Herstellung von Awareness
Auswirkungen	keine angeführt

2.4.8. Supporting different dimensions of adaptability in workflow modeling

Divitini und Simone (2000) stellen ein System zur Unterstützung von etablierter kooperativer Arbeit in Form eines adaptiven Workflow-Systems vor, dessen Verhalten durch die Durchführung von „Articulation Work“ beeinflusst werden kann bzw. die Durchführung derselben unterstützt.

Kontext

Die Autoren bauen in ihrer Arbeit auf die in (Schmidt und Simone, 1996) vorgestellten Konzepte auf und konkretisieren den Aspekt der Koordination von etablierten Arbeitsabläufen („coordinating predefined work“). Dabei nehmen sie Bezug auf die Verwendung von Workflow-Management-Systemen und integrieren die in diesem Bereich etablierten Konzepte mit dem „Ariadne“-Ansatz aus (Schmidt und Simone,

1996). Die Unterstützung von „Articulation Work“ erfolgt damit klar mit technischen Hintergrund.

Unterstützung

An dieser Stelle werden lediglich jene Unterstützungs-Maßnahmen beschrieben, die über die in der Besprechung von (Schmidt und Simone, 1996) bereit genannt wurden hinausgehen. Die Autoren schlagen vor, verteilte Arbeit bzw. deren Abstimmung anstatt mit WfMS³⁶ mit „Computational Coordination Mechanisms“ zu unterstützen. Diese Koordinations-Mechanismen sind an den jeweiligen Anwendungsfall angepasst computerbasierte Werkzeuge, die die Abstimmung der beteiligten Individuen während der Durchführung eines Arbeitsablaufs unterstützen sollen. Wesentlich ist, dass das hier vorgeschlagene Konzept die domänenabhängige Spezifikation dieser „Computational Coordination Mechanisms“ umfasst bzw. auf diesem basiert.

Aufbauend auf den in Abschnitt 2.3 bereits beschriebenen abzustimmenden Arbeitsaspekten (hier: „Categories of Articulation Work“) können abhängig vom abzustimmenden Aspekt Sprachen entworfen bzw. existierenden Sprachen auf diese Aspekte abgebildet werden. Das ermöglicht eine domänengerechte Spezifikation von „Computational Coordination Mechanisms“ und damit die Unterstützung von „Articulation Work“ bereits in der Phase der Planung eines Arbeitsablaufs.

Das so spezifizierte System unterstützt in der Folge die Ausführung von Arbeitsabläufen. Es erlaubt auch (gesteuert durch ein festgelegtes Berechtigungssystem) unterschiedlich starke, temporäre oder permanente Änderungen des spezifizierten Koordinationsmechanismus.

Auswirkungen

Die Autoren machen keine Angaben über eine konkrete Anwendung des Ansatzes in der Praxis. Durch die flexible Spezifikationsmöglichkeit der Koordinationsmechanismen ist laut den Autoren eine reduzierte Komplexität bei der Spezifikation bzw. die exaktere Abbildung der relevanten Information möglich³⁷.

Bewertung

³⁶Workflow-Management-System

³⁷„[...] overcome some of the limits of almost all current approaches to process modeling [...]: either a restricted language focusing on a specific type of representation or a language comprehensive of all potentially needed features, requiring the user to manage an overwhelming complexity.“(Divitini und Simone, 2000, S. 377)

2. Articulation Work

Die vorliegende Arbeit nimmt erstmals auf die Rolle von Modellen bei der technischen Unterstützung von „Articulation Work“. Die Autoren argumentieren, dass die Adäquatheit der Repräsentationsform von Information im Rahmen expliziter „Articulation Work“ ein wesentliches Erfolgskriterium ist. Obwohl die konkrete technische Umsetzung nicht beschrieben ist, ist diese konzeptionelle Anforderung für das weitere Vorgehen berücksichtigenswert.

Kontext	Integration von CSCW-Konzepten mit WfMS-Ansätzen
Art von AW	„working out original arrangements“, explizite „coordination of predefined work“, explizites „resolving contingencies“ sowie „reworking arrangements“
Unterstützung	methodisch: Auswahl bzw. Definition einer adäquaten Modellierungssprache, um Koordinationsmechanismen zu spezifizieren, technisch: Möglichkeit der Ausführung und Anpassung dieser Koordinationsmechanismen
Auswirkungen	Vereinfachte Spezifikation der Koordinationsmechanismen, flexible Koordination bei der Ausführung des Arbeitsablaufs

2.4.9. Mundane knowledge management and microlevel organizational learning: An ethological approach

Davenport (2002) beschreibt „Articulation Work“ als eine Form von „alltäglichem Wissensmanagement“, mit Hilfe dessen beteiligte Individuen im Arbeitsprozess lernen und ihre Kompetenzen erweitern („situated learning“).

Kontext

Davenport (2002) betrachten „Articulation Work“ aus einer Lern-Perspektive (unter Bezugnahme auf „situated learning“ (Lave und Wenger, 1991)) und bezeichnen es als alltägliches Wissensmanagement („mundane knowledge management“). Sie führen in weiterer Folge aus, inwiefern „situated learning“ in geographisch verteilten Gruppen durch Informationstechnologie unterstützt werden kann und beziehen sich dabei auf das Konzept der „Communities of Practice“ (Wenger, 1999).

Die Autorin verfolgt dabei eine enge Auffassung von „Articulation Work“ und bezieht sich lediglich auf „routine interaction“ in Gruppen von Individuen.

Unterstützung

Anhand einer Fallstudie identifiziert die Autorin die unterschiedlichen Handlungsphasen, die im Rahmen eines Lernprozesses im Rahmen von „Articulation Work“ auftreten und zeigt, wie diese im konkreten Fall unterstützt wurden. Wesentlich ist hier nicht die konkrete technische Unterstützung, die sich auf die Bereitstellung einer Kommunikation-Plattform beschränkte, sondern der verfolgte methodische Ansatz, die auf die Konzepte der „Communities of Practice“ zurückgreift.

Während der eigentliche Arbeitsdurchführung arbeitet die betreffende Gruppe von Individuen selbstgesteuert und koordiniert sich selbst. In Fällen, in denen ein Problem auftritt („resolving contingencies“), erweist sich das Eingreifen eines „facilitators“ im Sinne von (Wenger, 1999) als hilfreich. Dieser übernimmt die Aufgabe, die durchzuführende „Articulation Work“ zu koordinieren und zu moderieren, so dass die aufgetretene problematische Situation gelöst werden kann. Die Rolle des „facilitators“ kann dabei auch dynamisch von einem bereits beteiligten und in dieser Hinsicht kompetenten Individuum übernommen werden und muss nicht permanent besetzt werden. „Kompetent“ bedeutet in diesem Zusammenhang, die eingesetzten technischen Werkzeuge zu beherrschen, Domänenwissen zu haben und Kompetenz in der Extraktion und Zusammenfassung von Wissen der einzelnen beteiligten Individuen zu haben.

Auswirkungen

In der Arbeit sind keine konkreten Ergebnisse der getroffenen Maßnahmen angeführt. Die Generalisierbarkeit der aus der Fallstudie abgeleiteten Maßnahmen wird explizit nicht behandelt und bleibt offen.

Bewertung

Die Arbeit führt das Konzept der „Communities of Practice“ mit jenem der „Articulation Work“ zusammen. Die Autorin zeigt, dass die nicht formalisierte Interaktion in Communities ein geeignetes Mittel zu sein scheint, um zumindest die einfacheren (im Sinne von „unproblematischen“) Fälle von „Articulation Work“ zu bewältigen. Inwieweit sich diese Organisationsform auch für komplexere Problemfälle eignet, bleibt offen.

Die Arbeit zeigt jedoch als eine von wenigen Ansätzen einen nicht durch Technik getriebenen Weg der Unterstützung von „Articulation Work“ auf und ist aus diesem Grund berücksichtigenswert.

2. Articulation Work

Kontext	„situated learning“, Wissensmanagement, „Communities of Practice“
Art von AW	„coordinating predefined work“ & „resolving contingencies“
Unterstützung	durch die organisationale und technische Unterstützung von „Communities of Practice“
Auswirkungen	—

2.4.10. Modelling Cooperative Work: Chances and Risks of Structuring

Herrmann et al. (2002) beschäftigen sich mit Modellen von soziotechnischen Arbeitsprozessen und zeigen auf, dass zu deren (kooperativen Erstellung) „Articulation Work“ notwendig ist.

Kontext

(Herrmann et al., 2002) beschreiben die (kooperative) Bildung von diagrammatischen Modellen in sozio-technischen Systemen und deren Auswirkung auf den realen Arbeitsablauf. Sie beziehen sich dabei nur am Rande auf „Articulation Work“ (im Kontext der CSCW-Forschung von (Schmidt und Bannon, 1992)) und bezeichnen damit jene Vorgänge, die notwendig sind, um die individuellen Sichtweisen der am Arbeitsablauf beteiligten Personen abzustimmen und in einem gemeinsamen Modell zu repräsentieren („articulating and negotiating“ – siehe Abbildung 2.7). In anderen Publikationen der Autoren wird ein Modellierungswerkzeug (Herrmann et al., 2004a) und eine Methode (Herrmann et al., 2004b) beschrieben, das diesen Vorgang unterstützen soll.

Unterstützung

Im Rahmen mehrerer Fallstudien zeigen die Autoren, dass die Verwendung von diagrammatischen Modellen als externalisierte Repräsentation von individuellen Sichten auf Arbeitsabläufe bei deren Abstimmung und Verbesserung hilfreich ist. Externalisierte Modelle bringen jedoch auch Risiken mit sich, die durch eine adäquate Unterstützung des Modellierungsprozesses gemindert werden müssen. Konkret erwähnen die Autoren die schwierige Veränderbarkeit einmal niedergeschriebener Arbeitsabläufe, die potentiell fehlerhafte oder unzureichende Abbildung des Arbeitsablaufs im Modell sowie (im Kontext der kooperativen Erstellung) jene Unzulänglichkeiten des Modells, die durch bewusst falsch oder nicht offengelegte Arbeitsaspekte aus „politischen“ Gründen entstehen.

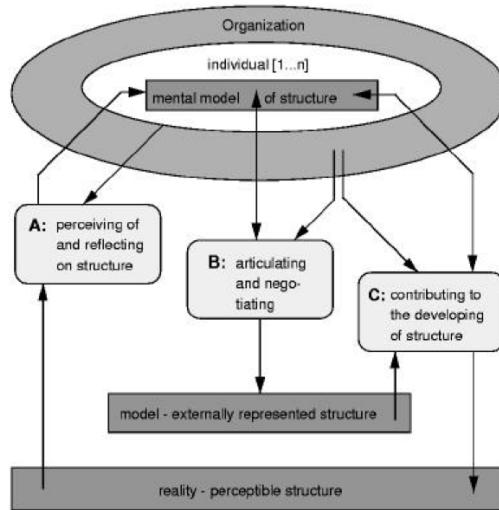


Abbildung 2.7.: Mentale Modelle im Kontext der Arbeitsmodellierung (entnommen aus (Herrmann et al., 2002))

In der Folge entwickeln Herrmann et al. Anforderungen an einen Modellierungsprozess sowie eine Modellierungssprache, die diese Risiken so weit möglich vermeiden und „Articulation Work“ unterstützen:

- Modelle müssen „aus dem Arbeitsablauf heraus“ entstehen, also von den unmittelbar betroffenen Personen entwickelt werden und dürfen nicht von „außerhalb“ übergestülpt werden.
- Die Modellierungssprache muss es erlauben, die Dynamik der abgebildeten Strukturen darzustellen.
- Sie muss sowohl zur Abbildung (präskriptiver) exakter Arbeitsabläufe („scripts“) als auch zur Abbildung des (Orientierung gebenden) Arbeitskontext („maps“) geeignet sein.
- Detaillierungs- und Abstraktionsgrad der Abbildung müssen frei wählbar sein.
- Information über den Modellierungsverlauf selbst muss im Modell abbildbar sein.
- Die Modellierungssprache muss die Abbildung beliebiger Aspekte von Arbeit (nicht nur Aktivitäten und Ressourcen, sondern z.B. auch Rollen oder Kompetenzen) erlauben.
- Der Modellierungsvorgang muss durch ein flexibles Werkzeug unterstützt werden, dass die Verwendung der Unterstützungsfunktionalitäten ermöglicht aber nicht erzwingt.

2. Articulation Work

- Das Werkzeug sollte den Export der Modelle in beliebige Datenformate ermöglichen, um deren Weiterverarbeitbarkeit zu gewährleisten.

Auswirkungen

Anhand der Fallstudien zeigen die Autoren, dass das ein Vorgehen nach den beschriebenen Richtlinien sowie eine entsprechende Werkzeugunterstützung bei der Abstimmung von kooperativen Arbeitsabläufen hilfreich ist (siehe dazu auch (Herrmann et al., 2000)).

Bewertung

Die Arbeit kann als ein Beitrag zur Unterstützung von „Articulation Work“ im Sinne von „making agreements“ verstanden werden, auch wenn dies nicht explizit so benannt wird. In diesem Zusammenhang führen die Autoren die Nützlichkeit von externalisierten Modellrepräsentation bei der Abstimmung und Aushandlung von Arbeitsabläufen an. Für diese Form der Unterstützung werden konkrete Anforderungen sowohl methodischer als auch technischer Natur gegeben, was diese Arbeit zu einer wertvollen Quelle für die weiteren Ausführungen in der vorliegenden Arbeit macht.

Kontext	Modellierung von Arbeit,
Art von AW	explizites „agreement making“
Unterstützung	flexible Modellierung von Arbeitsabläufen unter Einbeziehung der unmittelbar beteiligten Individuen
Auswirkungen	Erleichterung bzw. Ermöglichung der Abstimmung oder Aushandlung von kooperativen Arbeitsabläufen

2.4.11. Recursive Articulation Work in Ariadne: The Alignment of Meanings

(Sarini und Simone, 2002a) beschäftigen sich mit „recursive Articulation Work“, also jener Form, deren Gegenstand selbst wiederum „Articulation Work“ ist. Die Autoren leiten Anforderungen an die Unterstützung dieser Form von „Articulation Work“ ab und zeigen die konkrete Umsetzung als Teil des „Reconciler“-Systems.

Kontext

Die Arbeit steht in der Tradition der oben bereits beschriebenen Arbeiten (Schmidt und Bannon, 1992) und (Divitini und Simone, 2000) und beschäftigt sich mit der

technischen Unterstützung von „Articulation Work“ durch CSCW-Systeme. Konkret wird auf jene Fälle von „Articulation Work“ eingegangen, wo „alignment of meaning“ (also die Abstimmung der individuellen Verständnisse der Arbeitsdomäne) notwendig ist. Es ist kein spezifischer Anwendungsbereich angeführt, das Konzept hat den Anspruch, generisch anwendbar zu sein.

Unterstützung

Die Arbeit behandelt zwei unterschiedliche Phasen von „Articulation Work“: jene, in denen die beteiligten Individuen die auftretende Probleme abgrenzen, detailliert spezifizieren und (Teil-)Lösungen aushandeln („agreement making“) und jene, in denen Koordinationsprobleme durch die gesammelte Information bereits vor dem Auftreten verhindert werden („coordination of predefined work“). Während zweitere Unterstützung an dieser Stelle nicht näher behandelt wird, da es sich bei der vorgeschlagenen Implementierung lediglich um ein Vorschlagssystem für Begriffe an der Benutzungsschnittstelle handelt, muss erstere Unterstützungsform näher betrachtet werden.

Der dabei angestossene Unterstützungsprozess ist eine Koordinationsmechanismus im Sinne von (Divitini und Simone, 2000). Im Rahmen der Ausführung desselben wird ein „Reconciliation Artifact“ generiert und enthält eine Abbildung der individuellen Begriffswelten der beteiligten Individuen, die untereinander in Korrespondenz gesetzt werden. Wie die Externalisierung der Begriffe sowie deren Assoziation untereinander ablaufen, schränkt der Ansatz bewusst nicht ein bzw. legt sich nur vorläufig fest³⁸. Das „Reconciler“-System unterstützt dabei die Erfassung und Verwaltung der offengelegten Begriffe. Wie es den Artikulationsprozess konkret unterstützt, ist in (Mark et al., 2002a) angeführt, hier aber nicht Gegenstand einer detaillierteren Betrachtung, da vorrangig technische Implementierungsdetails beschrieben wurden.

Auswirkungen

In der vorliegenden Arbeit werden keine Aussagen hinsichtlich der tatsächlichen Auswirkungen der Unterstützung getroffen. Auch Mark et al. (2002a) kommen auf Basis einer empirischen Studie zu dem Schluss, dass die Unterstützungsleistung nicht allgemein nachgewiesen werden kann und ist stark von der individuellen Nutzungs-bereitschaft abhängig.

³⁸ „For sake of testing the integration we are aiming at, we defined the simplest protocol: all the users involved in the reconciliation process can communicate among themselves to define the correspondences, while a single Actor assumes the Role of Manager of the Reconciliation Artifact and is in charge of keeping it updated.“ (Sarini und Simone, 2002a, S. 10)

2. Articulation Work

Bewertung

Die Autoren gehen auf die Unterstützung jenes Teils von „Articulation Work“ ein, der zur Abstimmung der individuellen Sichten auf den Arbeitsablauf und der jeweiligen Begriffswelten eingesetzt wird. Auch wenn sie in der konkreten methodischen Umsetzung vage bleiben, ist erstmals eine explizite Beschäftigung mit der Unterstützung dieser Phase vorhanden.

Kontext	CSCW
Art von AW	explizites „agreement making“, „coordination of predefined work“
Unterstützung	durch die Abstimmung der individuellen Sichten auf eine Domäne, der Offenlegung des Vokabulars in einem geteilten Artefakt und dessen Verwendung während der Arbeit
Auswirkungen	—

2.4.12. Combining Communication and Coordination Toward Articulation of Collaborative Activities

Raposo et al. (2004) decken in ihrer Arbeit zur (technischen) Unterstützung kooperativer Arbeit explizit alle Zeitpunkte, in denen „Articulation Work“ auftreten kann, ab („pre-articulation“, „coordination“, „post-articulation“).

Kontext

Die Autoren schlagen eine technische Unterstützung von „Articulation Work“ mittels CSCW-Werkzeugen vor. Sie fassen dabei das Konzept „Articulation Work“ sehr weit und inkludieren neben der Planungs- und Durchführungsphase eines Arbeitsablaufs auch dessen Nachlauf („post-articulation“), in dem die durchgeführte Koordination reflektiert wird. Zur Unterstützung verfolgen die Autoren einen formalisierten Ansatz, der auf Konzepten der theoretischen Informatik aufbaut. Das Anwendungsgebiet ihres Ansatzes schränken die Autoren nicht ein und berücksichtigen explizit lose wie auch eng gekoppelte kooperative Arbeitsabläufe mit beliebig vielen beteiligten Individuen.

Unterstützung

Je nach Phase der „Articulation Work“ unterscheiden die Autoren zwischen unterschiedlichen Arten der Unterstützung. Für die „pre“ und „post-articulation“-Phase wird die Verwendung von „conversation clichés“ zur Aushandlung von „commitments“ vorgeschlagen. Während der „coordination“-Phase werden die „commitments“ automationsgestützt zur Anwendung gebracht.

„Conversation clichés“ sind im Wesentlichen Zustandsautomaten, die den Rahmen einer Konversation zwischen Individuen vorgeben. Sie dienen jeweils der Erreichung einer bestimmten Art von „commitment“. „Commitments“ sind logische Konstrukte, in denen die individuelle Zustimmung oder Ablehnung zu Aussagen ausgedrückt werden kann, die die Koordination beschreiben. Weiters ist es möglich, explizit kein „commitment“ zu einer Aussage abzugeben. Auf Basis dieser „commitments“ bestimmt ein Algorithmus die Modalitäten der Zusammenarbeit.

Während der „coordination phase“ kommt ein „task/interdependency-model“ zum Einsatz, das im Wesentlichen einem auf die „commitments“ abgestimmten Aktivitätsdiagramm entspricht. In diesem Modell werden letztlich die durchzuführenden Aufgaben und deren gegenseitige Abhängigkeiten definiert. Während der Ausführung kann das System dann auf diese Abhängigkeiten hinweisen bzw. die notwendigen Koordinationsmechanismen auslösen.

Auswirkungen

Konkrete Auswirkungen werden in der Publikation nicht angeführt. Die Autoren erwähnen die Simulierbarkeit und Verifizierbarkeit der erstellten Modelle als Vorteil der formalen Repräsentation.

Bewertung

Die Autoren verfolgen einen – im Gegensatz zu anderen Arbeiten – stark regulatorischen Ansatz zur Unterstützung von „Articulation Work“. Die vorgeschlagenen Mechanismen zur Abstimmung und Koordination von kooperativer Arbeit versuchen, den Arbeitsverlauf durch computergestützte Vorgaben in einem definierten Rahmen zu halten und so das Auftreten von problematischen Situationen zu verhindern.

Der Vorteil dieses eher restriktiven Vorgehens liegt in der Formalisierbarkeit der Interaktionsmuster und Arbeitsabläufe und der damit verbundenen Verifizierbarkeit und Simulierbarkeit der erstellten Modelle.

2. Articulation Work

Kontext	CSCW, Formale Modelle von Interaktion
Art von AW	explizites „making arrangements“, explizite „coordination of predefined work“
Unterstützung	„making arrangements“: Aushandlungsunterstützung durch „Conversation Clichés“, „coordination“ durch die Ausführung der ausgehandelten Aufgabenmodelle
Auswirkungen	formale Verifizierbarkeit und Simulierbarkeit

2.4.13. Interactive Process Models

Jørgensen (2004) beschreibt die Verwendung von „interaktiven“ Prozessmodellen in organisationalen Arbeitsprozessen und die Veränderung dieser Prozesse durch Modellierungsvorgänge. Dabei bezeichnet er den Modellierungsvorgang als „Articulation Work“.

Kontext

Im Gegensatz zu den anderen hier behandelten Arbeiten steht in (Jørgensen, 2004) nicht die Unterstützung der Koordination von Arbeitsprozessen im Vordergrund. Die Arbeit beschäftigt sich mit Prozessmodellen (also ablauforientierten Modellen von Arbeit) und sieht „Articulation Work“ als jene Tätigkeit, die zur Erstellung dieser Modelle notwendig ist. Die Rückwirkung der Modelle auf die reale Arbeitswelt wird als „Model Activation“ bezeichnet. Modellierung ist hierbei eine Aktivität, die von den betroffenen Personen selbst ausgeführt wird, die erstellten Modelle bilden dementsprechend die individuelle Sicht auf eine Arbeitsablauf ab. Insofern ist der vorgestellte Ansatz tatsächlich als eine Form von „making arrangements“ (als Variante von expliziter, planender „Articulation Work“) zu sehen.

Unterstützung

Jørgensen führt Anforderungen an eine Modellierungssprache an, die „Articulation Work“ unterstützt. Diese sollte:

- einfach sein und nur wenige Basiselemente enthalten. Elemente der realen Welt sollten eindeutig in das Modell abgebildet werden können. Die Aussage des Modells sollte möglichst intuitiv erschließbar sein.
- visuell bzw. graphisch darstellbar sein. Die Darstellung sollte sowohl einen Überblick über das Gesamtmodell als auch detaillierte Ansichten einzelner Ausschnitte ermöglichen.

- Konstrukte enthalten, die auf die jeweilige Anwendungsdomäne der Modellierenden abgebildet bzw. angepasst werden können.
- nicht statisch hinsichtlich der Bedeutung der verwendeten Modellelemente sein. Die Bedeutung der Modellelemente kann sich während der Erstellung der Modelle durch Aushandlung oder Reflexion verändern.

Der Autor führt zur Erfüllung dieser Anforderungen eine Modellierungssprache ein, die den Ansatz des „semantic holism“ verfolgt. In Sprachen, die den Ansatz des „semantischen Holismus“ berücksichtigen, haben Modellelemente keine eindeutige, fix vorgegebenen Bedeutung. Vielmehr erschließt sich der Bedeutung der einzelnen Elemente erst aus dem Gesamtzusammenhang des Modells, also aus dem Zusammenspiel aller Elemente.

Auch im Feld der „coordination of predefined work“ liefert Jørgensen (2004) Ansätze zur Unterstützung von Articulation Work, indem er die erstellten Modelle semi-automatisiert („interactive“) ausführbar macht („activation“).

Auswirkungen

Der Autor führt keine konkrete Evaluierung seines Ansatzes an, weshalb keine Aussagen zu den Auswirkungen des Ansatzes in der Praxis gemacht werden können. Aus der zugrunde liegenden Literatur leitet er jedoch ab, dass der gewählte Ansatz zur Modellierung eine adäquatere und einfachere Abbildung der Realität durch in der Modellierung ungeübte Individuen ermöglicht als andere gängige Modellierungs-Methoden.

Bewertung

Die Fokussierung auf Modellbildung als Aktivität im Rahmen von „Articulation Work“ ist ein Alleinstellungsmerkmal dieser Arbeit. Jørgensen berücksichtigt die für „Articulation Work“ notwendige Flexibilität in der technischen und methodischen Unterstützung derselben und stellt Anforderungen auf, die eine Modellierungssprache erfüllen muss, um „Articulation Work“ adäquat zu unterstützen. Die Unterstützung der Ausführung der Modelle in interaktiver Form ohne die im WfMS üblichen strikten Abläufe kommt einer flexiblen Koordination des eigentlichen Arbeitsablaufs ebenfalls entgegen.

2. Articulation Work

Kontext	Prozessmodellierung, Workflow-Management
Art von AW	explizites „making arrangements“, „coordinating predefined work“
Unterstützung	„making arrangements“: durch flexible, anwenderzentrierte Modellbildung, „coordinating“: durch interaktive Ausführung der erstellten Modelle
Auswirkungen	adäquate und einfache Abbildung sowie intuitive Interpretierbarkeit der Modelle und damit Unterstützung von expliziter „Articulation Work“ durch Modelle von Arbeit

2.4.14. Torres, a Conceptual Framework for Articulation Work across Boundaries

Cabitza et al. (2006) stellen mit „Torres“ ein Framework vor, das sich speziell zur Unterstützung von „globaler Articulation Work“ eignet.

Kontext

Cabitza et al. (2006) entwickeln den weiter oben beschriebenen Ansatz von Sarini und Simone (2002a) weiter und wenden ihn unter Bezugnahme auf Færgemann et al. (2005) auf „globale Articulation Work“ an. Die Autoren stehen damit in der Tradition der technischen Unterstützung von „Articulation Work“ durch CSCW-Werkzeuge und wenden diese im speziellen auf Situationen an, in denen Individuen oder Organisationseinheiten kooperieren müssen, die außerhalb des betreffenden Arbeitsablaufs keine kooperativen Tätigkeiten durchführen.

Unterstützung

Die Autoren bleiben grundsätzlich bei dem Ansatz von Sarini und Simone (2002a), die Koordination von Arbeit mittels Artefakten zu unterstützen, die auf den jeweiligen Anwendungsfall angepasst sind. „Torres“ ist dabei ein Framework zur Erstellung derartiger Artefakte. Dazu wird ein zweistufiger Prozess vorgeschlagen, der die domänenübergreifende Verständlichkeit der artikulierten Informationen sicherstellen soll.

In der ersten Phase werden von jedem beteiligten Individuum bzw. von jeder beteiligten Instanz „Local Formal Representations“ der jeweiligen Domäne erstellt werden. Diese enthalten nicht Modelle der Arbeitsabläufe selbst sondern lediglich die wesentlichen Konzepte der Domäne, die potentiell missverständlich sein können.

In der zweiten Phase werden die „Local Formal Representations“ zu einem einheitlichen, vorgegebenen Modell von „Articulation Work“ mittels ebenfalls vorgegebener Relationen in Beziehung gesetzt. Dadurch ist es möglich, etwaig auftretende Missverständnisse automationsgestützt aufzulösen bzw. zu vermeiden.

Auswirkungen

Die Autoren leiten die Eigenschaften ihres Systems aus den Beobachtungen in einer Fallstudie aus dem medizinischen Bereich ab. Die Implementierung war jedoch zum Zeitpunkt der Publikation nicht abgeschlossen, auch in Folgepublikationen (z.B. (Cabitza et al., 2009)) ist keine explizite Beschreibung von Auswirkungen des Systems im praktischen Einsatz vorhanden.

Bewertung

Die Autoren entwickeln den in (Sarini und Simone, 2002a) vorgeschlagenen „Reconciler“-Ansatz weiter und detaillieren vor allem jene Phase, in der „alignment of meanings“ durchgeführt wird. Methodisch schlagen die Autoren einen individuell durchzuführende zweistufigen Prozess vor, dessen Ergebnisse automatisiert mit den Ergebnissen der anderen Individuen zusammengeführt werden. Methodisch ist dies der detaillierteste Ansatz der Arbeiten der Gruppe rund um Simone und wird deswegen an dieser Stelle betrachtet.

Kontext	CSCW
Art von AW	„making arrangements“
Unterstützung	Abstimmung unterschiedlicher Domänenmodelle und Erstellung von Informations-Artefakten, die domänenübergreifend verständlich sind
Auswirkungen	—

2.4.15. Gegenüberstellung und Zusammenfassung

Die Unterstützung von „Articulation Work“ zeigt ob der großen Spannweite möglicher Ausprägungen eine hohe Vielfalt an möglichen Varianten. Im Allgemeinen zeigt sich die Tendenz, dass die Unterstützung bei einfacheren Formen (wie „coordination of predefined work“) eher mittels rein organisationalen Mitteln erfolgen kann während bei komplexeren Formen von „Articulation Work“ zusätzlich auch technische Mittel eingesetzt werden. Außerdem soll durch technische Unterstützung der Phase, die dem eigentlichen Arbeitsablauf voraus geht (Planung, „making original

2. Articulation Work

arrangements“) das Auftreten von problematischen Situationen (und damit der Bedarf nach komplexen Formen von „Articulation Work“) während der Durchführung des Arbeitsablaufs vermieden werden.

Für einfach Formen von „Articulation Work“, in denen etablierte Arbeitsabläufe koordiniert bzw. kleine Unklarheiten oder Hindernisse beseitigt werden müssen, werden in der Literatur immer wieder soziale Mechanismen als ausreichendes Regulativ beschrieben (z.B. in (Raposo et al., 2001) oder (Schmidt, 1994)). Davenport (2002) nennt „Communities of Practice“ (Wenger, 1999) als mögliche methodische Herangehensweise zur Institutionalisierung dieser sozialen Mechanismen. Ein Spezialfall stellen hier jene Arbeitsabläufe dar, die verteilt abgewickelt werden und in denen deshalb die sozialen Mechanismen durch technische Mittel ermöglicht werden müssen (Færgemann et al., 2005). In diesen Fällen werden auch einfache Formen von „Articulation Work“ technisch unterstützt (z.B. bei (Divitini und Simone, 2000) oder (Fuchs et al., 2001)).

Je komplexer die Arbeitssituation wahrgenommen wird, desto notwendiger wird eine explizite Beschäftigung mit den abzustimmenden Arbeitsaspekten (auch bei Arbeitsabläufen, die nicht oder nur in Teilespekten kooperativ bearbeitet werden (Fjuk et al., 1997)). Letztendliches Ziel ist es immer, einen Status (wieder-)herzustellen, in dem soziale, implizite Mechanismen zur Koordination ausreichen. Organisationale (z.B. bei (Grinter, 1996)) oder auch technische Unterstützung (z.B. bei (Schmidt und Simone, 2000) und allen auf dieser Publikation aufbauenden Arbeiten) kann hierbei hilfreich sein.

Bei der technischen Unterstützung von „Articulation Work“ können zwei Ansatzpunkte unterschieden werden. Einerseits kann die Lösung von aufgetretenen Problemen unterstützt werden (z.B. bei (Grinter, 1996)) oder deren Auftreten von vorne herein verhindert werden (z.B. bei (Raposo et al., 2004)). Andererseits kann „Articulation Work“ bei der Planung von Arbeitsprozessen unterstützt werden, wobei unter anderem die Koordinationsmechanismen während der Durchführung des Arbeitsablaufs vereinbart werden (z.B. in (Sarini und Simone, 2002b) und den darauf aufbauenden Arbeiten). Ziel ist es hier, schon im Vorfeld den Arbeitsablauf bzw. die unterschiedlichen Sichten darauf soweit abzustimmen, dass die Koordination möglichst unproblematisch und implizit abgehandelt werden kann.

Diese Form von „Articulation Work“ wird vom Großteil der in diesem Bereich tätigen Autoren durch Modelle der zu koordinierenden Arbeit als explizite Artefakte im Artikulationsprozess unterstützt (z.B. bei (Divitini und Simone, 2000), (Sarini und Simone, 2002a), (Raposo et al., 2004) oder (Jørgensen, 2004)). Diese Modelle werden von den beteiligten Personen erstellt und müssen syntaktisch einfach bzw. intuitiv zu handhaben und semantisch flexibel sein (Herrmann et al., 2002) (Jørgensen, 2004). Der Inhalt der Modelle kann sowohl der eigentliche Arbeitsab-

lauf sein (Divitini und Simone, 2000) als auch die allgemeinen Struktur der Arbeitsdomäne konzeptionell abbilden (Sarini und Simone, 2002a).

Die existierenden Arbeiten in diesem Bereich setzten allerdings die Existenz dieser Modelle voraus bzw. schaffen die konzeptionellen Rahmenbedingungen, um die Erstellung derartiger Modelle zu ermöglichen. Dabei sind sie immer durch den Anspruch eingeschränkt, die Modelle automationsgestützt zur Unterstützung der Koordination im Arbeitsablauf selbst weiterzuverarbeiten. Die Anforderung an die Modellierungssprache sind also nur zum Teil aus den Bedürfnissen der Benutzer abgeleitet, sondern sind vielmehr ein Kompromiss zwischen den Anforderungen des technischen Systems und der Bedürfnisse der Benutzer. Mit der konkreten Erstellung der Modelle selbst beschäftigen sich lediglich Herrmann et al. (2002) sowie Jørgensen (2004), auch sie gehen aber nicht auf die konkrete Unterstützung der Individuen im Modellierungsprozess ein.

Die Unterstützung der an einem Arbeitsablauf beteiligten Individuen bei Modellbildung und -abstimmung als „Articulation Work“ weist an dieser Stelle also Lücken auf, die bislang nicht behandelt wurden. Dies ist insofern problematisch, als dass vor allem komplexe Formen von „Articulation Work“ auf externe Repräsentationen zurückgreifen, um die Abstimmung zu erleichtern. Im weiteren Verlauf dieser Arbeit wird deswegen versucht, methodische und technische Hilfsmittel zu entwickeln, die auf Modellen basierende „Articulation Work“ unterstützen kann. Einen möglichen Ansatzpunkt dazu liefert bereits Strauss (1993).

2.5. Thought processes und Articulation Work

Zur Zielsetzung von „Articulation Work“ und deren Unterstützung treffen die betrachteten Arbeiten klare Aussagen. Offen bleiben jedoch vor allem bei der Unterstützung komplexerer Formen von „Articulation Work“ explizite Aussagen zu den notwendigen Leistungen der Individuen im Prozess der Artikulation, deren konkrete Ausgestaltung und der möglichen Unterstützung.

Bereits Strauss ist sich dieser (konzeptionellen) Auslassung bewusst³⁹, und beschäftigt sich in späteren Arbeiten (Strauss, 1993) auch mit jenen kognitiven Vorgängen, die von ihm als „thought processes“ oder „mental activities“ bezeichnet werden und die untrennbar mit jeder Art von Tätigkeit und Interaktion verbunden sind⁴⁰ und diese beeinflussen⁴¹.

³⁹ „[...] many social scientist pay almost no attention to interior activity: ignoring it, taking it for granted, but leaving it unexamined, or giving it the kind of abstract but not very detailed analysis [...]“ (Strauss, 1993, S. 131)

⁴⁰ „These [thought processes] accompany visible action, as well as precede and follow in conditional and consequential modes“ (Strauss, 1993, S. 146)

2. Articulation Work

Im Kontext der Abstimmung von Arbeitsabläufen kommt den „thought processes“ der Individuen große Bedeutung zu, da sie den sichtbaren individuellen Handlungen zugrunde liegen bzw. diese beeinflussen. „Articulation Work“ wirkt sich also auf die „thought processes“ der beteiligten Individuen aus (wie auch von (Davenport, 2002) im Kontext des „situated learning“ erwähnt). „Thought processes“ umfassen „*images, imaginations, projections of scenes, [...] flashes of insight, rehearsals of action, construction and reconstruction of scenarios, the spouting up of metaphors or comparisons, the reworking and reevaluating of past scenes and one's actions within them, and so on and on*“ (Strauss, 1993, S. 130) - also im Wesentlichen alle kognitiven Vorgänge, die unmittelbar oder mittelbar im Zusammenhang mit den sichtbaren Arbeitsaspekten, insbesondere den Tätigkeiten zur Zielerreichung und der wahrgenommenen Arbeitsumgebung, stehen. Strauss interessiert sich allerdings ausschließlich für die dynamischen Aspekte der Interaktion zwischen Individuen, nicht aber für die Ausgangspunkte und Ergebnisse der zugrunde liegenden „thought processes“.⁴²

Die Repräsentationen, auf denen „thought processes“ beruhen und operieren, sind jedoch für die Unterstützung von „Articulation Work“ von Interesse. Vorhandene Arbeiten beschäftigen sich lediglich mit bereits externalisierten Repräsentationen, gehen jedoch ebenfalls nicht auf deren Erstellung oder Ursprung ein. Die kognitionswissenschaftlichen Ansätze zu Schemata ((Rumelhart und Norman, 1978) (vgl. nach Hanke, 2006)) und mentalen Modellen ((vgl. Seel, 1991)) sind ein Erklärungsansatz für diese Lücke (Herrmann et al., 2002).

⁴¹ „*Even well-grooved, routine action and interaction may be accompanied by thought [...] directly relevant to the work at hand. As I vacuum the house, barely noticing my movements, still I give myself commands [...]*“ (Strauss, 1993, S. 132)

⁴² „*I use the gerund 'ing' after 'symbol' [bei der Beschreibung von 'symbolizing', Anm.] to signify that my principal interest is, again, in interaction rather than its products, for symbols are precipitates of interaction*“ (Strauss, 1993, S. 149)

3. Mentale Modelle

In diesem Kapitel wird das Konzept der mentalen Modelle eingeführt, das in dieser Arbeit als Erklärungsansatz für jene Aspekte von "Articulation Work" verwendet wird, die die nicht sichtbaren, kognitiven Beiträge eines beteiligten Individuums betreffen. Nach einer Einführung in die Begriffswelt der mentalen Modelle wird die Argumentation aus dem letzten Kapitel nochmals aufgegriffen und die mögliche Rolle mentaler Modelle für "Articulation Work" erörtert. In der Folge werden Methoden eingeführt mit denen mentale Modelle externalisiert und kommuniziert werden können. Basierend auf diesen Beschreibungen wird im letzten Teil des Kapitels untersucht, welche Herausforderungen sich bei der Anwendung dieser Methoden im Kontext von "Articulation Work" ergeben können.

3.1. Articulation Work und mentale Modelle

Wie bereits im vorgehenden Kapitel beschrieben, wird in vorhandenen Arbeiten zu Articulation Work deren Auftreten, Kontext und Wirkung beschrieben, nicht aber die individuellen Aspekte ihrer Durchführung. Der eigentliche Gegenstand der Abstimmung, die im Rahmen der Articulation Work erfolgen soll, wird ebenfalls nicht konkret festgelegt. Strauss spricht von „*putting together tasks, task sequences, task clusters - even aligning larger units such as lines of work and subprojects - in the service of work flow*“ (Strauss, 1988, S. 2), und konkretisiert „*the specific questions about tasks of course include: what, where, when, how, for how long, how complex, how well defined are their boundaries, how attainable are they under current working conditions, how precisely are they defined in their operational details, and what is the expected level of performance. (Which of those are the most salient dimensions depends on the organizational work context under study, and we cannot emphasize too much that it is the researcher who must discover these saliences.)*“ (Strauss, 1985, S. 6). Strauss lässt also offen, was es exakt ist, dass abgestimmt werden muss bzw. verlagert diese Frage in den konkreten Einzelfall.

Strauss spricht diese Auslassung in einer späteren Arbeit explizit an (Strauss, 1993, S. 131) und beschäftigt sich in dieser auch mit jenen kognitiven Vorgängen, die von ihm als „thought processes“ oder „mental activities“ bezeichnet werden und die un-

trennbar mit jeder Art von Tätigkeit und Interaktion verbunden sind (Strauss, 1993, S. 146) und diese beeinflussen (Strauss, 1993, S. 132).

Im Kontext der Abstimmung von Tätigkeiten kommt den „thought processes“ der Individuen große Bedeutung zu, da sie den sichtbaren individuellen Handlungen zugrunde liegen bzw. diese beeinflussen. „Articulation Work“ wirkt sich also auf die „thought processes“ der beteiligten Individuen aus. „Thought processes“ umfassen „*images, imaginations, projections of scenes, [...] flashes of insight, rehearsals of action, construction and reconstruction of scenarios, the spouting up of metaphors or comparisons, the reworking and reevaluating of past scenes and one's actions within them, and so on... and on*“ (Strauss, 1993, S. 130) - also im Wesentlichen alle kognitiven Vorgänge, die unmittelbar oder mittelbar im Zusammenhang mit den sichtbaren Arbeitsaspekten, insbesondere den Tätigkeiten zur Zielerreichung und der wahrgenommenen Arbeitsumgebung, stehen. Strauss interessiert sich allerdings ausschließlich für die dynamischen Aspekte der Interaktion zwischen Individuen, nicht aber für die Ausgangspunkte und Ergebnisse der zugrunde liegenden „thought processes“ (Strauss, 1993, S. 149)

3.2. Begriffsbestimmung

Das Konzept der „mentalen Modelle“ wird grundsätzlich verwendet, um zu erklären „*wie Menschen die Welt verstehen – genauer: wie sie ihr Wissen benutzen, um sich bestimmte Phänomene der Welt subjektiv plausibel zu machen*“ (Seel, 1991, S. VII). Mentale Modelle sind dabei Erklärungsmodelle der Welt, die Menschen auf Basis von Alltagserfahrungen, bisherigem Wissen und darauf basierenden Schlussfolgerungen bilden. Ein gebildetes mentales Modell wird dann als Basis verwendet, um die Welt zu verstehen und ggf. Vorhersagen über deren Verhalten zu bilden. (Seel, 1991, S. VII)

Im Wesentlichen wurde das Forschungsfeld der mentalen Modelle durch zwei Arbeiten maßgeblich beeinflusst. Johnson-Laird (1981) und de Kleer und Brown (1981) führen den Begriff als eigenständigen Forschungsgegenstand ein und legen damit die Grundlage für einen Großteil der nachfolgenden Arbeiten in dem Gebiet. Im Kontext dieser Arbeit werden dabei zwei dieser nachfolgenden Arbeiten näher betrachtet. Zum einen stellt Norman (1983b) den Begriff erstmals im den Kontext der Mensch-Maschine-Interaktion dar. Zum anderen versucht Seel (1991) die unterschiedlichen Richtungen der Forschung im Bereich der mentalen Modelle zusammenzuführen und daraus die Bedeutung von Mentalen Modellen für Lernvorgänge (unter die – im breiten Verständnis von Seel – auch die hier relevanten Abstimmungsvorgänge fallen) und Möglichkeiten zu deren Unterstützung abzuleiten.

Die folgenden Ausführungen basieren deshalb auf den Ausführungen von Seel und seiner Mitarbeiter (Ifenthaler (2006), Pirnay-Dummer (2006) und Hanke (2006)).

Mentale Modelle sind nach Ifenthaler (2006, S. 7) „*kognitive Konstruktionen, die abhängig von der jeweiligen Situation und vom semantischen Wissen einer Person ad hoc konstruiert werden*“. Ein mentales Modell ist also kein permanentes kognitives Konstrukt, sondern wird auf Basis vorhandenen Wissens in bestimmten Situationen ad-hoc gebildet (siehe dazu auch Abschnitt 3.3). In engem Zusammenhang mit dem Begriff der mentalen Modelle ist jener der „Schemata“ bei zu nennen. „Schemata“ unterscheiden sich ihrer Definition nach nur in Detail von „mentalnen Modellen“¹. Ein „Schema“ repräsentiert nach Seel (2003, S. 57) „*das aufgrund vielfältiger Einzelerfahrungen mit Objekten, Personen, Situationen und Handlungen erworbene verallgemeinerbare und abstrakte Wissen einer Person.*“ Schemata werden benutzt um „*Wissensstrukturen zu beschreiben, welche typische Zusammenhänge eines Realitätsbereiches repräsentieren.*“ (Ifenthaler, 2006, S. 8). Auf Basis dieser „Schemata“ treffen Individuen treffen Handlungentscheidungen in bestimmten Situationen. „Schemata“ sind dabei als „Vorlagen“ zu sehen, die adäquate Handlungen für einen bestimmten Situationstypus vorgeben (im Sinne der erwähnten „Verallgemeinerbarkeit“) und Individuen damit zur raschen, unmittelbaren Handlung befähigt, ohne ausführliche Planungstätigkeiten durchführen zu müssen. In Abgrenzung dazu werden „mentale Modelle“ ad-hoc in Situationen gebildet, wo keine Schemata vorhanden sind oder vorhandene nicht angewandt werden können.

Ifenthaler (2006) beschreibt den Zusammenhang zwischen Schemata und mentalen Modellen wie in Abbildung 3.1 dargestellt. Er bezieht sich dabei auf das „Äquibrations“-Prinzip nach Piaget (1976). Demnach entwickelt sich das Wissen eines Individuum durch die komplementären Prozesse „Assimilation“ und „Akkommodation.“

Solange eine wahrgenommene Situation auf existierende Schemata abgebildet werden kann und daraus unmittelbar Handlungen abgeleitet werden können, spricht man von „Assimilation“ der wahrgenommene Information. „Assimilation“ festigt bestehende Schemata, gestaltet diese ggf. in Details exakter aus oder um, stellt die grundlegenden Annahmen, die dem Schema zugrunde liegen, aber nicht in Frage. Kann die wahrgenommene Information nicht auf existierende Schemata abgebildet werden, kommt es zur „Akkommodation“, also der (ad-hoc) Bildung eines mentalen Modells und darauf aufbauend zur „*Restrukturierung, Veränderung und Neuorganisation*“ (Ifenthaler, 2006) der betreffenden Schemata. Schemata und mentale Modelle können damit auch als jene Strukturen interpretiert werden, die beim „Single“- bzw. „Double-Loop-Learning“ nach Argyris und Schön (1978) zum Einsatz kommen. Im

⇐

¹Tatsächlich wird nach Ifenthaler (2006) der Begriff der „mentalnen Modelle“ von manchen Autoren zugunsten von „Schemata“ als überflüssig bezeichnet, da zweitere die auftretenden kognitiven Phänomene ausreichend beschreiben würden.

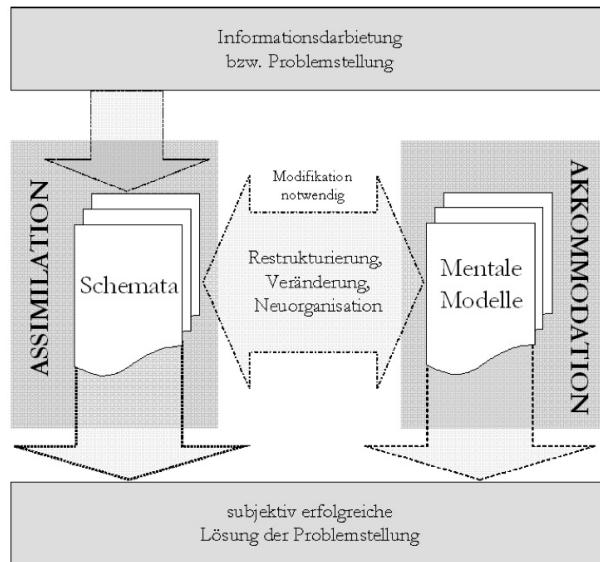


Abbildung 3.1.: Schemata und mentale Modelle (entnommen aus Ifenthaler (2006, S. 10))

Kontext von „Articulation Work“ sind mentale Modelle in jenen Situation von Interesse, die als so „problematisch“ wahrgenommen werden, dass keine Fortführung der operativen Arbeit mehr möglich ist (auf individueller Ebene also evtl. existierende „Schemata“ nicht mehr zum Einsatz gebracht werden können). In diesen Situationen muss „explizite Articulation Work“ durchgeführt werden, um auf Basis eines mentalen Modells dieses selbst zu verändern, auszugestalten und soweit mit der Umwelt abzustimmen, das eine Wiederaufnahme der operativen Arbeit (bzw. die Bildung von adäquaten Schemata) möglich wird. Um auf die Durchführung von „expliziter Articulation Work“ unter Bezugnahme auf die mentalen Modelle der Individuen näher eingehen zu können, werden im nächsten Abschnitt die Bildung und Veränderung mentaler Modelle näher betrachtet.

3.3. Bildung und Veränderung mentaler Modelle

Nach (Seel, 1991) umfasst die Bildung mentaler Modelle zwei Komponenten: Eine *deklarative Komponente*, in der bereichs- bzw. domänen-spezifisches Wissen in der Form von hier nicht näher spezifizierten, strukturierten Wissensbasen abgelegt wird und eine *operative Komponente*, in der auf Grundlage dieser Wissensbasen

3.3. Bildung und Veränderung mentaler Modelle

Schlüsse gezogen und neues Wissen abgeleitet wird, die über das ursprüngliche domänen spezifische Wissen hinausgeht.

Das in den Wissensbasen repräsentierte Wissen kann auf Alltagserfahrung begründet sein oder durch Vermittlung oder Instruktion begründet werden. Im ersten Fall ist das Wissen dann als konkret und handlungsbezogen angesehen werden, im zweiten Fall ist das Wissen eher auf abstrakter, formaler Ebene anzusiedeln. Analog dazu kann auch in der operativen Komponente die Schlussfolgerung induktiv auf Basis eines „intuitionsbegründeten“ Regelsystems gezogen werden oder durch Deduktion mittels einem formal begründbaren Regelsystem gebildet werden.

Die Modifikation und Erweiterung der eigenen Wissensbasen und die (Weiter-)Entwicklung der kognitiven Fähigkeiten, die für die Ableitung von Schlussfolgerungen notwendig sind, bezeichnet Seel (1991) als „Lernen“. Lernen ist *„mit der Verarbeitung individueller Erfahrungen mit sowie vermittelter Information über die Welt, ihre Struktur und Evidenz verbunden und kann als ein Prozess permanenter konzeptueller Veränderungen verstanden werden.“* (Seel, 1991, S. 23). Lernen setzt damit die Fähigkeit und Bereitschaft voraus, *„vermittelte Weltauffassungen zu verstehen, zu akzeptieren und sodann den eigenen gedanklichen Konstruktionen zugrunde zu legen“* (Seel, 1991, S. 23). Im Wesentlichen entspricht dies einer Verallgemeinerung jener Vorgänge die im Rahmen von nicht rein koordinierender sondern abstim mender und vor allem planender „Articulation Work“ durchgeführt werden.

In diesem Zusammenhang sind verschiedene Arten von mentalen Modellen zu unterscheiden. Seel (1991) differenziert zwischen „Novizenmodellen“ und „Expertenmodellen“. Ein „Novizenmodell“ ist ein Alltagsmodell, dass ad-hoc in einer Problemsituation gebildet wird und ist im dem Individuum, das es gebildet hat, in der aktuellen Situation plausibel (auch wenn es objektiv falsch ist). Es ist ausreichend, um adäquate Reaktionen auf die gegebene Situation abzuleiten, ohne das notwendigerweise eine Begründung der Handlungen möglich ist oder diese nicht mit dem tatsächlichen Grund der Problembewältigung übereinstimmen². Je öfter die Anwendung eines „Novizenmodells“ zum Erfolg führt, umso stabiler wird es zur Grundlage des Handelns des Individuums in der jeweiligen Situation. „Expertenmodelle“ (oder „wissenschaftliche Modelle“) sind hingegen inhaltlich vollständiger und bilden die Ursache-Wirkungs-Zusammenhänge der beobachtbaren Realität ab (sind also „objektiv korrekt“). Sie sind im allgemeinen differenzierter und bedienen sich einer adäquateren mentalen Codierung als „Novizensysteme“ (die sich i.A. existierender mentale Codierungen bedienen). Auch die Kompetenz des Individuums im Umgang mit dem mentalen Modell ist in diesem Fall höher. Der Übergang von ei-

² „[...] most people's understanding of the devices they interact with is surprisingly meager, imprecisely specified, and full of inconsistencies, gaps and idiosyncratic quirks.“ (Norman, 1983a, S. 8)

3. Mentale Modelle

nem „Novizenmodell“ zu einem „Expertenmodell“ erfolgt dabei durch „Lernen“ im oben genannten Sinn. (Ifenthaler, 2006)

„Expertenmodelle“ müssen aber nicht immer den erwünschten Endzustand eines Lernprozesses darstellen. Durch die gesteigerte Komplexität des Modells wird dessen ad-hoc-Anwendung schwieriger, die Nützlichkeit des Modells ist deshalb eingeschränkt (vgl. Ifenthaler, 2006, S. 20). Hier zeigt sich wiederum eine Analogie mit dem Bereich „Articulation Work“, wo es – wie im letzten Kapitel beschrieben – ebenfalls nicht als erforderlich angesehen wird, dass jedes beteiligte Individuum eine detaillierte Gesamtsicht auf den Arbeitsablauf hat sondern es ausreichend ist, die jeweils relevanten Schnittstellen zu abzustimmen und einen groben Überblick über den Gesamtzusammenhang zu haben.

Ifenthaler (2006) erweitert deshalb die binäre Klassifikation durch „Erklärungsmodelle“, die er konzeptionell zwischen „Novizen-“ und „Expertenmodellen“ ansiedelt. Ein „Erklärungsmodell“ „*beinhaltet alle notwendigen Informationen, um ein Problem bezüglich des Sachverhaltes und der Anforderungssituation richtig zu lösen. Einem Erklärungsmodell wird dabei ein hoher Grad an Nützlichkeit beigemessen, was in Bezug auf die kognitive Leistung zu einer ergonomischen Problemlösung führt. Je nach Komplexität des Sachverhaltes und der damit verbundenen Anforderungssituation kann ein Erklärungsmodell einem Novizenmodell oder einem Expertenmodell sehr ähnlich sein*“ Ifenthaler (2006, S. 21). „Erklärungsmodelle“ sind also je nach Art der zugrunde liegenden Problemstellung unterschiedlich aufgebaut. Ziel eines „Erklärungsmodells“ ist es immer, bestmöglich zur Problemlösung beizutragen, diese also für das Individuum im Kontext der jeweiligen Problemstellung möglichst einfach zu gestalten. Ein „Erklärungsmodell“ gewinnt dabei durch „Lernen“ an Reifegrad, es nähert sich einem „Expertenmodell“ immer weiter an.

- ⇒ Im Kontext von „Articulation Work“ ist der Begriff des „Erklärungsmodells“ ein hilfreiches Konstrukt. Je nach Reifegrad des betreffenden mentalen Modells wird eine bestimmte Arbeitssituation als mehr oder weniger komplex wahrgenommen. Je komplexer eine Arbeitssituation wahrgenommen wird, desto größer ist der Bedarf nach expliziter „Articulation Work“, also der expliziten Beschäftigung mit dem Arbeitsprozess, dessen Reflexion und der Abstimmung der eigenen Wahrnehmung mit anderen beteiligten Individuen. Diese Beschäftigung mit dem Arbeitsprozess, also jene Tätigkeiten, die im Rahmen der „Articulation Work“ durchgeführt werden, entsprechen – wie oben bereits beschrieben – dem hier beschriebenen „Lernen“, wobei in kooperativen Arbeitssituation die Quellen „vermittelter Information“ vorrangig die anderen beteiligten Individuen oder organisationale Artefakte sind, die den Arbeitsablauf beschreiben.

Die Veränderung mentaler Modelle weist zwei grundlegende Schwierigkeiten auf. Bei bereits als nicht adäquat erkannten mentalen Modellen (wie sie bei „Articula-

tion Work“, die in „non-routine work“ bzw. „problematic work“ ausgeführt wird, auftreten), besteht grundsätzlich die Bereitschaft zur Veränderung (im Sinne einer „Akkommodation“ des mentalen Modells an die als verändert wahrgenommene Umweltbedingungen), die Herausforderung besteht aber darin, die notwendigen Informationen vermittelt zu bekommen also an diese zu gelangen und diese adäquat dargeboten zu bekommen. Eine weitere Schwierigkeit ergibt sich in Situationen, in denen nicht alle involvierten Individuen die Situation als „problematisch“ wahrnehmen und deshalb keine grundlegende Bereitschaft zeigen, ihre der Arbeit zugrunde liegenden Annahmen (also ihre mentalen Modelle) zu verändern ((Ifenthaler, 2006) spricht von „hoher Veränderungsresistenz“). Dies tritt vor allem im Situationen auf, in denen „Articulation Work“ nicht aus einer allgemein wahrgenommenen Problemsituation heraus durchgeführt wird, sondern entweder mit rein planendem Charakter angestoßen wird oder nur für einzelnen beteiligte Individuen so stark „problematisch“ ist, dass eine explizite Beschäftigung mehrerer oder aller am Arbeitsablauf beteiligten notwendig ist.

Grundsätzlich müssen also aus der Theorie der mentalen Modelle heraus begründet zu erfolgreicher „expliziter Articulation Work“ drei Rahmenbedingungen gegeben sein:

1. Die Beteiligten müssen bereit sein, ihre mentalen Modelle abzustimmen und das individuelle Verständnis der Schnittstellen abzugleichen.
2. Die von den beteiligten Individuen benötigte Information über den Arbeitsablauf muss von den anderen Beteiligten zur Verfügung gestellt werden können oder in der Form organisationaler Artefakte vorliegen.
3. Die benötigte Information muss in adäquater Form dargeboten werden, um die individuellen mentalen Modellen mit diesen in Einklang zu bringen.

Anforderung 1 ist eine Frage des sozialen Verhaltens der beteiligten Personen bzw. der Organisationskultur und kann ggf. durch organisationale Maßnahmen (z.B. durch die Förderung von „Communities of Practice“ (Wenger, 1999)) unterstützt werden. Anforderung 2 kann trivial zu erfüllen sein, wenn der Kontext des Arbeitsablaufs (d.h. das Umfeld, in dem die Arbeit durchgeführt wird, u.a. inkl. aller beteiligten Individuen) bekannt ist. In komplexen, neuartigen oder unbekannten Arbeitssituationen muss auch die Erfüllung dieser Anforderung unterstützt werden. Ansatzpunkte dafür liefert im Bereich von „Articulation Work“ etwa (Grinter, 1996) oder (Fuchs et al., 2001), umfassend mit dieser Thematik beschäftigt sich der Forschungsbereich der „Organisational Memories“ (zur technischen Unterstützung siehe etwa (Abecker et al., 1998) oder (Diefenbruch et al., 2002)³).

³eine detaillierte Darstellung des Forschungsgebiets ist in (Maier, 2008) zu finden

3. Mentale Modelle

Anforderung 3 wird in der Forschung zum Thema „Articulation Work“ von Sariani und Simone (2002a) im Kontext des „alignment of meanings“ angesprochen, bei dem eine automationsgestützte Abbildung unterschiedlicher Domänenvokabulare die grundsätzliche Verständigung bzw. die Vermeidung von Missverständnissen vermeiden soll. Diese Maßnahme ermöglicht allerdings erst die Abstimmung mentaler Modelle, unterstützt diese aber noch nicht unmittelbar. Im Sinne der adäquaten Form der Darbietung argumentieren auch (Divitini und Simone, 2000), (Jørgensen, 2004) und vor allem (Herrmann et al., 2002) mit der Forderung von flexiblen bzw. an die Bedürfnisse der Benutzer anpassbaren Modellierungsprachen bei der Verwendung von externalisierten Modellen als Unterstützung der Durchführung von „Articulation Work“.

- ⇒ Hinsichtlich Anforderung 3 argumentiert Seel (1991) im Bereich der mentalen Modelle ebenfalls für die Nützlichkeit externalisierter Repräsentationen zur Verbesserung von mentalen Modellen. Er vertritt die Auffassung, dass „die Externalisierung die Konstruktion eines mentalen Modells 'vervollkommnet'“, was er darauf zurückführt, dass „erst aus der Zielsetzung heraus, sich einem anderen mitzuteilen, die Präzision einer gedanklichen Konstruktion resultiert, die für die Erklärung einer Weltergebnisheit erforderlich ist“ (Seel, 1991, S. 155). Die Bildung von Externalisierungen mentaler Modelle verbessert also einerseits das individuelle mentale Modell, indem sie Lücken und Inkonsistenzen bewusst macht, und ermöglicht andererseits die Kommunikation mit anderen Individuen und ist damit die Grundlage der Vermittlung von Information und damit des „Lernens“ und „Articulation Work“ im hier beschriebenen Sinn. Seel (1991) lässt jedoch offen, in welcher Form die Repräsentation erfolgt, um die Vermittelbarkeit bestmöglich sicherzustellen⁴. Ifenthaler (2006), Hanke (2006) und (Pirnay-Dummer, 2006) weisen an dieser Stelle jedoch auf qualitative Unterschiede der Eignung unterschiedlicher externer Repräsentationsformen mentaler Modelle zur Externalisierung und Vermittlung derselben hin. Auf diese unterschiedlichen Formen und deren Eignung wird im nächsten Abschnitt eingegangen.

Grundsätzlich konnte jedoch hier gezeigt werden, dass „Articulation Work“ in ihren komplexen Ausprägungen die mentalen Modelle der beteiligten Individuen verändert bzw. auf diesen aufbaut. In weiterer Folge wurde deutlich, dass zur expliziten Durchführung von „Articulation Work“ auch aus der Theorie der mentalen Modelle heraus die Verwendung von externalisierten Modellen (wie bereits von (Divitini und Simone, 2000), (Herrmann et al., 2002) und (Jørgensen, 2004) ohne den Hintergrund der mentalen Modelle vorgeschlagen) sinnvoll ist.

⁴„Internalisierung von Erkenntnismitteln setzen Zeichensysteme (auditiver, visueller oder anderer Natur) für die Verschlüsselung der semantischen Gebilde voraus“ (Seel, 1991, S. 155)

3.4. Externalisierung mentaler Modelle

Diese Forderung nach einer Externalisierung mentaler Modelle, um deren Abstimmung zu unterstützen wird auch durch die Ausführungen von Senge (1990) und (Kim, 1993) gestützt, die mentale Modelle⁵ als ein wichtiges Konzept im Kontext des organisationalen Lernens identifizieren. Mentale Modelle bilden dort die Grundlage für Handlungen von Individuen in Organisationen und müssen geteilt werden, um der Organisation selbst eine Weiterentwicklung (einen „organisationalen Lernschritt“) zu ermöglichen.

3.4. Externalisierung mentaler Modelle

Die Externalisierung von „mentalen Modellen“ ist immer ein zweistufiger Prozess (siehe Abbildung 3.2), in dem jeweils eine Transformation des Kodierungssystems stattfindet. Die wahrgenommene Realität (das „Weltwissen“) wird (ad-hoc) in ein mentales Modell abgebildet werden. Soll dieses externalisiert werden, ist dazu ein weiterer Übersetzungs- bzw. Abbildungsschritt notwendig. Gleichzeitig führt jede Modellbildung nach Stachowiak (1973) neben der „Abbildung“ auch zur „Verkürzung“, d.h. dass das Modell nicht die gesamte Information des Originals enthält, sondern nur jene Aspekte, die dem Ersteller relevant erscheinen. In diesem Sinne können sie nur in einem bestimmten (zeitlichen, personellen und operationalen) Kontext für das Original stehen („pragmatisches Merkmal“ – jedes Modells ist für einen bestimmten Zweck konstruiert).

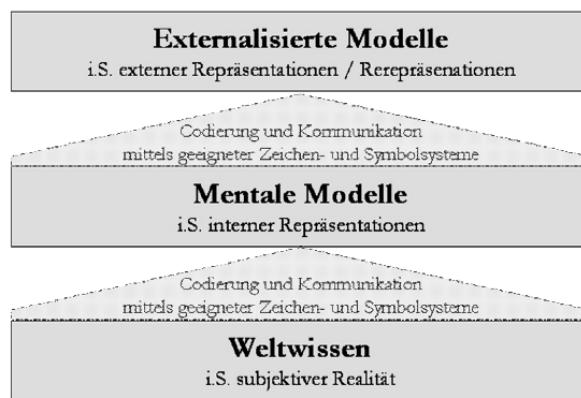


Abbildung 3.2.: Externalisierung mentaler Modelle (entnommen aus (Ifenthaler, 2006))

⁵in einem etwas breiteren Verständnis, welches im Wesentlichen auch Schemata umfasst: „[mental models are] deeply ingrained assumptions, generalizations, or even pictures and images that influence how we understand the world and how we take action.“Senge (1990)

3. Mentale Modelle

Herausfordernd ist im Kontext von „Articulation Work“ das Abbildungsmerkmal, also die notwendige Übersetzungsleistung bei der Externalisierung eines mentalen Modells. Externalisierung umfasst nach Hanke (2006) immer die „Repräsentation“ als auch die „Kommunikation“ eines mentalen Modells. Die Relevanz dieser beiden Aspekte verschiebt sich je nach Kontext bzw. Zielsetzung der Externalisierung. Dient diese eher der individuellen Verständnisbildung, steht die „Repräsentation“ im Vordergrund (diesen Zweck erfüllt nach (Seel, 1991) auch bereits das mentale Modell selbst, die Externalisierung hat schärfenden Charakter). Die externalisierte Repräsentation soll nach Seel (1991, S. 187) in Bezug auf das repräsentierte mentale Modell

- vollständig
- konzise
- kohärent und konkret
- bedeutungshaltig und korrekt

sein. Das Kodierungssystem muss hier also so gewählt werden, dass eine möglichst unmittelbare Abbildung der mentalen Modelle auf die externalisierte Repräsentation (und vice versa) möglich ist.

In Situationen, in denen mentale Modelle zusätzlich auch anderen Individuen vermittelt werden sollen, steht „Kommunikation“ im Vordergrund. Die „Kommunizierbarkeit“ eines mentalen Modells hat Auswirkungen auf die wählbaren Kodierungssysteme zur Externalisierung. Das gewählte Kodierungssystem muss allen beteiligten Individuen verständlich sein, während dieses Kriterium bei der individuellen Verständnisbildung irrelevant ist (Hanke, 2006). Im Rahmen von „Articulation Work“ steht im Allgemeinen die „Kommunikation“ bei der Externalisierung im Vordergrund, wobei diese ohne eine adäquate „Repräsentation“ nicht möglich ist. Ziel muss es also sein, Kodierungssysteme zur Verfügung zu stellen, die

- allen beteiligten Individuen verständlich sind, und
- eine möglichst unmittelbare Abbildbarkeit der mentalen Modelle auf die externalisierte Repräsentation ermöglicht.

Kodierungssysteme können „*auditiver, visueller oder anderer Natur*“ (Seel, 1991, S. 155) sein. Das gebräuchlichste Kodierungssystem ist die natürliche Sprache. Diese ist im Sinne der ersten Anforderung oft eine gute Wahl, bietet aber aufgrund ihrer Generizität nur wenig Möglichkeiten, sowohl den Repräsentations- als auch den Kommunikationsprozess bei der Externalisierung explizit zu unterstützen. Ifenthaler (2006) stellt mehrere Methoden vor, die sich spezifisch zum Zwecke der Externalisierung mentaler Modelle eignen und zu qualitativ höherwertigen Externalisierungsergebnisse führen sollen. Dies sind im Einzelnen:

- Methode des lauten Denkens

- Strukturlegetechniken
- Concept Mapping

Auch (Huss, 2003) erwähnt diese Ansätze im Zusammenhang mit der Externalisierung „mentaler Repräsentationen“. In der Folge werden die genannten Ansätze detaillierter betrachtet. Dabei kommt folgender Raster zum Einsatz:

Konzept beschreibt die grundlegenden Konzepte des Ansatzes und die darauf aufbauende Zielvorstellung

Vorgehen beschreibt, wie die Zielerreichung methodisch sichergestellt werden soll.

Unterstützung beschreibt, welche (technischen) Unterstützungsmaßnahmen vorgeschlagen werden.

Bewertung fasst die Eigenschaften der Methode zusammen und beurteilt sie hinsichtlich ihrer Eignung für „Articulation Work“

3.4.1. Methode des lauten Denkens

Die „Methode des lauten Denkens“ (Van Someren et al., 1994) beschreibt ein Vorgehen, bei dem Individuen während ihrer operativen Tätigkeit ihre Gedanken und die Motive für ihr Handeln verbalisieren.

Konzept

Die Grundidee des „Methode des lauten Denken“ basiert darauf, alle Gedanken, die im eine Tätigkeit begleiten, laut auszusprechen ohne sich auf diese Verbalisierung explizit zu konzentrieren (und etwa über Formulierungen nachzudenken oder Interpretationen durchzuführen). Es werden keine Fragen gestellt, das externalisierende Individuum wird ggf. lediglich daran erinnert, seine Gedanken auszusprechen. Nach Van Someren et al. (1994) hat diese Form der Externalisierung keinen negativen Einfluss auf die Durchführung der eigentlichen Tätigkeit.

Die „Methode des lauten Denkens“ wird immer mit dem Ziel durchgeführt, die kognitiven Prozesse in einer bestimmten Situation offenzulegen. Dazu muss eine Problemstellung ausgewählt werden, die diese Situation auslöst und möglichst keine oder geringe Nebeneffekte aufweist. Dazu gehört etwa die kognitive Überforderung des Individuums, wenn die Aufgabe als zu schwierig wahrgenommen wird. Gleichzeitig führt eine zu einfache Aufgabe zu einer routinisierten Abarbeitung, deren kognitiven Prozesse zumeist implizit und schwer zu externalisieren sind (vgl. „Operations“ in der „Activity Theory“ (Leont'ev, 1978)). Der wahrgenommene Schwierigkeitsgrad der Aufgabe steht in direktem Bezug mit der Expertise des Individuums

(als unabhängige Variable), die bei der Auswahl der Problemstellung berücksichtigt werden muss.

Die „Methode des lauten Denkens“ wird oft mit Retrospektion kombiniert. Retrospektion ist die angeleitete Reflexion über eine Tätigkeit im Nachhinein (also nach Abschluss der Tätigkeit). Im Falle der Kombination mit der „Methode des lauten Denkens“ wird diese Reflexion mit Protokollen der verbalisierten Gedanken durchgeführt, was Unklarheiten in diesen Protokollen und eine tiefergehende Reflexion ermöglichen kann.

Die Ergebnisse der „Methode des lauten Denkens“ werden auf Basis von Audio- oder Video-Aufnahmen möglichst exakt transkribiert. Die Auswertung der Protokolle erfolgt im Normalfall nicht durch das Individuum selbst sondern wird interpretativ durch Dritte durchgeführt. Von Van Someren et al. (1994) wird dazu unter anderem vorgeschlagen eine Aufgaben-Analyse durchzuführen, deren Ergebnis im Allgemeinen ein (diagrammatisches) Modell der Aufgaben und Tätigkeiten des Individuum zur Zielerreichung ist.

Vorgehen

Van Someren et al. (1994) beschreiben einen prototypischen Ablauf der „Methode des lauten Denkens“, der hier angegeben wird. Die Durchführung der Methode sollte in einer möglichst ungestörten, ruhigen Umgebung erfolgen. Das Individuum wird instruiert, bei der Problemlösung laut mitzusprechen und alles zu sagen, was ihnen durch den Kopf geht (für exakte Formulierungsvorschläge siehe (Van Someren et al., 1994, S. 43)). Bevor die eigentliche Problemstellung bekannt gegeben wird, kann bei in der Methode ungeübten Individuen eine „Aufwärmphase“ durchgeführt werden, in der etwa anhand der Lösung einer einfachen Schlussrechnung das Verbalisierung der Überlegungen zu deren Lösung geübt werden kann.

Während der Durchführung der Methode beschränkt sich der Untersuchungsleiter darauf, das Individuum an das Aussprechen seiner Gedanken zu erinnern, sobald dieses zu sprechen aufhört. Der gesamte Verlauf der Aufgabenbearbeitung wird mittels Audio- oder Video-Ausrüstung aufgezeichnet.

Nach Abschluss der Methode wird die gesamt Aufzeichnung transkribiert. Bei der Transkription muss auf höchste Exaktheit geachtet werden, auch Sprechpausen oder nichtverbale Geräusche des Individuums sind relevant. Liegt eine Videoaufzeichnung vor, so wird das Transkript um die beobachteten Tätigkeiten des Individuums ergänzt. Das fertige Transkript kann dem Individuum im Sinne der Retrospektion zur Kommentierung vorgelegt werden, bei der Kommentare oder Erklärungen angebracht werden können (aber als solche gekennzeichnet werden müssen).

Zur Auswertung der Ergebnisse der „Methode des lauten Denkens“ werden unterschiedliche Methoden herangezogen, im im Detail in (Van Someren et al., 1994) beschrieben sind. Da hier lediglich die Externalisierung selbst von Interesse ist, wird auf diese Methode an dieser Stelle nicht näher eingegangen.

Unterstützung

Eine technische Unterstützung der „Methode des lauten Denkens“ ist von den Entwicklern der Methode (Van Someren et al., 1994) grundsätzlich nicht vorgesehen. Lediglich zur Dokumentation der artikulierten Information wird eine Aufzeichnung mittels Video- oder Audio-Ausrüstung empfohlen. Diese Dokumentation ist notwendig, um eine möglichst vollständige Auswertung der Information zu gewährleisten und eine Abbildung auf eine strukturierte Externalisierung des zugrunde liegenden mentalen Modells zu ermöglichen.

(Senge, 1990) schlägt zur verbalen Externalisierung von mentalen Modellen einen Ansatz vor, der der „Methode des lauten Denkens“ nahe kommt, die Gedanken des Individuums aber verschriftlicht. Bei Einsatz der „left-hand column“⁶ wird ein Blatt Papier in zwei Spalten geteilt, wobei in der rechten Spalte eine Transkription der Handlungen bzw. der Konversation eines Individuums eingetragen wird. In der linken Spalte werden die Gedanken und handlungsmotivierenden Überlegungen eingetragen und den sichtbaren Handlungen in der rechten Spalte zugeordnet. Ein Nachteil dieser Methode ist, dass er – auch wenn er technisch unterstützt werden würde – nur im Nachhinein durchgeführt werden kann, um den eigentlichen Arbeitsablauf nicht zu unterbrechen.

Bewertung

Die „Methode des lauten Denkens“ (Van Someren et al., 1994) ist die einzige der gängigen Methoden zur Externalisierung mentaler Modelle, welche nicht auf eine graphische Repräsentationsform zurückgreift. Die externalisierende Person muss während der Aufgabenbearbeitung unmittelbar ihre kognitiven Prozesse und Denkmuster verbalisieren. Dies ist für viele Menschen ungewohnt und führt oft zu unvollständigen Repräsentationen. Detailliertes Nachfragen ist hier deshalb notwendig. Die gewonnenen Daten (etwa aus Audio- oder Videomitschnitten des Versuchsszenarios) werden strukturiert ausgewertet, kategorisiert und interpretiert. Hier liegt auch die Schwierigkeit des Verfahrens – in der Interpretation ist eine eindeutige Zuordnung zu bestimmten kognitiven Prozessen oft nicht möglich, die Repräsentation

⁶Eine detaillierte Beschreibung der Methode ist in (Senge et al., 1994) erschienen

des mentalen Modells bleibt unvollständig oder ist inkonsistent. Ifenthaler (2006, S. 28)

In einer informellen Variante ist die „Methode des lauten Denkens“ jedoch vor allem für den Einsatz in nicht komplexen Fällen von „Articulation Work“ geeignet (also etwa bei kleineren Änderungen im Arbeitskontext, die die Anpassung einzelner Tätigkeiten aber nicht die Adaption des gesamten Arbeitsablaufs benötigen). Dabei kann auf eine Aufzeichnung ggf. verzichtet werden, da die Durchführung der unmittelbaren Kommunikation dient und deren Ergebnisse nicht weiter interpretiert oder anderweitig verwendet werden müssen.

3.4.2. Strukturlegetechniken

Strukturlegetechniken sind Ansätze, in denen gelegte Strukturen zur Repräsentation von „Wissen“ eingesetzt werden. Die gelegten Strukturen (die im Wesentlichen aus Knoten und Kanten unterschiedlicher Bedeutung bestehen) bilden dabei die Zusammenhänge einzelner Konstrukte ab, wie sie die legende Person wahrnimmt. Der Prozess des Legens ist eine „*Rekonstruktion subjektiver Theorien*“ (Dann, 1992) und stellt eine „*[...] versteckende Beschreibung von Handlungen nicht aus der Perspektive eines außenstehenden Beobachters, sondern aus Sicht der handelnden Person, des Akteurs selber*“ (Dann, 1992) dar.

Konzept

Das Konzept der Strukturlegetechniken entstammt im Wesentlichen einem Forschungsprogramm zur Entwicklung von Ansätzen zur „rekonstruktiven Erhebung subjektiver Theorien“ (Dann, 1992). „Subjektive Theorien“ sind dabei im Wesentlichen den mentalen Modellen und Schemata gleichzusetzen⁷ (Kluwe, 1990) (zitiert nach (Huss, 2003)). Strukturlegetechniken sind nicht als reine Erhebungsmethoden zu sehen, sondern beeinflussen durch den Lege-Vorgang selbst die zu externalisierenden mentalen Modelle und bilden damit die Grundlage für eine mögliche Veränderung des Agierens im Arbeitskontext (Dann, 1992, S. 6). Die Grundidee von Strukturlegetechniken ist die freie Anordnung und Assoziation von Begriffen. Je nach Variante kann dies individuell oder in Gruppen, mit oder ohne Moderator bzw. Untersuchungsleiter geschehen. Der Prozess ist dann abgeschlossen, wenn die Beteiligten die Repräsentation als eine adäquate Abbildung ihrer Denkmodelle sehen. Vor allem in kooperativen Sitzungen ist dies mit Aushandlungs- und Abstimmungspro-

⁷ „Subjektive Theorien [...] sind nicht nur unmittelbar handlungserklärend, -rechtfertigend oder -leitend; d.h. sie beziehen sich über die unmittelbare Erklärung/Rechtfertigung etc. eigener Handlungen hinaus auf z.B. ganze Handlungskategorien [...]“ (Scheele und Groeben, 1988, S. 34)

zessen während der Repräsentation verbunden, was Strukturlegetechniken in der Durchführung potentiell aufwändig macht.

Strukturlegetechniken sind hinsichtlich der Art und dem Umfang der vorgegebenen Konstrukte nicht einheitlich aufgebaut. Es existieren Ansätze, in denen sämtliche Strukturelemente (also Konzepte und Arten von Beziehungen) vorgeben sind und die vom Externalisierenden „lediglich“ die Anordnung dieser Strukturelemente verlangen. Dem gegenüber stehen Strukturlege-Varianten, die weder die Konzeptklassen (und dementsprechend auch keine konkreten Konzepte) noch die Beziehungsarten vorgeben und deren Festlegung dem Externalisierenden überlassen (für einen Überblick über Varianten siehe (Ifenthaler, 2006, S. 29)). Im Fall der gängigen HSLT⁸ (Scheele und Groeben, 1988) wird ein zweistufiges Vorgehen gewählt, bei dem im ersten Schritt die Konzepte durch ein vorgegebenes Frageschema erhoben werden und im zweiten Schritt die Anordnung und Assoziation durchgeführt wird. Die Strukturen, die durch den Externalisierenden gebildet werden sind im Sinne von Stachowiaks „Allgemeiner Modelltheorie“ (Stachowiak, 1973) als „diagrammatische Modelle“ einzustufen (also ein im Normalfall graphisches, jedoch nicht ikonisches Darstellungsmodell).

Vorgehen

Je nach Variante von Strukturlegetechniken werden mehr oder weniger starke Vorgaben hinsichtlich des Ablaufs der Externalisierung gemacht. Der in der Literatur (z.B. (Ifenthaler, 2006)) als am elaboriertesten und etabliertesten bezeichnete Ansatz ist die Dialog-Konsens-Methodik, die im Rahmen der Heidelberger Strukturlegetechnik (Scheele und Groeben, 1988) zur Anwendung kommt.

Die Dialog-Konsens-Methodik sichert die Adäquatheit der während des Legeprozesses entstehenden mentalen Modelle durch laufende „kommunikative Validierung“ des Verständnisses ab. Um die kognitive Last zu reduzieren, wird dem eigentlichen Strukturlegeprozess eine Erhebungs-Phase vorgelagert, in der die relevanten Strukturelemente (Konzepte und Assoziationen) identifiziert werden.

In der Erhebungsphase werden mittels einem semistrukturierten Interview (exemplarischer Aufbau siehe (Scheele und Groeben, 1988)) werden Konzepte identifiziert, die für den jeweiligen Problembereich von Interesse sind. Die Identifikation erfolgt durch den Untersuchungsleiter auf Basis des Interview-Protokolls und nicht durch das externalisierende Individuum selbst. Das Individuum bestätigt oder erweitert in der Folge im Dialog mit dem Untersuchungsleiter die identifizierten Konzepte. Die Strukturierung der Konzepte erfolgt mittel vorgegebenen Relationen (die so wie die Konzepte als Kärtchen vorliegen). Die HSLT definiert insge-

⁸Heidelberger Strukturlegetechnik

3. Mentale Modelle

samt 20 Relationsarten und sieht keine Erweiterung derselben vor. Die Strukturierung wird sowohl vom externalisierenden Individuum als auch von Untersuchungsleiter unabhängig voneinander vorgenommen und dann im Rahmen eines Dialog-Konsens-Prozesses gegenübergestellt und das Verständnis abgeglichen. Ziel ist hier, dass der Untersuchungsleiter das mentale Modell des Individuums versteht. Scheele und Groeben (1988) schlagen vor, bei komplexen Sachverhalten die Konsensbildung über die Konzepte in einem separaten Dialog-Konsens-Prozess durchzuführen, bevor die Strukturierung vorgenommen wird.

Auch andere Strukturlegetechniken (siehe (Dann, 1992)) bleiben beim Konzept des Dialog-Konsenses und trennen zwischen der Phase der Konzepterhebung und der der Konzeptstrukturierung. Sie unterscheiden sich in der Zielsetzung der Externalisierung (und sind zum Teil nur für bestimmte Formen von mentalen Modellen geeignet) sowie im Grad der Vorstrukturierung (als inwieweit Konzepte und/oder Beziehungen bereits vorgegeben sind). Entsprechend der jeweiligen Offenheit bzw. Eingeschränktheit der Strukturlegetechnik ist die Phase der Konzept- (bzw. Beziehungs-)Sammlung mehr oder weniger stark ausgeprägt. Gemein ist allen Strukturlegetechniken, dass zwei dedizierte Phasen der Konzeptsammlung und der Konzeptstrukturierung durchgeführt werden, die in der Folge im Dialog-Konsens iterativ solange verfeinert werden, bis alle beteiligten Personen (also der Untersuchungsleiter und das externalisierende Individuum) mit dem Ergebnis zufrieden sind.

Unterstützung

Als technologische Unterstützung von Strukturlegetechniken wird in der Literatur mehrfach (u.a. bei (Huss, 2003) und (Ifenthaler, 2006)) die Software MaNET⁹ (Eckert, 1998) erwähnt¹⁰. Von den Entwicklern dieses Produkts wird dieses aber wiederholt als Software zur computerunterstützten Generierung von „Concept Maps“ (siehe Abschnitt) bezeichnet. Tatsächlich verschwimmen ob der fehlenden physischen Repräsentation des Modells (es wird ausschließlich am Rechner konstruiert) und dem offenen semantischen Konzept (im Gegensatz zur HSLT) die Grenzen zu „Concept Mapping“-Werkzeugen im Sinne der Entwickler dieses Ansatzes (Novak und Cañas, 2006). Ein ähnliches Bild zeigt sich bei (Mandl und Fischer, 2000), die bei der Unterstützung von Methoden zur „Strukturdarstellung“ auf „Concept-Mapping“-Werkzeuge verweisen.

Eine explizite Unterstützung des physischen Legeaspekts von Strukturlegetechniken wird in der Literatur nicht erwähnt. Aktuell existieren allerdings Bestrebungen, computerunterstützte „Concept Mapping“-Ansätze in den physischen Raum

⁹Mannheimer Netzwerk-Elaborations-Technik

¹⁰<http://www.marescom.net> (Abruf am 21.08.2009)

zu transferieren (Do-Lenh et al., 2009) (Tanenbaum und Antle, 2009). Diese Ansätze werden in Abschnitt 5.5.1 im Rahmen der Beschreibung der verwandten Arbeiten genauer betrachtet.

Bewertung

Strukturlegetechniken bedienen sich einer physischen Abbildung der mentalen Modelle durch die externalisierende Person. Sie zählen damit hinsichtlich des Ergebnisses zu den graphischen Verfahren zur Externalisierung mentaler Modelle. Konzeptuell besteht keine Einschränkung auf individuelles Externalisieren, das Verfahren kann auch in Gruppen angewandt werden. Die beteiligten Personen bilden Begriffsnetzwerke, die deren Handlungen zugrunde liegenden Annahmen und Modelle abbilden. Strukturlegetechniken werden in den gängigen Varianten durch Dialog-Konsens-Methoden unterstützt, in denen die Modelle in Interaktion zwischen dem Externalisierenden und dem Moderator bzw. Versuchsleiter entstehen. Grundsätzlich ist dies aber nicht notwendig und wird auch nicht in allen Strukturlege-Varianten angewandt.

Hinsichtlich der Auftrennung des Externalisierungsprozesses in zwei Phasen (Konzept-Sammlung und -Strukturierung) erscheint bei „Articulation Work“ eine fakultative Durchführung der ersten Phase möglich und angemessen. Durch die inhaltliche Offenheit von „Articulation Work“ sind viele Konstrukte nur in ihrem Kontext sinnvoll verständlich und müssen deshalb unmittelbar in die Struktur eingebettet werden oder werden erst aus dieser ersichtlich. Eine strikte Teilung in Sammlung und Strukturierung ist daher in diesem Anwendungsbereich fragwürdig. Bei der Modellierung komplexer Zusammenhänge scheint außerdem eine möglichst hohe Flexibilität der Repräsentationsform von Vorteil zu sein, um den Modellierungsprozess nicht zu behindern und eine Fokussierung auf den Modellierungsgegenstand zu ermöglichen (Goguen, 1993, S. 6). Dies wird im Kontext von Articulation Work als Wesentlich erachtet (Schmidt und Simone, 2000, S. 10).

Im Falle von „Articulation Work“ ist von einer wechselseitigen Abstimmung der mentalen Modelle der beteiligten Individuen auszugehen (obgleich es Szenarien geben kann, in der klassische Experten-Laien-Settings im Sinne eines unidirektionalen Wissenstransfers auftreten – diese werden hier jedoch als Spezialfall des allgemeinen, wechselseitigen Szenarios betrachtet). Dazu ist eine Auflösung der in der ursprünglichen Methode vorgesehenen strikten Trennung zwischen „Proband“ und „Untersuchendem“ hin zu einer gleichberechtigten Rolle aller Beteiligten notwendig. Zu untersuchen bleibt, ob die Rolle des Moderators und „Ermöglichers“ (im Sinne der Unterstützung bei der Werkzeugbenutzung), die ansonsten vom Untersuchungsleiter eingenommen wird, nach wie vor explizit wahrgenommen werden muss

(durch eine Person, die ansonsten nicht in den Dialog-Konsens-Prozess eingebunden ist).

Kritisch betrachtet wird die lange Durchführungsdauer der Externalisierungs-Prozesse, die eine nicht unwesentliche Belastung der Teilnehmer darstellt. Auch die Komplexität mancher Ansätze (etwa der HSLT mit ihren 20 unterschiedlichen Beziehungstypen) stellt eine nicht unwesentliche kognitive Belastung der Teilnehmer dar. Neuere Ansätze empfehlen zur Reduktion des Aufwandes den Einsatz von rechnerbasierten Werkzeugen, ohne dabei jedoch spezifischer zu werden. Ifenthaler (2006, S. 29f)

3.4.3. Concept Mapping

Concept Mapping (Novak und Cañas, 2006) ist eine Methode, in der semantische offene diagrammatische Modelle graphisch erstellt werden. Sie dienen der flexiblen Abbildung von Begriffen (Konzepten) und deren Zusammenhänge. Die erstellte Struktur entspricht einem Graphen mit Knoten, die die Konzepte repräsentieren und Kanten, die gerichtet oder ungerichtet die Beziehungen zwischen den Konzepten herstellen und durch Beschriftung zusätzlich spezifiziert werden können. Concept Mapping sollte ob der potentiellen Komplexität der entstehenden Modelle nach Novak und Cañas (2006) durch rechnerbasierte Werkzeuge unterstützt werden.

Konzept

Concept Maps sind graphische Strukturen, in denen durch Knoten und Kanten Begriffe und deren Zusammenhänge dargestellt werden. Die Begriffe („Konzepte“) werden dabei anhand des Themas der Concept Map ausgewählt, die zumeist in Form einer Fokus-Frage vorliegt. Ein Konzept ist nach (Novak und Cañas, 2006, S. 1) „*a perceived regularity in events or objects, or records of events or objects, designated by a label*“. Konzepte sind also allgemeine Aussagen über Phänomene oder Objekte, die durch einen Bezeichner beschrieben werden. Diese Bezeichner sind im Allgemeinen kurz und sollten 1-2 Worte umfassen.

Die Konzepte werden untereinander mit Beziehungen verbunden, wobei die Kombination aus zwei oder mehreren Konzepten und einer Beziehung als „Proposition“ bezeichnet wird. „Propositions“ sind nach (Novak und Cañas, 2006, S. 1) „*statements about some object or event in the universe, either naturally occurring or constructed*“. Beziehungen können grundsätzlich gerichtet oder ungerichtet sein und müssen durch eine Beschriftung („linking word“) mit beliebiger Bedeutung versehen werden. Nach (Novak und Cañas, 2006) enthalten Concept Maps meist eine hierarchische

Struktur, in der die allgemeinen Konzepte am oberen Rand angeordnet sind und nach unten hin immer spezifischer werden. Daneben gibt es „cross-links“, die Beziehungen außerhalb der hierarchischen Struktur darstellen und Konzepte zueinander in Beziehung setzen, die in unterschiedlichen Bereichen der Concept Map stehen. Die grundlegende Struktur einer Concept Map ist in Abbildung 3.3 als Concept Map dargestellt.

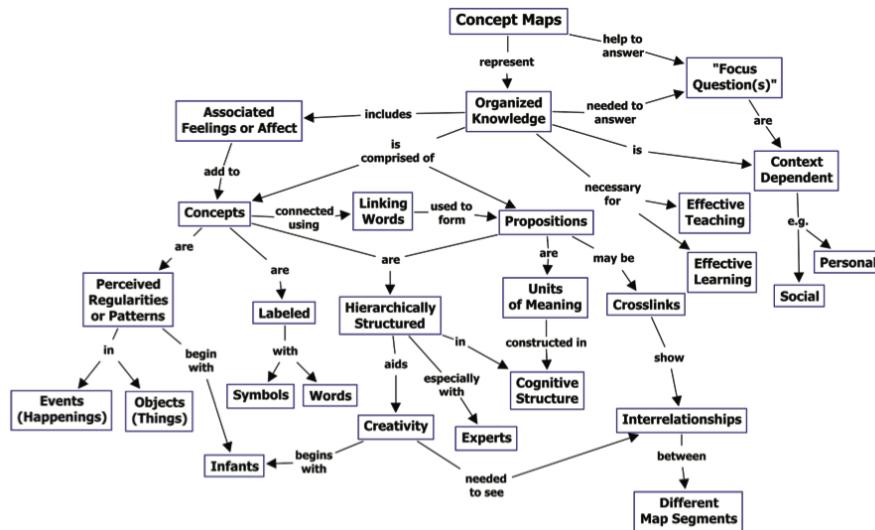


Abbildung 3.3.: Struktur einer Concept Map (entnommen aus (Novak und Cañas, 2006, S. 2))

Concept Maps werden verwendet, um exploratives Lernen zu unterstützen. In diesem Fall bilden Individuen die ihnen bewussten Zusammenhänge der Realität in der Concept Map ab und erschließen bei der individuellen oder kooperativen Erstellung den Problembereich vollständiger, was zur Entwicklung eines umfassenderen Verständnisses beiträgt. Nach (Novak und Cañas, 2006) können Concept Maps auch verwendet werden, um (implizites) Expertenwissen abzubilden (die Autoren beziehen sich hier auf (Nonaka und Takeuchi, 1995). Im Wesentlichen ermöglichen Concept Maps damit das Externalisieren sowohl von Laien- als auch Expertenmodellen im Sinne von (Seel, 1991) und unterstützen auch die Weiterentwicklung von Laienmodellen hin zu ausgereifteren Erklärungs- oder Expertenmodellen. Damit ist eine grundsätzliche Eignung für den Einsatz im Rahmen von auf der Externalisierung von Modellen basierender „Articulation Work“ gegeben.

Vorgehen

Novak und Cañas (2006) schlagen vor, bei der Konstruktion einer Concept Map mit der Festlegung einer Fokus-Frage zu beginnen. Die Fokus-Frage muss klar formuliert sein und spezifisch auf das Problem oder den Sachverhalt eingehen, der in der Concept Map repräsentiert werden soll. Die Fokus-Frage dient nicht nur der Festlegung des Gegenstands der Concept Map sondern auch deren Abgrenzung nach außen (d.h. dass sie so spezifisch sein muss, dass Abweichungen vom intendierten Gegenstand der Concept Map erkannt werden können).

Im nächsten Schritt werden die relevanten Konzepte gesammelt. Novak und Cañas (2006) sprechen von 15-25 Konzepten, die im ersten Durchlauf maximal verwendet werden sollten. Diese Konzept können grob entsprechend ihrer Abstraktheit vorsortiert werden, um die Erstellung der Concept Map im nächsten Schritt zu erleichtern.

Die vorläufige Concept Map, die im nächsten Schritt erstellt wird, basiert auf den hierarchischen Zusammenhängen zwischen den gesammelten Konzepten. Zwischen diesen wird in der Folge nach „cross-links“ gesucht. Alle identifizierten Beziehungen müssen benannt werden. Im Zuge dieser ersten Herstellung von Beziehungen ergeben sich im Normalfall weitere Konzepte, die in die Concept Map aufgenommen werden müssen. Dies erfolgt im Zuge eines erneuten Durchlaufs durch den beschriebenen Prozess. Nach Novak und Cañas (2006) benötigt eine Concept Map mindestens drei dieser Durchläufe, um ausreichende Qualität erreichen zu können.

Unterstützung

Novak und Cañas (2006) erwähnen, dass Concept Mapping mittels Haftnotizen auf Papier oder Whiteboards durchgeführt werden kann, empfehlen aber, ein rechner-basiertes Werkzeug – die CMapTools (Cañas et al., 2004) – einzusetzen, dass den Erstellungsprozess unterstützt und den Umgang mit der entstehenden Komplexität erleichtert.

Dieses Werkzeug ermöglicht neben der Unterstützung des Mapping-Prozesses (d.h. der Nachverfolgung des Prozesses und der Möglichkeit, einzelne Schritte rückgängig zu machen) auch eine erweiterte Abbildung des Concept Map selbst. Zweites umfasst unter anderem auch die Einbindung von externen Ressourcen (Dateien am Rechner), was von Novak und Cañas (2006) als Wesentlich zur Einbettung der Concept Map in den Kontext des Problemumfelds angesehen wird.

Bewertung

Concept Mapping ist ein Ansatz zur computer-basierten Strukturierung und Visualisierung von Konzept-Netzwerken (Novak und Cañas, 2006). Durch die Rechner-

unterstützung ergeben sich Vorteile hinsichtlich der Flexibilität der Darstellung und der Archivierung der Modelle. Konzeptuell werden wie bei Strukturlegetechniken Begriffsnetzwerke gebildet, wobei die methodische Hinterlegung bei Concept Mapping Ansätzen nicht so variantenreich und detailliert ausgeführt ist.

Durch die digitale Repräsentation ist eine Concept Map leichter ohne Konsequenzen zu manipulieren, da Änderungen jederzeit rückgängig gemacht werden können. Dies ermöglicht Experimente mit dem Modell und erlaubt dem Externalisierenden eine umfassendere Ergründung und Reflexion der Modelle. Kritisch wird jedoch die im Gegensatz zu Strukturlegetechniken fehlende Unmittelbarkeit der Externalisierung betrachtet – jeder Externalisierung-Prozess muss am Rechner umgesetzt werden und setzt damit Kompetenz im Umgang mit diesem Medium voraus. Ifenthaler (2006, S. 30f)

3.5. Externalisierung im Rahmen von Articulation Work

Basierend auf den Schlussfolgerungen, die Ifenthaler (2006) hinsichtlich der Eignung der beschriebenen Methoden zur Externalisierung von mentalen Modellen zieht, scheinen jene Ansätze, die auf der Bildung diagrammatischer Modelle basieren besser für die Unterstützung expliziter „Articulation Work“ geeignet zu sein als Methoden, die auf einer rein natürlichsprachlichen Repräsentation aufbauen. Dies liegt vor allem in der höheren Abstraktion begründet, die die externe Repräsentation als interindividuellen Ankerpunkt für Kommunikation besser geeignet macht. Dies deckt sich mit den Aussagen von Sarini und Simone (2002a), Herrmann et al. (2002), Raposo et al. (2004) oder Jørgensen (2004), die aus Sicht von „Articulation Work“ für die Verwendung von (diagrammatischen) Modellen zur Unterstützung argumentieren.

Betrachtet man nun die beiden Vertreter der auf diagrammatischen Modellen aufbauenden Methoden – Strukturlegetechniken und Concept Mapping –, so zeigt sich hinsichtlich der Eignung zum Unterstützung von „Articulation Work“ kein eindeutiger Vorteil für eine der beiden Methoden. Vielmehr weisen beide in diesem Kontext Vor- und Nachteile auf. Hier wird deshalb versucht, die Vorteile von Strukturlegetechniken – im Wesentlichen die Unmittelbarkeit der physischen Repräsentation – mit jenen von Concept Mapping – der Flexibilität der Modellierung sowie der Möglichkeit der Unterstützung des Modellierungsprozesses durch Computersysteme – zu vereinen.

Dabei wird auf das für „Articulation Work“ besser geeignete methodische Vorgehen von „Concept Mapping“ zurückgegriffen, während die Modellierungsumgebung an das Setting von „Strukturlegetechniken“ angepasst wird.

3.5.1. Durchführungsrahmen

Der Rahmen, in explizite „Articulation Work“ mit Unterstützung von externalisierten Modellen durchgeführt wird, ist an den Aufbau von Strukturlegetechniken angelehnt. Eine wesentliche Eigenschaft ist hierbei die physische Modellierungsoberfläche, auf der das Modell mittels real vorhandener und unmittelbar manipulierbaren Elementen aufgebaut wird.

Im Sinne der Abstimmung unterschiedlicher Sichten muss eine kooperative, nicht exklusive Manipulierbarkeit des Modells gewährleistet sein. Das Modell selbst ist – orientiert an der Offenheit der Repräsentation bei „Concept Mapping“ – weder in der Art der Elemente noch der Beziehungen eingeschränkt.

3.5.2. Vorgehen

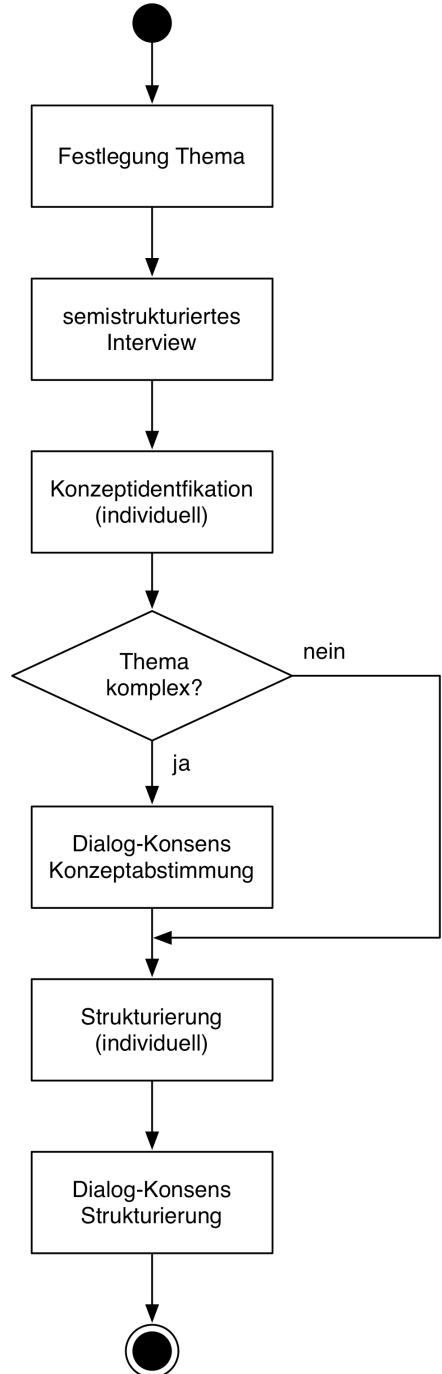
Sowohl im Bereich der Strukturlegetechniken als auch im „Concept Mapping“ wird vorgeschlagen, den initialen Modellierungsprozess in zwei Phasen – Konzeptsammlung und Konzeptstrukturierung – zu teilen und in der Folge das Modell iterativ so lange zu verändern bzw. zu erweitern, bis alle Beteiligten mit der Lösung zufrieden sind (im Bereich der Strukturlegetechniken wird die als „Dialog-Konsens“ bezeichnet, im „Concept Mapping“ spricht man von „Revisions“ des Modells, die erstellt werden müssen). Beide Abläufe sind abstrahiert in Abbildung 3.4 dargestellt.

Das „Dialog-Konsens“-Vorgehen nach Scheele und Groeben (1988) ist stark reglementiert und in den einzelnen Schritten mit definierten Methoden bzw. Vorgehensvorschriften hinterlegt. „Concept Mapping“ ist hier offener und erscheint damit für die Anwendung im Rahmen von „expliziter Articulation Work“ besser geeignet. Dies liegt am eher informellen Durchführungsrahmen von „expliziter Articulation Work“ begründet, deren Ausgestaltung individuell verschieden ist und zwischen den Beteiligten (im gängigsten Fall implizit) ausgehandelt wird. Ziel ist hier, den Artikulations-Prozess zu unterstützen und nicht, ihn zu formalisieren und in vorgegebene Ablauf-Grenzen zu pressen.

Auch die zweiphasige Durchführung des Modellierungsprozesses muss unter diesem Gesichtspunkt hinterfragt werden. Die Unterteilung in zwei Phasen erscheint bei der Beschreibung von konzeptionellen Modellen sinnvoll. Begründet liegt dieses in der sich Bei Modellen von Abläufen (für AW relevant, in den obigen Ansätzen nicht wirklich betrachtet) nur bedingt sinnvoll (Aktivitäts-Konzepte werden im

3.5. Externalisierung im Rahmen von Articulation Work

Dialog-Konsens-Prozess
nach HSLT
(Scheele & Gröben 1988)



Concept Mapping
nach Novak (1991)

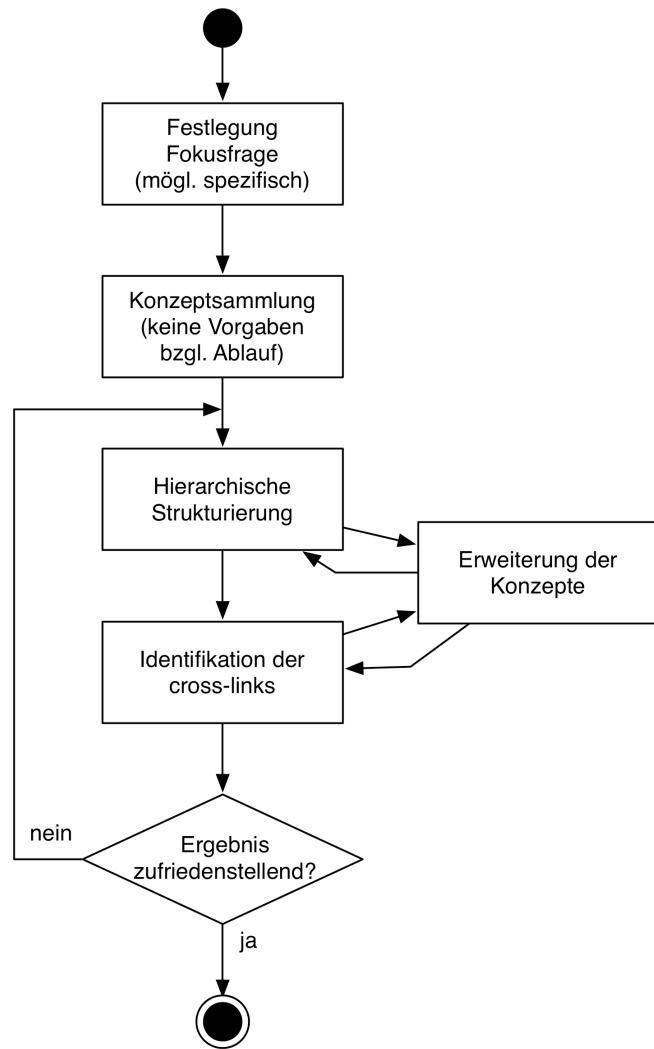


Abbildung 3.4.: Externalisierung mentaler Modelle mittels Strukturlegetechniken und Concept Mapping

3. Mentale Modelle

Normalfall bereits in deren kausalen Abfolge externalisiert und dann bzw. parallel mit zusätzlichen Konzepten hinterlegt) - entspricht der Unterscheidung zwischen Alignment of Workflow und Alignement of Meaning bei Simone.

3.6. Fazit

„Mentale Modelle“ sind eine Erklärungskonzept für

Teil II.

Unterstützung

Einleitung

Basierend auf den diese Arbeit motivierenden Grundlagen, die in den letzten Kapiteln beschrieben wurden, wird in diesem Teil auf die konkreten Maßnahmen zur Unterstützung von Articulation Work eingegangen. Ziel dieses Teils ist es, sowohl das hier vorgeschlagene Vorgehen bei der Unterstützung expliziter Articulation Work als auch die Unterstützung, die ein Werkzeug dabei leisten kann, umfassend darzustellen.

Auf Grundlage der Methodik, die im Kontext von Concept Mapping und Strukturlegetechniken vorgeschlagen wird und unter Berücksichtigung der Anforderungen, die aus dem inhärent kollaborativen Anwendungsszenario abgeleitet werden können, muss das Vorgehen zur Durchführung von expliziter Articulation Work festgelegt werden.

In Rahmen der Festlegung des Vorgehens werden auch jene Aspekte identifiziert, in denen Unterstützung durch technische Werkzeuge sinnvoll und notwendig ist. Die Anforderungen, die sich aus diesen Aspekten ableiten lassen, bilden die Grundlage für die Konzeption und Umsetzung eines Werkzeugs, das diese Unterstützung bietet. Die technischen Details der Implementierung dieses Werkzeugs und die zugrunde liegenden konzeptionellen und technologischen Grundlagen bilden den Kern dieser Arbeit.

Der Aufbau dieses Teils folgt dem eben umrissenen inhaltlichen Vorgehen. Am Ende des Kapitels 3 wurde die Methodik zur Unterstützung expliziter Artikulation Work beschrieben. Aus diesem werden im folgenden Kapitel 4 jene Bereiche identifiziert, in denen eine technologische Unterstützung notwendig ist und die Anforderungen an ein Werkzeug abgeleitet, das diese Unterstützung bietet. Die vier folgenden Kapitel beschäftigen sich mit der Umsetzung des Werkzeugs. Dabei wurde der Implementierungsstand beschrieben, mit dem das Werkzeug im Großteil der Evaluierungen eingesetzt wurde. Jene Weiterentwicklungen, die in den letzten Phasen der Evaluierung umgesetzt wurden, betreffen ausschließlich die Interaktion mit dem Werkzeug und hatten keinen Einfluss auf die technische Realisierung. Die Veränderungen gründeten jeweils auf aus vorhergehenden Anwendungen gebildeten Hypothesen und wurde zur Überprüfung derselben prototypisch umgesetzt. Die Beschreibung der Veränderung zum in der Folge beschriebenen Referenzstand wird dementsprechend im Rahmen der Beschreibung der Evaluierungsergebnisse in den Kapiteln 11 bis 13 vorgenommen.

Kapitel 5 beschäftigt sich mit den konzeptionellen Grundlagen des Forschungsgebiets "Tangible Interfaces", das die Basis für die technische Umsetzung bildet. Kapitel 6 beschäftigt sich mit jenen Technologien und Softwarekomponenten, die für die Informationseingabe in das technische System verwendet werden. Dabei wird auch

3. Mentale Modelle

auf die konkrete Interaktion der Benutzer mit dem System eingegangen. Kapitel 7 beschreibt die Ausgabeseite des technischen Systems und behandelt die Umsetzung des Informationsflusses vom System zu den Benutzern. Letztendlich wird in Kapitel 8 beschrieben, welche Maßnahmen zu Sicherung der Ergebnisse der expliziten Articulation Work getroffen werden müssen und welche Möglichkeiten der technischen Umsetzung bestehen bzw. gewählt wurden.

4. Anforderungen an ein Werkzeug

Physische Abbildung beliebiger diagrammatischer Modelle Ein Werkzeug zur Unterstützung von Strukturlegetechniken muss das grundlegende Konzept der Methodik vollständig unterstützen. Es muss möglich sein, Konzepte auf einer Modellierungs-Oberfläche zu platzieren und zueinander in Beziehung zu setzen. Der gesamte Modellstatus muss visuell auf der Oberfläche erkennbar sein.

Kollaborative und unmittelbare Manipulierbarkeit des Modells Zur Unterstützung von expliziter „Articulation Work“ muss das Werkzeug kollaborative Strukturlege-Prozesse erlauben. Es muss möglich sein, das gelegte Modell simultan zu erweitern oder zu verändern.

Nicht vorgegebene Semantik der Modellierungselemente Wie oben bereits argumentiert, sind zur Unterstützung von expliziter „Articulation Work“ vor allem Varianten von Strukturlegetechniken geeignet die keine Vorgaben hinsichtlich der zu verwendenden Konzepte und Verknüpfungen machen. Das Werkzeug muss dementsprechend die Offenheit bieten, beliebige Klassen von Konzepten und Verknüpfungen zu definieren (z.B. Klasse „organisationale Rolle“) und von diesen beliebige Instanzen zu bilden und zu benennen (z.B. Instanz „Geschäftsführer“). Gleichzeitig muss sichergestellt werden, dass die festgelegte Semantik im Modell mit abgebildet wird und nicht verloren geht.

Unterstützung der iterativen Aushandlung des Modells Im Sinne der Unterstützung der Dialog-Konsens-Methodik sind ist der Austausch über das Modell durch das Werkzeug zu unterstützen. Vor allem muss es möglich sein, Anmerkungen über Konsens oder Dissens über einzelnen Modellteile oder das gesamte Modell explizit mit in die Repräsentation aufzunehmen.

Persistente Ablage des Modells und Möglichkeit zur Rekonstruktion Die persistente Ablage eines Modells (z.B. als digitale Repräsentation) und Werk-

4. Anforderungen an ein Werkzeug

zeugunterstützung zur Rekonstruktion eines abgelegten Modells erlaubt die Wiederaufnahme eines unterbrochenen Strukturlegeprozesses bzw. die Reflexion und Anpassung bereits erstellter Modelle zu einem späteren Zeitpunkt.

Ermöglichung experimenteller Veränderungen am Modell Es muss möglich sein, das Modell experimentell zu verändern und ggf. zu einem früheren stabilen Modellzustand zurückzukehren. Dies erlaubt eine konsequenzlose Erkundung von Lösungsräumen und unterstützt damit den Dialog-Konsens-Prozess. Das Werkzeug muss also stabile Modellzustände erfassen und deren Rekonstruktion unterstützen.
@

Verknüpfung mit digitalen Ressourcen Die Einbindung von digitalen Ressourcen (Dateien, Hyperlinks,...) ermöglicht die Einbindung des Modells in den organisationalen Kontext und erleichtert so einerseits die Verständnisbildung und ermöglicht andererseits die Verwendung der Repräsentation als unmittelbare Handlungsanleitung mit Verknüpfungen zu den betroffenen Arbeitsgegenständen.

Bearbeitung von beliebig komplexen Modellen Komplexe Modelle enthalten oft eine große Anzahl von Konzepten und viele Verknüpfungen. Das Werkzeug muss das Modell in einer Form darstellen, die dessen Erfassung und Manipulation ermöglicht, ohne die Repräsentierenden kognitiv zu sehr zu belasten.

1

¹By recording and replaying the authoring process, navigable history can re-situate an author after a gap in the authoring process. Similarly, in a collaborative authoring process, an author can play through the events since his/her last authoring session to quickly determine the activity of the other authors. Finally, in many situations, information becomes harder to interpret as its context changes over time. By returning to the state of the information space at the time of authoring, disambiguation of the information may become possible. For the reader who is not also the writer of the hypertext there are additional uses of navigable history. A reader replaying the author's writing process can gain insight into the motivation of the author and have a greater understanding of the author's writing style. Such an understanding is important in collaborative work and in other contexts, like education and literary analysis. (Shipman und Hsieh, 2000)

5. Grundlagen der Implementierung

Zur Umsetzung des Werkzeugs wurde – wie in Kapitel XY gefordert – ein „Tangible Tabletop Interface“ verwendet. Tabletop Interface zeichnen sich im Generellen dadurch aus, dass im Gegensatz zu handelsüblichen Rechnern nicht nur die Software sondern auch die Hardware applikationsspezifisch ist und nicht für beliebige Anwendungen eingesetzt werden kann. Die Hardware bildet dabei einen Teil oder die gesamte Benutzungsschnittstelle sowohl ein- als auch ausgabeseitig ab. Im speziellen Fall eines „Tangible Tabletop Interfaces“ basiert der Benutzerinteraktion auf der Verwendung physischer Bausteine („Tokens“), die auf der physischen Oberfläche des Interfaces manipuliert werden. Diese Interaktionsform wird von Tabletop Interfaces ergänzt, die die Benutzerinteraktion ausschließlich auf Gesten bzw. Berührungen der Oberfläche abbilden (horizontal verbaute „Touch“ bzw. „Multi-Touch-Displays“).

Tabletop Interfaces wurden Mitte der 1990er-Jahren in den Arbeiten von Ishii & Ullmer erstmals vorgestellt. Auch die erste Anwendung, die sich mit Modellierung auf Tabletop Interfaces beschäftigt, stammt aus dieser Zeit. Mit dem Fortschreiten der technologischen Entwicklung ist heute ein Status erreicht, in dem mit Hilfe generischer Identifikations-Frameworks schnell und ohne großen Aufwand Applikationen mit „tangiblen“ Eingabekanälen erstellt werden können. Zur Zeit noch im Prototypenstatus befinden sich Systeme, die sich mit Möglichkeiten des tangiblen Informationsoutputs beschäftigt. Der Rückkanal vom Rechner zum Benutzer wird heute zumeist durch die Projektion von Inhalten auf die Arbeitsoberfläche umgesetzt.

In den folgenden Abschnitten wird die historische Entwicklung von Tabletop Interfaces sowie der aktuelle Stand der Entwicklung im Anwendungsbereich dieser Arbeit betrachtet. Es werden dabei die grundlegenden Konzepte und Eigenschaften der jeweiligen Arbeiten betrachtet und das Potential hinsichtlich der Umsetzung von in Kapitel XY identifizierten Anforderungen an das hier entwickelte Werkzeug betrachtet.

5.1. Historische Entwicklung von Tangible Interfaces

Wellner (1993) und Suzuki und Kato (1995) werden als jene Arbeiten angesehen, die die Vision von alternativen Benutzungsschnittstellen für Computer maßgeblich geprägt haben. „Alternativ“ bedeutet in diesem Zusammenhang die Verwendung von Ein- und Ausgabekanälen, die sich von den herkömmlichen Werkzeugen wie Tastatur und Maus bzw. Bildschirmen insofern unterscheiden, als dass sie eine unmittelbare Interaktion mit der digital repräsentierten Information ermöglichen und deren Zustand in der realen Welt wiederspiegeln.

Der Begriff der „Tangible“ bzw. „Graspable Interfaces“ – also der „berührbaren“ oder „begreifbaren“ Benutzungsschnittstellen — stammt ebenfalls aus der Mitte der neunziger Jahre des zwanzigsten Jahrhunderts. Fitzmaurice et al. (1995) werden im Allgemeinen als die ersten betrachtet, die den Begriff des „Graspable User Interfaces“ prägen und damit die Manipulierbarkeit digitaler Information durch physische Mittel beschreiben. Fitzmaurice (1996) präzisiert später den Begriff durch die Abgrenzung zwischen (herkömmlichen, maus-, tastatur- und bildschirmbasierten) zeitlich gemultiplexten Schnittstellen, bei denen der Informationsaustausch zwischen Benutzer und System über einen Kanal zeitlich hintereinander erfolgt und den (neuartigen, berührbaren) räumlich gemultiplexten Schnittstellen, bei denen mehrere Kanäle gleichzeitig zur Interaktion zwischen Benutzer und System verwendet werden können.

Der Begriff des „Tangible User Interfaces“ (TUI) wurde kurz danach bzw. parallel dazu von Ishii und Ullmer (1997) eingeführt. Ishii und Ullmer verfolgen dabei bei der Definition den umgekehrten Weg und sprechen von einer „Augmentation der realen Welt durch eine Kopplung von digitaler Information and physische Objekte“¹.

Die erwähnte Kopplung wurde von Beginn an bidirektional verstanden. Dies bedeutet, dass sowohl der Zustand der realen Welt die digitale Welt beeinflusst als auch umgekehrt die digitale Welt auf die reale Welt zurück wirkt. Der Fokus der Entwicklung lag in den ersten zehn Jahren der Entwicklung des Feldes klar auf den erstgenannten Aspekt. Erforscht wurde im Wesentlichen die Manipulation von digitaler Information mittels physischer Werkzeuge, der Zustand der digitalen Welt wurde zumeist ausschließlich dargestellt, beeinflusste aber die reale Welt nicht weiter. Die Kopplung des physischen Zustands der realen Welt an digitale Information (etwa die Veränderung der Form oder Position von physischen Interface-Objekten durch die Ergebnisse einer im Rechner durchgeföhrten Berechnung) wurde nur in Ausnahmefällen angesprochen (zumeist unter der Bezeichnung „Ambient Interface“

¹“augment the real physical world by coupling digital information to everyday physical objects and environments”(Ishii und Ullmer, 1997)

5.2. Konzeptualisierung und Klassifikation von Tangible Interfaces

(Gross, 2003)), strukturiert untersucht wurde dieser Aspekt erstmals von Patten und Ishii (2007). Nach wie vor verzichtet ein Großteil der vorgeschlagenen Systeme auf die automatisierte Manipulation der realen Welt und beschränkt sich auf die Ausgabe der Information über einen optischen Kanal.

Technologisch gründet sich das Gebiet der Tangible Interfaces auf den Entwicklungen im Bereich des Ubiquitous Computing (Weiser, 1991) und der Augmented Reality (Azuma, 1997). Auch heute ist die Abgrenzung noch eher konzeptionell zu sehen. Die technischen Mittel der Umsetzung sind vor allem hardwareseitig nach wie vor nicht spezifisch den einzelnen Gebieten zuzuordnen, lediglich softwareseitig ist eine Spezialisierung durch die Entwicklung dedizierter Software-Frameworks zu erkennen.

Die Anwendungsbereiche von Tangible Interfaces sind historisch breit gestreut, lassen jedoch eine Fokussierung auf die Unterstützung von kreativen und planenden Prozessen erkennen. Ein wichtiger Anwendungsbereich war schon in den ersten Jahren aktiver Forschung im Bereich der Tangible Interfaces das Gebiet der Unterstützung von Lernprozessen (Resnick et al., 1998). Zurückgreifend auf die Ideen Pestalozzis, Montessoris (Montessori, 2005) und Piagets (Piaget, 1976) schlagen Resnick et al. (1998) physische Objekte mit Mitteln der IT anzureichern um Spielzeuge („*digital manipulatives*“) zu schaffen, mit denen Kinder interagieren können und sich so physikalische bzw. systemtheoretische Konzepte spielerisch zu erschließen. Die Objekte fungieren dabei einerseits als Baumaterialien und „Eingabekanal“ für den Rechner, andererseits gleichzeitig auch als „Ausgabe“- bzw. Feedbackkanal über die Wirkung die in der digitalen Welt (etwa in einer Simulation) mit der Eingabekonstellation erzielt wurde. Die Erkenntnisse der Entwicklungen und empirischen Untersuchungen in diesem Bereich wurden sowohl in der frühkindlichen Bildung (Zuckerman et al., 2005) als auch in organisationalen Lernprozessen zur Erhebung und Abstimmung von Strukturen und Abläufen in Unternehmen (The LEGO Group, 2002) eingesetzt.

5.2. Konzeptualisierung und Klassifikation von Tangible Interfaces

Die Entwicklung des Forschungsgebiets der „Tangible Interfaces“ wurde von mehreren konzeptionellen Arbeiten maßgeblich beeinflusst. Die dort vorgeschlagenen Erklärungsmodelle definieren das Gebiet und grenzen es gegenüber anderen Forschungsbereichen ab. Sie dienen außerdem als Grundlage für die strukturierte Be trachtung und Konzeption konkreter Tangible Interfaces. Im Folgenden werden die-

5. Grundlagen der Implementierung

se konzeptionellen Arbeiten beschrieben und auf deren Kontext und Spezifika eingegangen.

Zur strukturierten Betrachtung von Tangible Interfaces ist es außerdem notwendig, jene Dimensionen zu identifizieren, an denen sich konkrete Systeme einordnen und unterscheiden lassen. Die Ausprägungen dieser Dimensionen ergeben ein Begriffssystem, dass bei der Aufbereitung von unterschiedlichen Tangible Interfaces sowie deren Vergleich helfen kann. Die hier vorgestellten Ansätze tragen verschieden detailliert und von unterschiedlichen Standpunkten aus zu dieser Thematik bei. Sie werden in der Folge strukturiert dargestellt und in Kapitel 9 auf das in dieser Arbeit entwickelte System angewandt, um so das System-Design aus konzeptioneller Sicht zu reflektieren und potentielle Verbesserungs- und Erweiterungsmöglichkeiten zu identifizieren.

Allgemein ist anzumerken, dass eine Vielzahl von Ausdrücken im sich entwickelnden Forschungsgebiet mehrfach belegt wurden und/oder nicht eindeutig definiert sind. Im Folgenden werden die Ausdrücke der jeweiligen Autoren übernommen. Eine Interpretation bzw. Abbildung auf die Terminologie anderer Autoren wird nur vorgenommen, wo sie im jeweiligen Artikel explizit angeführt wurde. In der Zusammenfassung dieses Abschnitts wird versucht, die unterschiedlichen Terminologien nochmals zusammenzufassen und einen Satz an Ausdrücken festzulegen, der im Folgenden für diese Arbeit Anwendung findet.

Am Ende jeder Beschreibung sind in einer tabellarischen Zusammenfassung jeweils die zentralen Konzepte und Beiträge des Ansatzes angeführt. Die Tabellen weisen einheitlich Einträge zu folgenden Themen auf:

Kategorien Kategorien von Tangible Interfaces, die im Beitrag identifiziert werden (inkl. Unterscheidungsmerkmal zur Kategoriebildung)

Konzepte Konzepte, die bei der Betrachtung bzw. beim Design eines Tangible User Interfaces zur Anwendung kommen

Eigenschaften Eigenschaften, die ein Tangible User Interface bzw. dessen Komponenten aufweisen können

PD-Brücke Aussagen, die der Beitrag zur Natur oder Ausgestaltung der Brücke zwischen physischer und digitaler Welt macht

Wird in einem Beitrag keine Aussage zu einem oder mehreren dieser Themen gemacht, so ist dies explizit durch „—“ angeführt.

Die Ansätze, die in dieser Betrachtung berücksichtigt wurden, sind in der Reihenfolge der zeitlichen Entstehung angeordnet. Eine Beschreibung und Visualisierung der kausalen Zusammenhänge und Bezugnahmen erfolgt in der Zusammenfassung der Ergebnisse in Abschnitt 5.2.13. Bei der Beschreibung der Arbeiten wird nicht auf sämtliche vorgestellten Aspekte eingegangen. Es wird vielmehr auf jene Teile

eingegangen, die sich mit der Konzeptualisierung oder Klassifikation von Tangible User Interfaces beschäftigen. Die berücksichtigten Ansätze wurden im Rahmen einer umfassenden Literaturstudie identifiziert und sind im Einzelnen:

- Bricks (Fitzmaurice et al., 1995)
- Graspable User Interfaces (Fitzmaurice, 1996)
- Tangible Bits (Ishii und Ullmer, 1997)
- Containers, Tokens und Tools (Holmquist et al., 1999)
- Tangible Object Meaning (Underkoffler und Ishii, 1999)
- Das MCRpd Interaktions-Modell (Ullmer und Ishii, 2000)
- Tokens and Constraints (Ullmer, 2002)
- Degree of Coherence (Koleva et al., 2003)
- Tokens and Constraints zur Spezifikation (Shaer et al., 2004)
- Kategorien von TUI²-Anwendungen (Klemmer et al., 2004)
- Tangible User Interfaces Taxonomy (Fishkin, 2004)
- Tangible Bits: Beyond Pixels (Ishii, 2008)

5.2.1. Bricks

Mit „Bricks“ stellen Fitzmaurice et al. (1995) das erste als „Graspable User Interface“ bezeichnete System vor. Dieses bildet die Grundlage für weitere Arbeiten der Autoren (z.B. (Fitzmaurice, 1996)), die in der Folge noch behandelt werden (siehe Abschnitt 5.2.2).

Konzeptionell spannen die Autoren am Ende der Arbeit einen „Design Space“ auf, der mögliche Eigenschaften und Parameter eines Tangible User Interfaces definiert und auch deren möglichen Ausprägungen festlegt. Dabei werden einerseits technische Aspekte des Interfaces abgedeckt, andererseits wird aber auch der Aufbau und die Verwendung des TUI berücksichtigt. Die Ausprägungen sind dabei rein beschreibender Natur und werden nicht für eine wie auch immer geartete Bewertung oder Klassifikation genutzt. Das Beschreibungsschema ist stark auf auf Interfaces mit aktiv untereinander kommunizierenden Komponenten („Bricks“) abgestellt, die zur Manipulation einer digitalen Anwendung verwendet werden und auf einer physischen Oberfläche bedient werden. Passive bzw. rein informationstragende Elemente ohne Manipulationsfunktion werden nicht berücksichtigt. Dies liegt in der Natur der von den Autoren entwickelten Anwendung und dem zum Zeitpunkt der Erstellung herrschenden Mangel an alternativen Systemen begründet.

²Tangible User Interface

Brick's internal ability Haben die physischen Elemente Mechanismen, die zur Darstellung oder Manipulation von Information benutzt werden können? Diese Mechanismen können physischer oder elektronischer Natur sein. (Mögliche Ausprägungen: „inert“ – „simple expressions“ – „smart“)

Input & Output Welche Eigenschaften der physischen Objekte werden zur Eingabe von Information erfasst bzw. zu Ausgabe verwendet? (Keine Ausprägungen, Aufzählung der Eigenschaften für Input und Output)

Spatially aware Kann ein Brick seine Umgebung und/oder andere Bricks wahrnehmen? (Mögliche Ausprägungen: „unaware“ – „mutually aware“ – „aware of surroundings“)

Communication Wie kommunizieren Bricks untereinander bzw. mit einer eventuell vorhandenen Hintergrund-Infrastruktur (Mögliche Ausprägungen: „Wireless“ – „Tethered“ – „Grid board“)

Interaction time span Wie lange dauert eine Interaktion der Benutzer mit dem System bei der Erfüllung einer vorgegebenen Aufgabe? (Mögliche Ausprägungen: „quick (within seconds)“ – „interaction cache (seconds - minutes)“ – „long-term (days, months, years)“)

Bricks in use at same time Wieviele physische Elemente werden simultan verwendet? (Mögliche Ausprägungen (höhere Werte stehen für Größenordnung): „1“ – „2“ – „5-10“ – „50-100“)

Function assignment Wie oft und mittels welchem Vorgehen wird Bricks Funktionalität zugeordnet? (Mögliche Ausprägungen: „permanent“ – „programmable“ – „transient“)

Interaction representations Werden physische und virtuelle Objekte gleichzeitig verwendet um den Systemzustand darzustellen bzw. zu manipulieren? (Mögliche Ausprägungen: „all physical“ – „mix, physical dominates“ – „balanced mix“ – „mix, virtual dominates“ – „all virtual“)

Physical & Virtual layers Werden physische und virtuelle Interaktions-Schichten separat oder kohärent dargestellt? (Mögliche Ausprägungen: „direct (superimposed)“ – „indirect (separated)“)

Bond between physical & virtual layers Wie stark sind die physischen mit den virtuellen Objekten gekoppelt? (Mögliche Ausprägungen: „tightly coupled (real time sync)“ – „loosely coupled (batch sync)“)

Operating granularity Wie groß ist der Refenzrahmen für die Interaktion mit den physischen Elementen und wie genau werden die Positionen der Elemente aufgelöst? (Mögliche Ausprägungen: „Desktop (fractions of inches accuracy)“ – „Room (inch accuracy)“ – „Building (room accuracy)“)

Operating surface type Werden die physischen Elemente auf einer fix vorgegebenen, unveränderlichen Oberfläche verwendet oder kann sich die Oberfläche dynamisch verändern (z.B. Information anzeigen)? (Mögliche Ausprägungen: „static (e.g. paper)“ – „dynamic (e.g. screen)“)

Operating surface texture Welche Auflösung oder Textur besitzt die Arbeitsoberfläche, auf der die physischen Elemente bewegt werden? (Mögliche Ausprägungen: „discrete (e.g. grid board)“ – „continuous, smooth movement“)

Betrachtungsgegenstand ist bei diesem Ansatz das gesamte TUI, auf speziifische Eigenschaften einzelnen Komponenten wird nicht separat eingegangen. Die physischen Elemente (hier „Bricks“) werden nicht im Detail betrachtet, sondern nur hinsichtlich ihrer Verwendung im Gesamtsystem und ihrer technischen Einbindung berücksichtigt.

Kategorien	—
Konzepte	Brick
Eigenschaften	<i>Gesamtsystem</i> : Brick's internal ability, Input & Output, Spatially aware, Communication, Interaction time span, Bricks in use at same time, Function assignment, Interaction representations, Physical & virtual layers, Bond between physical & virtual layers, Operating granularity, Operating surface type, Operating surface texture
PD-Brücke	—

5.2.2. Graspable User Interfaces

Fitzmaurice legt in jener Arbeit, in der der Begriff des „Graspable User Interfaces“ erstmals ausführlich eingeführt wird (Fitzmaurice, 1996), auch Eigenschaften fest, anhand deren sich die „Graspability“ einer Benutzungsschnittstelle zeigt und beurteilen lässt. Im wesentlichen handelt es sich dabei um einen auf die für die Beurteilung der „Graspability“ relevanten reduzierten Satz an Eigenschaften, die bereits in (Fitzmaurice et al., 1995) angeführt wurden (siehe Abschnitt 5.2.1). Die Beurteilung erfolgt auf einer generischen Skala mit Ausprägungen von „niedrig“ bis „hoch“, wobei „hohe“ Werte in mehreren Eigenschaften auf eher hohe „Graspability“ hinweisen. Die Eigenschaften beziehen sich auf Tangible User Interfaces, die lediglich Werkzeuge zur Manipulation von digitalen Daten besitzen, jedoch keine physische Repräsentation von Information aufweisen. Diese Einschränkung ist wiederum auf die historische Entwicklung des Gebiets und den ersten Versuchen, GUI-Paradigmen in den physischen Raum zu übersetzen, zurückzuführen.

Space-multiplexing Ändern sich die Zuordnungen zwischen physischen Elementen und digitalen Funktionen über die Zeit oder existiert eine permanente Zuordnung? (Mögliche Ausprägungen: kontinuierlich von „transient, always reassigned“ – „permanent, never reassigned“)

Concurrency Ist die gleichzeitige Ausführung mehrere Operationen mit mehreren physischen Elementen (abhängig oder unabhängig voneinander) möglich und vorgesehen? (Mögliche Ausprägungen: „1“ – „occasionally 2“ – „2“ – „3“ – „more than 3“)

Physical form Lässt sich aus der physischen Form auf die ausgeführte Funktion schließen oder sind die Werkzeuge generisch für unterschiedliche Funktionen verwendbar? (Mögliche Ausprägungen: „generic“ – „specific“)

Spatially aware Erfasst und berücksichtigt das System die Positionen, Orientierung und/oder Nähe der physischen Objekte bei der Interpretation der Interaktion? (Mögliche Ausprägungen: „unaware“ – „aware“)

Spatial reconfigurability Kann das System in unterschiedlichen physischen Kontexten betrieben werden oder kann es ausschließlich ortsgebunden eingesetzt werden? (Mögliche Ausprägungen (bezogen auf TUIs aus einzelnen, handhabbaren Objekten): „permanent location“ – „stationary“ – „track“ – „tethered“ – „free-ranging (rapid layout)“)

Fitzmaurice stellt mit diesen Eigenschaften ein Framework für die Einordnung eines TUI als Gesamtsystem zur Verfügung und geht nicht auf einzelne Komponenten ein. Die Eigenschaften beziehen sich auf das Design des Systems und ignorieren dessen Verhalten. Die ersten beiden und die letzte Eigenschaft sind mit einer kontinuierlichen Skala hinterlegt, *physical form* und *spatially aware* sind binäre Eigenschaften, die entweder gegeben sein können oder nicht.

Kategorien	—
Konzepte	—
Eigenschaften	<i>Gesamtsystem</i> : Space-multiplexing (transient – permanent), Concurrency (1 – mehr als 3), Physical form (generic, specific), Spatially aware (unaware, aware), Spatial reconfigurability (permanent location – free-ranging)
PD-Brücke	—

5.2.3. Tangible Bits

Ishii und Ullmer (1997) stellen in ihrer Arbeit die Idee der „Tangible Bits“ vor, die die virtuelle Welt mit der realen Umwelt verknüpfen soll. Dabei wird digitale Informa-

tion mit realen Objekten oder Phänomenen gekoppelt und so „tangibel“ gemacht. Die Autoren unterscheiden drei grundsätzliche Kernkonzepte, mit Hilfe derer die Idee umgesetzt werden kann:

Interactive Surfaces, bei denen beliebige Oberflächen in der realen Welt (etwa Wände oder Schreibtische) zu aktiven Schnittstellen zur virtuellen Welt werden

Coupling Bits and Atoms, wo physischen Objekten Information zugeordnet wird, so dass die realen Objekte zu Trägern von und Schnittstellen zu digitaler Information werden

Ambient Media, bei deren Einsatz Information über die Umgebung der Benutzer vermittelt wird (z.B. mittels Veränderung der Beleuchtung), ohne die eigentliche Tätigkeit zu unterbrechen.

Die Autoren ordnen Tangible Bits an die Schnittstelle zwischen den Forschungsgebieten „Ubiquitous Computing“ und „Augmented Reality“ ein. „Ubiquitous Computing“ (Weiser, 1991) beschäftigt sich mit Anwendungen, in denen Computer nicht mehr als solche wahrnehmbare Werkzeuge eingesetzt werden, sondern in der Umgebung integriert sind und von Benutzern nicht mehr bewusst wahrgenommen sondern nur noch als Teil der Alltagswelt benutzt werden. Tangible Bits erben von dieser Forschungsrichtung die Idee der physischen, natürlichen Interaktion zwischen Benutzern und Rechner, unterscheiden sich aber insofern, als das nach wie vor ein Interface zur virtuellen Welt vorhanden ist, diese also zum Teil noch bewusst wahrgenommen wird. In diesem Aspekt ähneln Tangible Bits den Ideen die aus der „Augmented Reality“ Forschung stammen. Augmented Reality beschäftigt sich mit Methoden, die die reale Welt mit digitaler Information nahtlos ergänzen bzw. anreichern. Vor allem im Bereich der Ausgabetechnologien werden bei Tangible Bits viele aus dem „Augmented Reality“-Bereich stammende Konzepte eingesetzt.

Nach dieser grundlegenden Begriffsklärung stellen die Autoren Prototypen vor, die das Konzept der „Interactive Surfaces“ als auch der „Ambient Media“ umsetzten. Erwähnenswert ist im Zusammenhang mit dieser Arbeit der Prototyp „metaDESK“ (Ullmer und Ishii, 1997), eine interaktive Oberfläche in Form eines Tisches, auf der die klassischen Interaktionselemente eines GUI³ in die reale Welt transferiert wurden. Dabei wurden die GUI-Elemente auf TUI-Elemente abgebildet⁴:

- Windows werden auf „Lenses“ abgebildet, also Linsen, die Information zu realen, physischen Elementen anzeigen.
- Icons werden in der physischen Welt als „Phicons“ abgebildet und sind im Wesentlichen physischen Objekte, die Information repräsentieren.

³Graphical User Interface

⁴dies stellt im Übrigen die historisch erste Erwähnung des Begriffs „Tangible User Interface“ dar

5. Grundlagen der Implementierung

- Menüs werden durch „Trays“ repräsentiert, in denen Phicons an unterschiedlichen Stellen abgelegt werden können, wobei jeweils eine spezifische Operation an die Ablageposition gebunden ist.
- Handles (Elemente zur Manipulation von GUI-Objekten) werden durch „Phandles“ abgebildet. Dies sind im Wesentlichen Phicons, die zur Eingabe von Information verwendet werden können.
- Widgets (Kontroll- und Steuerelemente) werden durch „Instruments“ abgebildet, welche Phicons sind, die zur Steuerung der jeweiligen Applikation dienen.

Ohne hier weiter auf die Beispiele der Autoren einzugehen, ist doch das den Prototypen zugrunde liegende Designkonzept erwähnenswert, etablierte Metaphern aus der realen oder digitalen Welt für die Benutzerinteraktion zu verwenden. In der Diskussion ihrer Ergebnisse identifizieren die Autoren die Thematik der Metaphern, die die Brücke zwischen realer und virtueller Welt schlagen, als eine der interessantesten offenen Fragen für weitere Forschung auf dem Gebiet. Als Schlussfolgerung ihrer Erfahrungen mit den erstellten Prototypen schlagen sich außerdem vor, „optischen“ Metaphern besondere Aufmerksamkeit zu schenken, da diese für den Brückenschlag besonders geeignet wären.

Als optische Metaphern verwenden die Autoren vor allem Beleuchtung, Schattenwurf und Linsen. Sie bilden diese realen Phänomene auf die Schnittstelle zwischen digitaler und realer Welt ab, so dass z.B. digitale Beleuchtung Information sichtbar macht, die in „unbeleuchteten“ Gebieten fehlt, dass reale Objekte digitale Schatten werfen können und so zusätzliche Information transportieren oder dass eine digitale Linse verwendet wird, um reale Objekt „näher“ zu betrachten und Zusatzinformation einzublenden. Bei der Verwendung von Metaphern wird darauf hingewiesen, dass es wichtig wäre, auf ein realitätsgetreues Verhalten der digital augmentierten Artefakte zu achten, um eine nahtlose Integration der digitalen Information in die reale Welt zu ermöglichen.

Kategorien	Interactive Surfaces, Coupling Bits and Atoms, Ambient Media (Einordnung nach Anwendungsfall)
Konzepte	Lenses, Phicons, Tray, Phandles, Instruments
Eigenschaften	—
PD-Brücke	zentraler Aspekt: Verwendung etablierter Metaphern

5.2.4. Containers, Tokens und Tools

Holmquist et al. (1999) legen ihre Arbeit als konzeptionelle Betrachtung von interaktiven Systemen an, in denen physische Objekte verwendet werden, um auf digitale Information zuzugreifen bzw. diese zu manipulieren. Das Einteilungsschema,

das die Autoren vorschlagen, basiert auf der Art und Weise, in der Information an diese physischen Objekte gebunden ist. Grundsätzlich unterschieden sie zwischen *Containern*, *Tokens* und *Tools*. Eine exakte Abgrenzung bzw. eindeutige Zuordnung zu einer Kategorie ist dabei nicht immer möglich oder sinnvoll.

Der hier vorgestellte Ansatz fokussiert auf die physische Interaktion als Eingabemedium, auf den Aspekt der Informationsausgabe wird nicht eingegangen. Dies ist für das Verständnis der folgenden Beschreibungen im Kontext der späteren historischen Entwicklung wichtig und muss bei der Anwendung dieses Ansatzes berücksichtigt werden.

Containers

Als Container werden all jene Objekte bezeichnet, an die beliebige digitale Information gebunden werden kann. Eine Container ist also ein unspezifisches physisches Objekt in einem Tangible User Interface. Sein Aussehen oder andere physische Eigenschaften lassen keine Aussage über die Art der angebundenen Information bzw. die Information selbst zu.

Beliebige physische Objekte können als Container agieren, sofern sie die Möglichkeit bieten, Information in bzw. auf ihnen abzulegen oder wenn sie durch eine Infrastruktur eindeutig identifizierbar sind, so dass Information an sie gebunden werden kann. Container agieren somit ausschließlich als physische Informationsträger und können verwendet werden, um Information zwischen Systemen zu transportieren.

Ein typisches Beispiel ist die Verwendung eines Füllfederhalters als Container, an den beliebige Information gebunden werden kann, um diese von einem Ort zum anderen transportieren zu können. Der Füllfederhalter steht in keinem direkten Zusammenhang mit der angebundenen Information, aus seinem Erscheinungsbild oder seinen Eigenschaften kann nicht auf die angebundene Information geschlossen werden.

Tokens

Tokens sind physische Objekte, deren äußeres Erscheinungsbild bzw. deren Eigenschaften in irgendeiner Weise mit der durch sie repräsentierten Information zusammenhängen. Das physische Objekt und die angebundene Information sind nicht mehr voneinander unabhängig, sondern stehen in einem konzeptionellen Zusammenhang. Die äußere Form oder andere physische Eigenschaften dienen oft als Hinweis auf die angebundenen Informationsart oder stehen sogar in Zusammenhang mit der konkret angebundenen Information.

5. Grundlagen der Implementierung

Ein typisches Beispiel für ein Token wäre ein Buch, an das über eine eindeutige Identifikation (etwa ein RFID⁵-Tag) der jeweilige Text oder Zusatzmaterial gebunden wird. Das physische Buch steht dabei in direktem Zusammenhang mit der angebundenen digitalen Information.

Tools

Tools sind physische Elemente, die nicht Information, sondern Funktionen repräsentieren. Die Anwendung von Tools hat dabei nicht unbedingt physische Auswirkungen, jene digitalen, virtuellen Objekte, auf die das Tool angewandt wird, werden aber entsprechend der Funktion des Tools manipuliert.

Beispiele für Tools sind physische Objekte, die zur Auswahl digitaler Objekte dienen oder Objekte wie Linsen, deren Anwendung zusätzliche Information zu anderen Containern oder Tokens abruft.

Zugriff auf und Interaktion mit Tokens und Containern

Der Zugriff auf die Information, die an ein Token oder einen Container gebunden ist, erfolgt über *Information Faucets* (also „Informations-Zapfhähne“ oder „Armaturen“). Diese Faucets sind aktive Komponenten (im Gegensatz zu Tokens und Containern, die im Allgemeinen passive Komponenten sind, also keine dedizierte Elektronik enthalten). Deren Aufgabe besteht darin, aus Tokens oder Containern, die in deren Reichweite gelangen, die angebundene Information zu extrahieren und auszugeben. Faucets können auch dazu verwendet werden, den Zugriff auf Information einzuschränken. So kann die Ausgabe von Information an eine bestimmte Kombination von Tokens oder Containern gebunden werden oder von einem bestimmten Aufenthaltsort abhängig gemacht werden.

Die Anbindung von Information an ein Token oder einen Container kann ebenfalls eingeschränkt sein, bzw. ist im Fall von Tokens per Definition durch den notwendigen Zusammenhang zwischen physischem Element und Information eingeschränkt. Neben dieser konzeptionell notwendigen Einschränkung können auch weitere Regeln geprüft werden oder z.B. die Bindung zwischen Objekt und Information statisch (d.h. unveränderbar) gespeichert werden.

⁵Radio Frequency Identification

Kategorien	—
Konzepte	Container, Token, Tool, Faucet
Eigenschaften	—
PD-Brücke	Zugriff auf an physische Elemente gebundene Information nur via Faucets, kein direkter Zugriff vorgesehen

5.2.5. Tangible Objects Meaning

Im Rahmen der Entwicklung einer „Interactive Surface“ zur Stadtplanung („Urp“) stellen Underkoffler und Ishii (1999) ein Kontinuum von möglichen Bedeutungen bzw. Verwendungszwecken der physischen Objekte eines TUI vor. Die Ausprägungen auf der Achse unterscheiden sich in der Stärke der Abstraktion der verwendeten Objekte von ihren Gegenstücken in der realen Welt. Im Zentrum der Achse stehen nicht abstrahierte Objekte, die in Struktur und Verhalten ihren realen Entsprechungen ähneln.

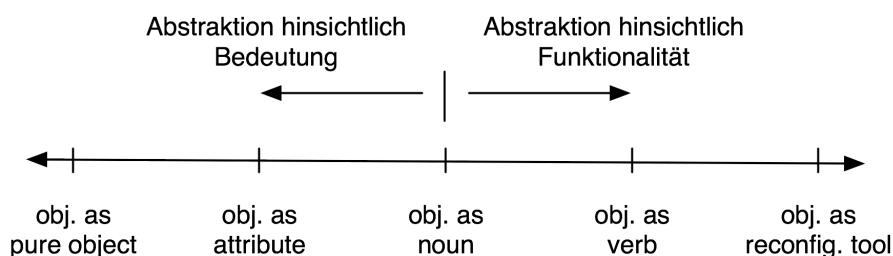


Abbildung 5.1.: Bedeutung von Objekten in TUIs (adaptiert übernommen aus (Underkoffler und Ishii, 1999))

Object as noun Derartige Objekte sind Platzhalter für Objekte der realen Welt, denen sie in Form und Verhalten weitgehend entsprechen. Sie stehen für das reale Objekt und werden in dessen Sinne verwendet.

Object as verb Die Bedeutung betreffender Objekte reduziert sich auf deren Funktionalität, die Objekte sind nicht mehr Teil der Repräsentation des Systemzustandes sondern werden dazu verwendet, diesen (entsprechend ihrer Funktion) zu verändern.

Object as reconfigurable tool Das Objekt wird funktional vollständig von seiner eigentlichen Natur abstrahiert verwendet. Die physischen Eigenschaften stehen nicht mehr in Beziehung zu seiner Verwendung zur Manipulation des Systemzustandes. Die Funktionalität ist dynamisch zuweis- bzw. auswählbar.

Object as attribute Objekte werden auf eine ihrer Eigenschaften reduziert. Nur diese wird verwendet, um den Systemzustand abzubilden. Mögliche Eigenschaften sind z.B. Form, Farbe oder Gewicht des Objekts.

Object as pure Object Das Objekt wird zum reinen Informationsträger, bei dem die Information nicht in Bezug zur physischen Form oder den Eigenschaften des Objekts steht.

Die Achse ist dabei als konzeptioneller Ring zu verstehen, der sich an den beiden Enden wieder schließt. Einem Objekt, dessen physischen Eigenenschaften keine Rolle im TUI mehr spielen, kann beliebig Inhalt oder Funktionalität zugewiesen werden, wodurch die Grenze zwischen linkem und rechtem Ende der Achse verschwimmt.

Kategorien	—
Konzepte	Object as pure object, Object as attribute, Object as noun, Object as verb, Object as reconfigurable tool
Eigenschaften	—
PD-Brücke	abhängig von der Art der Tokens

5.2.6. Das MCRpd Interaktions-Modell

Basierend auf früheren Arbeiten (siehe Abschnitt 5.2.3) entwickeln Ullmer und Ishii (2000) ein Modell zur Beschreibung der Interaktion mit Tangible User Interfaces. Es handelt sich bei dieser Arbeit um den ersten Ansatz, der sich diesem Bereich aus Sicht der Systemstruktur nähert und nicht ausschließlich eine reine Klassifikation nach bestimmten Merkmalen eines Systems vornimmt.

Die Autoren grenzen TUIs von GUIs insofern ab, als das TUIs über eine nahtlose Integration zwischen Repräsentation des Systemzustandes und dessen Kontrolle aufweisen (im Gegensatz zu GUIs, bei denen der Systemzustand über einen grafischen Kanal ausgegeben wird und über andere Kanäle, etwa Tastatur und Maus, kontrolliert wird). Mit der Forderung nach nahtloser Integration von Repräsentation und Kontrolle – also im Wesentlichen einer Einheit von Eingabe- und Ausgabe-Kanälen – wird ein eher striktes Verständnis von TUIs vorgeschlagen. Als Tangible User Interface kann ein System demnach nur bezeichnet werden, wenn es Ein- und Ausgabe kohärent über einen Kanal führt.

Diese Verständnis setzten die Autoren in der Folge in einem Interaktionsmodell für TUIs um, dass die Struktur eines Tangible User Interfaces konzeptionell beschreibt. Das Modell wird dabei analog zum MVC⁶-Modell konzipiert, in dem „View“ und „Controller“ strikt getrennt betrachtet werden. Entsprechend der enge-

⁶Model-View-Controller

5.2. Konzeptualisierung und Klassifikation von Tangible Interfaces

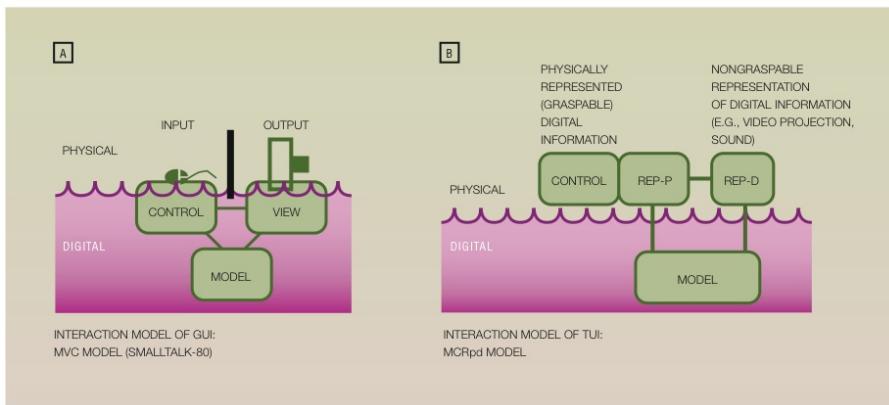


Abbildung 5.2.: Interaktionsmodelle für GUI und TUI (übernommen von Ullmer und Ishii (2000))

ren Kopplung zwischen Repräsentation und Kontrolle wird die Bezeichnung MCRpd⁷-Modell vorgeschlagen (siehe Abbildung 5.2). Anhand der Gegenüberstellung zwischen MVC- und MCRpd-Modell wird die Unterscheidung zwischen GUI und TUI deutlich. Beiden Ansätzen ist gemein, dass der Systemzustand im Rechner durch ein *Model* dargestellt wird. Unterschiede zeigen sich in der Art der Manipulation und Manifestation dieses *Models*. Bei GUIs ist Eingabe und Ausgabe strikt getrennt. Die Manipulation des Systemzustandes erfolgt durch die Komponente *Control*, in der physische Kontrollgeräte eine Veränderung des Zustandes erlauben. Die Kontrollgeräte sind dabei im Allgemeinen generisch, d.h. unabhängig vom konkreten Anwendungsfall (also etwa Maus oder Tastatur). Erst durch digitale Kontroll-Elemente wird die Ein- und Ausgabe applikationsspezifisch angepasst. Die Ausgabe erfolgt durch die Komponente *View*, wobei auch in diesem Fall ein generisches Ausgabegerät (etwa ein Bildschirm) verwendet wird. Im Falle eines TUI existiert keine Trennung zwischen Ein- und Ausgabe. Das *Model* manifestiert sich durch eine *Representation* in der realen Welt. Diese *Representation* hat eine physische Komponente (*REP-P*) und eine digitale Komponente (*REP-D*), die miteinander verknüpft sind. Die digitale Komponente der Repräsentation umfasst dabei all jene Information, die nicht durch physische, berührbare Elemente dargestellt wird (etwa Projektion, Audio, ...). Sie steht dabei jedoch nicht für sich selbst sondern ist immer von einer physischen Repräsentations-Komponente abhängig bzw. dieser zugeordnet. Die physischen Komponenten (*REP-P*) spielen insofern eine zentrale Rolle, als dass ihnen auch die *Control* zugeordnet ist und über sie damit der Systemzustand manipuliert werden kann. Die physischen Komponenten des Ausgabe-Kanals fungieren also zugleich als Instanzen des Eingabe-Kanals.

⁷Model-Control-Representation (physical and digital)

5. Grundlagen der Implementierung

Basierend auf diesem Ansatz identifizieren die Autoren vier Kern-Charakteristika von Tangible User Interfaces, die sich jeweils an den physischen Komponenten zeigen:

- Physikalische Repräsentationen sind mit digitaler Information gekoppelt
- Physikalische Repräsentationen enthalten Mechanismen zur Kontrolle des Systems
- Physikalische Repräsentationen sind in der Wahrnehmung der Benutzer mit digitalen Repräsentationen gekoppelt
- Der physische Zustand eines Tangible Interfaces stellt die Kernaspekt des Systemzustandes dar

Aufbauend auf diesen Eigenschaften entwickeln die Autoren in der Folge ein Schema, das unterschiedliche Ansätze bei der Konzeption eines Tangible Interfaces abdeckt. Als zentrale Ansätze werden „spatial“ (räumliche), „relational“ (relationale) und „constructive“ (konstruierende) Ansätze unterschieden. Räumliche Ansätze nutzen die Anordnung der physischen Elemente in einem Referenzrahmen, um den Systemzustand zu repräsentieren bzw. zu manipulieren. Bei relationalen Ansätzen ist die räumliche Anordnung unwesentlich, lediglich die Beziehungen zwischen den Elementen codieren den Systemzustand. Konstruierende Ansätze sind zwischen den beiden erstgenannten Ansätzen einzurordnen, da sie Beziehungen zwischen Elementen durch eine räumliche Anordnung der Elemente zueinander abbilden. Systeme, in denen Information lediglich in der Zuordnung zu einem physischen Element codiert wird und nicht in der Beziehung zwischen den Elementen, werden in die Kategorie der „associative“ (assoziativen) Ansätze eingeordnet.

Als zusätzliche Gestaltungsdimension führen die Autoren die Konzeption der physischen Repräsentationen an, die bereits in (Ullmer und Ishii, 1997) eingeführt wurde. Unterschieden wird hier zwischen „iconic“ und „symbolic representations“, wobei ikonische Elemente einen Bezug zu dem jeweils repräsentierten Objekt der realen Welt haben (etwa ein Bild einer Person), während diese Möglichkeit der Zuordnung bei symbolischen Elementen nicht gegeben ist (etwa bei der Repräsentation einer Person durch ein rotes Rechteck). Aus einer umfassenden Betrachtung der zum Zeitpunkt der Erstellung des Artikels verfügbaren Tangible User Interfaces schließen die Autoren, dass ikonische Repräsentationen vorrangig in räumlichen und assoziativen Ansätzen zum Einsatz kommen, während symbolische Repräsentationen eher bei relationen oder konstruierenden Ansätzen anzutreffen sind.

Kategorien	spatial, relational, constructive, associative (Einordnung nach Art der Informationsrepräsentation)
Konzepte	Model, Rep-P, Rep-D, Control
Eigenschaften	<i>Rep-P</i> : iconic, symbolic
PD-Brücke	Informationsrepräsentation immer an ein physisches Objekt gebunden. Manipulation der Information erfolgt mittels dem gleichen Objekt

5.2.7. Tokens und Constraints nach Ullmer

Der Token+Constraint-Ansatz wurde von Ullmer (2002) erstmals vorgestellt und Ullmer et al. (2005) aktualisiert veröffentlicht. Ullmer et al. gehen dabei erstmals von dem Grundsatz aus, das ein TUI zwei grundlegende Arten von physischen Komponenten enthält: Objekte, die digitale Information repräsentieren, und Objekte, die die Interaktion mit Computersystemen bzw. die Manipulation des Systemzustands ermöglichen.

In ihren weiteren Ausführungen identifizieren die Autoren zwei grundlegende Arten von TUIs, die sich historisch herausgebildet hätten. Einerseits existieren „interactive surfaces“, bei denen physische Objekte benutzt werden, um Information auf einer aktiven Oberfläche (zumeist projiziert oder durch einen Bildschirm realisiert) zu manipulieren. Andererseits werden „constructive assemblies“ genannt, deren physische Objekte ohne umgebende Infrastruktur ausschließlich durch reale oder logische Verbindungen zwischen ihnen den Systemzustand ausdrücken. Die Autoren definieren nun eine dritte Art von Systemen und platzieren diese konzeptionell zwischen den beiden zuvor genannten Kategorien. Mit „tokens and constraints“ werden Systeme mit zwei unterschiedlichen Arten von physischen Elementen eingeführt. „Tokens“ stehen für Objekte, die digitale Information repräsentieren. „Constraints“ werden verwendet, um (digitale) Funktionalität auf diese Tokens anzuwenden (siehe Abbildung 5.3).

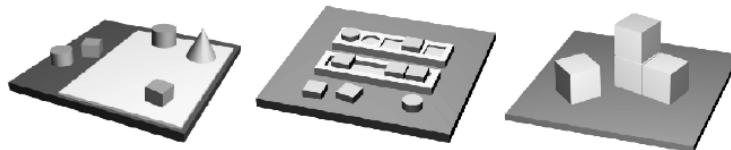


Abbildung 5.3.: Arten von Tangible User Interfaces – von links nach rechts: interactive surfaces, tokens+constraints, constructive assemblies (übernommen aus Ullmer et al. (2005))

5. Grundlagen der Implementierung

In der Folge beschreiben Ullmer et al. die Details des Tokens+Constraints-Ansatzes. Sie gehen dabei zuerst auf die Benutzer-Interaktion in einem Tokens+Constraints-basierten System ein. Diese ist immer in zwei Phasen unterteilt, wobei in der ersten Phase „associate“ Tokens einem oder mehreren Constraints zugeordnet werden. In der zweiten Phase „manipulate“ werden die Tokens im Kontext der Constraints manipuliert, wodurch die digitalen Daten, die an das Token gebunden sind, geändert werden.

Danach widmen sich die Autoren der Abbildung digitaler Information bzw. deren Manipulation auf physische Elemente und definieren vier grundlegende Arten von Beziehungen, die Information abbilden sein können:

- die absolute Position eines Tokens in Bezug zu einem Constraint
- die relative Position eines Tokens in Bezug zu einem Constraint
- die absolute Position eines Tokens in Bezug zu anderen Tokens
- die relative Position eines Tokens in Bezug zu anderen Tokens

Unter „Position“ sind dabei auch andere spatiale Parameter eines Tokens (wie die Orientierung) zu verstehen.

Nach einer Reihe von Betrachtungen der konzeptionellen Hintergründe und Beispielen kommen die Autoren zu einem weiteren relevanten Punkt, in dem Sie ausführen, wie bzw. warum Tangible Interfaces für Benutzer leichter verständlich sein können als traditionelle GUI-basierte Systeme. Bezugnehmend auf Bellotti et al. (2002) versuchen Sie Antworten auf fünf von diesen Autoren gestellten Fragen zu finden, die für jede Benutzungsschnittstelle geklärt werden müssen. An dieser Stelle stehen nun nicht die Antworten von Ullmer et al. im Mittelpunkt des Interesses, sondern die Fragen, die einen konzeptionellen Rahmen für das Design einer Benutzungsschnittstelle liefern. Die Fragen stammen aus der Arbeit von Bellotti et al. (2002):

Address Wie weiß das System, dass Benutzer mit ihm und nicht mit anderen (Systemen oder Personen) interagiert?

Attention Wie bemerken Benutzer, wenn das System auf eine Interaktionsanforderung reagiert?

Action Wie weiß das System, welchem Objekt der Befehl der Benutzer gilt?

Alignment Wie wissen Benutzer, dass das System ihren Befehl korrekt verstanden und ausgeführt hat?

Accident Wie werden Missverständnisse zwischen den Benutzern und dem System aufgelöst?

Ullmer et al. beantworten diese Fragen für den Tokens+Constraints-Ansatz jeweils aus konzeptioneller und technologischer Sicht.

Am Ende des Artikels geben die Autoren Varianten des Token+Constraints-Ansatzes an. Durch die Festlegung auf die notwendige Physikalität von Tokens und Constraints ist der Design-Raum eingeschränkt, obwohl die konzeptionellen Überlegungen auch breitere Anwendung finden können. Aus diesem Grund variieren die Autoren ihren Ansatz und geben Alternativen an, die den Grundüberlegungen entsprechen aber unter Umständen nicht alle Vorteile des Kernansatzes aufweisen:

- Verwendung von visuellen, graphischen Constraints und physischen Tokens
- Verwendung von physischen Constraints für graphische Token
- Verwendung von physischen Constraints für nicht-massive Tokens (z.B. Flüssigkeiten)
- Verwendung von Tokens und Constraints in durch die Benutzer anpassbaren Größen
- Alternative Semantik bei der Abbildung zwischen physischer und digitaler Welt

Abschließend wird betont, dass ausgereifte TUIs selten in „Reinform“ auftreten, d.h. dass sie nicht exakt einer Kategorie wie „interactive surface“ oder „tokens+constraints“ zugeordnet werden können.

Kategorien	interactive surfaces, tokens+constraints, constructive assemblies
Konzepte	Tokens, Constraints
Eigenschaften	<i>Token</i> : Form, absolute und relative Position in Bezug zu Constraints oder anderen Tokens
PD-Brücke	Codiert in Tokens und in den Beziehungen zwischen Tokens und Constraints

5.2.8. Degree of Coherence

Koleva et al. (2003) schlagen ein Framework vor, dass die Klassifikation von Tangible Interfaces ermöglicht und so das Verständnis der möglichen Verknüpfungen zwischen physischer und digitaler Welt vertiefen soll. Im Gegensatz zu dem von Ullmer und Ishii (2000) mit dem MCRpd-Modell vorgeschlagenen relativ strikten Verständnis eines Tangible Interfaces öffnen Koleva et al. den Begriff und definieren eine kontinuierliche Skala von „Tangibility“, die sich am Grad der „Coherence“ (Kohärenz) zwischen realer und digitaler Welt bemisst. Hohe Kohärenz sagt dabei aus, dass das physische Objekt und sein digitales Gegenstück als ein „Ding“ wahrgenommen werden, also nicht voneinander abgrenzbar sind.

Entlang diesem Kontinuum führen die Autoren Kategorien ein, die die typischen Eigenschaften eines Systems im betreffenden Bereich der Skala beschreiben. Diese Kategorien dienen als Einordnungsschema für Elemente von Tangible Interfaces und können als Grundlage einer Gegenüberstellung unterschiedlicher TUIs bzw. deren Komponenten dienen. Entlang der Kohärenz-Skala legen die Autoren folgende Kategorien fest, die sich jeweils auf die Bedeutung des physischen Objekts für die Benutzungsschnittstelle beziehen (beginnend mit niedriger Kohärenz):

General-purpose tool Ein Werkzeug, das zur Manipulation verschiedener digitaler Objekte benutzt werden kann und dabei unterschiedliche Operationen auf diesen Objekten auslösen kann.

Specialized tool Ein Werkzeug, das eine bestimmte Operation auslöst, diese aber auf unterschiedliche digitale Objekte anwenden kann.

Identifier Objekte, die als Referenz auf ein digitales Objekt agieren. Die Referenz ist nicht zwangsweise permanent sondern kann unter Umständen dynamisch verändert werden.

Proxy Objekte, die insofern enger an die digitale Information gekoppelt sind als Identifier, als dass durch sie die digitale Information manipuliert und nicht nur abgerufen werden kann.

Projection Objekte, die so eng an die digitale Repräsentation gebunden sind, dass dessen physische Eigenschaft Information direkt repräsentieren und die Existenz der Information von der Existenz des Objekts abhängig ist.

Illusion of same objects Keine Kopplung im engeren Sinne sondern Identität zwischen realem Objekt und digitaler Information. Ist dann gegeben, wenn beide Komponenten ausschließlich gemeinsam auftreten oder für Benutzer nahtlos von der digitalen in die reale Welt und umgekehrt übergehen.

In der Folge detaillieren die Autoren diese Kategorien und identifizieren Eigenschaften, die eine Verknüpfung zwischen realer und digitaler Welt aufweisen kann und an denen sich der Grad an Kohärenz bemisst bzw. an denen er sichtbar wird:

Transformation Beschreibt, wie Manipulationen am realen Objekt in die virtuelle Welt umgesetzt werden. *Literal Mediation* liegt vor, wenn Manipulationen in realer und virtueller Welt analog umgesetzt werden (wenn z.B. eine Bewegung eines Objekts auch in eine Bewegung dessen realer Repräsentation umgesetzt wird). *Transformed Mediation* liegt vor, wenn eine Manipulation eines realen Objekts eine nicht unmittelbar assozierbare Reaktion in der digitalen Welt auslöst (bei der Rotation eines Tokens etwa die Lautstärke eines Audiokanals verändert).

Sensing of Interaction Beschreibt auf welche Parameter eines Objektes der realen Welt die digitale Repräsentation reagiert. Die möglichen Ausprägungen

reichen von einer einfachen Reaktion auf die Präsenz eines Objekts bis zur kontinuierlichen Reaktion auf alle sechs Freiheitsgrade des physischen Objekts.

Configurability of Transformations Gibt an, ob die Transformation, die zwischen physischem Objekt und digitaler Repräsentation angewandt wird, vorgegeben oder konfigurierbar ist. Mögliche Ausprägungen sind *konfigurierbar* und *fixiert*.

Lifetime of Link Gibt an, ob die Assoziation zwischen physischem Objekt und digitaler Repräsentation nach der Kopplung permanent ist oder zur Laufzeit verändert werden kann. Mögliche Ausprägungen sind *temporär* und *permanent*.

Autonomy Beschreibt, ob eine Existenzbeziehung zwischen dem realen Objekt und der digitalen Repräsentation besteht, ob also die digitale Ressource unabhängig vom realen Objekt existiert oder bei der Kopplung erzeugt wird. Mögliche Ausprägungen sind *autonom* und *abhängig*.

Cardinality of Link Gibt an, ob die Zuordnung zwischen realem Objekt und digitaler Repräsentation eindeutig ist oder ein mehrdeutige Zuordnung von physischer zu digitaler Welt oder umgekehrt möglich ist. Die vorrangig auftretende Ausprägung ist eine eindeutige Abbildung (*1:1*), aber auch *1:n*- oder *n:1*-Abbildungen sind möglich (wobei *n* unbeschränkt oder beschränkt sein kann).

Link Source Gibt an, ob der Gegenstand der Manipulation das physische Objekt oder die digitale Repräsentation ist. Die digitale Repräsentation ist dann die Quelle der Kopplung, wenn Änderungen an ihr Auswirkungen auf das physische Objekt haben.

Das hier vorgestellte Framework stellt erstmals die Art der Verknüpfung zwischen realer und digitaler Welt in das Zentrum der Betrachtung und klassifiziert Tangible User Interfaces entlang dieser Dimension. Die Idee der Berücksichtigung dieses Aspektes bei der Einordnung von TUIs wird später von Fishkin (2004) (siehe Abschnitt 5.2.11) wieder aufgegriffen und – vereinfacht – in seine mehrdimensionale Taxonomie für Tangible User Interfaces integriert.

5. Grundlagen der Implementierung

Kategorien	—
Konzepte	General-purpose tool, specialized Tool, Identifier, Proxy, Projection, Illusion of same objects
Eigenschaften	<i>PD-Brücke</i> : Transformation, Sensing of Interaction, Configurability of Transformation, Lifetime of Line, Autonomy, Cardinality of Link
PD-Brücke	Zentraler Aspekt: Coherence - Maß für die Enge der Bindung zwischen physischem Objekt und digitaler Information (kann für die einzelnen Teile eines Systems unterschiedlich sein)

5.2.9. Tokens und Constraints nach Shaer et al.

Shaer et al. (2004) haben in ihrer Arbeit den Anspruch, einen Satz von Konstrukten zu identifizieren, der eine umfassende Beschreibung der Struktur und Funktionalität von Tangible User Interfaces ermöglicht. Letztendliches Ziel ist es, eine konzeptionelle Basis zu schaffen, die die Entwicklung eines Software-Toolkits zur Spezifikation, Simulation und Implementierung von TUIs ermöglicht. Shaer et al. (2004) bauen dabei auf dem „Token & Constraints“-Ansatz von Ullmer (2002) auf und detaillieren das *Constraint*-Konzept, so dass es eine umfassendere Beschreibung eines TUIs erlaubt.

Die grundlegenden Konzepte des Ansatzes orientieren sich an der Annahme, dass die Struktur und Funktion eines TUI an den Beziehungen zwischen physischen Objekten und digitaler Information festgemacht werden kann. Diese Konzepte sind im Einzelnen:

Pyfo Ein physisches Objekt, das als Teil eines TUI eingesetzt wird

Token Ein Pyfo, das digitale Information oder eine Funktion zur Veränderung von Information repräsentiert.

Constraint Ein Pyfo, das das Verhalten des Tokens, dem es zugeordnet ist, einschränkt. Die physischen Eigenschaften des Constraints weisen auf die Art der Manipulation des Tokens und die Interpretation der Kombination zwischen Token und Constraint hin. Constraints können auf drei Arten einschränkend wirken:

- Die physischen Eigenschaften des Constraints (Form, Material, Orientierung, ...) weisen auf die möglichen und nicht erwünschten Manipulationen des zugehörigen Tokens hin
- Das Constraint schränkt den physischen Interaktionsraum des Tokens ein

- Das Constraint wirkt als Referenzrahmen für das Token und erlaubt dessen Interpretation

Variable Digitale Information oder eine Funktion zur Veränderung von Information. Können für sich existieren oder an ein Pyfo gekoppelt sein und dadurch ein Token erzeugen

TAC Ein TAC⁸ repräsentiert die Beziehung zwischen einem Token, dessen zugeordneter Variable und einem oder mehreren Constraints. TACs legen fest, wie Benutzer mit dem System interagieren können

Basierend auf diesen Konzepten werden fünf Kerneigenschaften, die ein Tangible User Interface bzw. dessen Elemente aufweisen können. Diese Kerneigenschaften sind:

Couple Ein Pyfo muss an eine Variable gekoppelt sein, um als Token zu gelten

Relative Definition Jedes Pyfo muss ein Token, ein Constraint oder beides sein

Association Ein TAC bildet sich aus der physischen Zuordnung eines Tokens zu einem Constraint. Dem TAC können weitere Constraints zugeordnet werden

Computational Interpretation Physische Manipulation eines Tokens hat eine eindeutig interpretierbare Auswirkung auf die digitale Welt

Manipulation Jedes TAC kann in der physischen Welt diskret, kontinuierlich oder auf beide Arten manipuliert werden. Das Constraint legt die möglichen Arten der Manipulation fest.

Zur Spezifikation eines Tangible User Interfaces werden nun diese Konzepte zur Anwendung gebracht um sowohl Struktur als auch Verhalten des TUI festzulegen. Bei der Spezifikation werden die TACs definiert, indem auf Seite der Struktur das betroffene Token und die zugehörigen Constraints angeführt werden. Zur Spezifikation des Verhaltens wird die betroffene Variable, die Aktion, die in der physischen Welt ausgeführt wird und die zu erwartende Reaktion des Systems angeführt.

Kategorien	—
Konzepte	Pyfo, Token, Constraint, Variable, TAC
Eigenschaften	Couple, Relative Definition, Association, Computational Interpretation, Manipulation
PD-Brücke	Variables (Elemente der digitalen Welt) machen ein Pyfo (Objekt der realen Welt) zu einem Token

5.2.10. Kategorien von TUI-Anwendungen

⁸Token and Constraint

5. Grundlagen der Implementierung

Auf Basis einer umfassenden Literaturrecherche im Bereich von interaktiven Systemen, bei denen physische Objekte zur Interaktion und Informationseingabe benutzt werden, identifizieren Klemmer et al. (2004) vier Kategorien von Ansätzen, die bei der Umsetzung eines TUI verfolgt werden können.

Spatial Applications Applikationen, bei denen die Interaktion auf einer beliebigen Oberfläche (Wände, Tische, Whiteboard,...) abgewickelt wird und Information über diese Oberfläche dargestellt und manipuliert werden kann. Dazu gehören auch Systeme, die ohne physische Elemente (außer der Oberfläche) arbeiten.

Topological Applications Applikationen, bei denen die Kontrolle des Systems über die Herstellung von Beziehungen zwischen physischen Objekten durchgeführt wird.

Associative Applications Applikationen, bei denen physische Objekte als Referenz auf digitale Information fungieren und diese durch Interaktion mit einer Hintergrund-Infrastruktur abgerufen werden kann.

Forms Applikationen, bei denen papierbasierte Interaktionen zu einem bestimmten Zeitpunkt (nicht kontinuierlich) in die digitale Welt übernommen werden (z.B. durch Scannen und OCR⁹-Verarbeitung).

Die Autoren gehen nicht weiter ins Detail und verzichten auf eine Betrachtung der unterschiedlichen Eigenschaften des Systems. Sie geben lediglich an, dass die meisten der betrachteten Applikationen Gemeinsamkeiten über die Grenzen der Applikationskategorien hinweg aufweisen. Dies umfasst die Tatsache, dass das vorherrschende Systemdesign die Verbindung zwischen physikalischer (tangibler) Eingabe und graphischer Ausgabe ist. Die graphische Ausgabe erfolgt dabei entweder kohärent mit der Eingabe (auf der gleichen Oberfläche), auf einem separaten Bildschirm, der sich aber nahe dem Eingabemedium befindet oder entfernt auf von den eingebenden Benutzern nicht unmittelbar zugreifbaren Ausgabemedien. Interfaces, die die Ausgabe auf andere Arten (z.B. haptisch) gestalten, werden von den Autoren explizit vernachlässigt, da sie für die eigene Forschung keine Relevanz haben.

Kategorien	Spatial, Topological, Associative, Forms
Konzepte	—
Eigenschaften	—
PD-Brücke	—

5.2.11. Taxonomie für Tangible User Interfaces

⁹Optical Character Recognition

Fishkin (2004) versucht in seiner Arbeit, den Begriff des Tangible User Interfaces zu definieren und ein Kategorienschema zu schaffen, in das sich auf tangibler Interaktion basierende Systeme einordnen lassen. Sein Ziel ist es, ein Framework zur Verfügung zu stellen, auf Basis dessen sich Systeme vergleichen lassen und das das Design von Tangible Interfaces unterstützen kann.

Fishkin fasst den Begriff des Tangible Interfaces sehr breit und definiert ein Interaktives System mit tangiblem Interface als eines, in dem die Eingabe über die Manipulation (im wörtlichen Sinn, also mit den Händen) von physischen Objekten vorgenommen wird und die Ausgabe die physische Natur eines Objektes verändert. Diese Definition umfasst auch Systeme mit „herkömmlichen“ Interfaces.

Nach dieser umfassenden Definition strukturiert Fishkin den Raum möglicher Tangible Interfaces durch die Einführung zweier Analysedimensionen, anhand derer er seine Taxonomie aufspannt. Diese beiden Dimensionen sind „Embodiment“ und „Metaphor“, die orthogonal zueinander stehen. Hohe Werte dieser beiden Dimensionen bezeichnen „tangiblere“ Systeme. „Hohe Tangibilität“ ist jedoch kein Qualitätskriterium sondern lediglich eine Eigenschaft, die ein System für einen bestimmten Anwendungsfall besser oder schlechter geeignet machen kann.

Embodiment

Die Dimension „Embodiment“ beschreibt, wie eng die Eingabe am Interface mit der Ausgabe gekoppelt ist. Das Kriterium zu Einordnung ist hier der Ort der Wahrnehmbarkeit des Systemzustandes und der Systemaktivität. Je kohärenter die Ausgabe- und Eingabe-Kanäle sind – je näher sich also die Informationsausgabe bei der Eingabe befindet – desto höher ist die Ausprägung dieser Dimension. Fishkin unterscheidet hier vier Ausprägungen:

Full Bei „full Embodiment“ ist das Ausgabegerät gleichzeitig das Eingabegerät. Der Zustand des Geräts ist direkt in seinen physischen Eigenschaften abgebildet.

Nearby „nearby Embodiment“ tritt auf, wenn die Ausgabe nahe dem Eingabeobjekt auftritt und eng an dieses gebunden ist, also in direktem, unmittelbaren Zusammenhang steht.

Environmental „environmental Embodiment“ ist gegeben, wenn die Ausgabe im unmittelbaren Umfeld des auftritt aber sich nicht unmittelbar am tangiblen Eingabeobjekt manifestiert. Typische Vertreter dieser Ausprägung sind akustische Ausgabekanäle.

Distant Von „distant Embodiment“ spricht man, wenn sich Ein- und Ausgabekanäle vollständig räumlich entkoppelt sind, der Fokus der Aufmerksamkeit der Benutzer also nicht gleichzeitig auf Ein- und Ausgabekanal liegen kann.

Metaphor

Die Dimension „Metaphor“ bildet die Eigenschaft von Tangible Interfaces ab, auf eine Benutzerinteraktion so zu reagieren, wie die reale Welt auf eine entsprechende Aktion reagieren würde. Die Ausprägung in „Metaphor“ ist also dann hoch, wenn das System analog zu realem, physikalisch begründbarem Verhalten reagiert. Hier sind grundsätzlich zwei Kategorien zu unterscheiden, in denen der Bezugspunkt der „Metaphor“ verschieden ist. „Metaphor“ kann sich entweder auf das Aussehen des jeweiligen Objektes beziehen oder auf die Bewegung des Objektes Bezug nehmen. Im ersten Fall spricht Fishkin von „Metaphor of Noun“, im zweiten Fall von „Metaphor of Verb“. Die Ausprägungen auf der „Metaphor“-Dimension gruppieren sich dann wie Folgt:

None Die Interface-Objekte zeigen weder in Form noch Funktion eine Analogie zur Realität

Noun Diese Analogie ist gegeben, wenn am Interface Objekte existieren, die eine reale Entsprechung haben, aber nicht wie diese manipuliert werden können. Ein klassisches Beispiel aus traditionellen interaktiven Systemen ist die „Fenster“- oder „Schreibtisch“-Metapher (sind analog zu realen Fenstern bzw. Schreibtischen ausgelegt, bieten aber andere Interaktionsmöglichkeiten). Bei Tangible User Interfaces ist diese Zuordnung dann gegeben, wenn ein Eingabeobjekt so aussieht wie ein Objekt der realen Welt, aber keine weiteren Eigenschaften mit diesem teilt.

Verb Eine Zuordnung zu dieser Kategorie erfolgt, wenn die Interaktion mit einem Objekt eine reale Entsprechung hat, dieses jedoch selbst keine Analogie zur realen Welt bildet. Diese Ausprägung tritt bei TUIs unter anderem bei Gestensteuerung von Systemen auf.

Noun and Verb Hier hat das betreffende Objekt selbst eine Entsprechung in der realen Welt und auch dessen Verwendung ist analog zu jener der realen Entsprechung. Die Objekte sind dennoch nach wie vor unterschiedlich, das reale Objekt kann nicht im Tangible Interface eingesetzt werden, umgekehrt bietet das TUI-Objekt nicht die reale Funktionalität des realen Objektes.

Full In der höchsten Ausprägung existiert kein Unterschied zwischen TUI-Objekt und realem Objekt - es gibt keine Analogie mehr, weil die Objekte identisch sind. Dieser Zustand ist erreicht, wenn Benutzer das TUI-Objekt manipulieren und sich die reale Welt entsprechend verändert. Beispiele für Systeme auf dieser Stufe sind zum Beispiel digital augmentierte Whiteboards, wo mit elektronischen Markern auf eine Oberfläche „geschrieben“ wird, wobei die hinterlassene „Tinte“ simultan projiziert wird.

Anwendung der Taxonomie

Fishkin wendet seine Taxonomie auf die oben bereits beschriebenen Ansätze von (Holmquist et al., 1999) und (Underkoffler und Ishii, 1999) an und zeigt, dass sich diese einordnen lassen. Er ordnet nach einer umfassenden Literaturstudie außerdem über 60 konkrete Tangible Interfaces in seine Taxonomie ein und stellt diese anhand deren Ausprägungen gegenüber. Ein offener Punkt ist die Verbindung zum MCRpd-Framework (Ullmer und Ishii, 2000), dass Fishkin als komplementär bezeichnet und das auf einer anderen Abstraktionsstufe operiere.

Kategorien	—
Konzepte	—
Eigenschaften	<i>Embodiment</i> : full, nearby, environmental, distant, <i>Metaphor</i> : none, noun, verb, noun and verb, full
PD-Brücke	wird in Struktur und Verhalten durch die Ausprägungen auf den beiden Dimensionen der Taxonomie charakterisiert (kann für die einzelnen Teile eines Systems unterschiedlich sein)

5.2.12. Tangible Bits: Beyond Pixels

Ishii (2008) fasst die bisherige Entwicklung des Forschungsgebiets der Tangible User Interfaces zusammen und schlägt konzeptionell die Brücke von den wegweisenden *Tangible Bits*-Arbeiten über das MCRpd-Interaktions-Modell bis hin zu heute aktuellen Kategorien von Tangible User Interfaces.

Die Grundlage der Betrachtungen ist das in (Ullmer und Ishii, 2000) MCRpd-Interaktions-Modell, das hier (und bereits in (Ullmer et al., 2005)) als MCRit¹⁰-Modell bezeichnet wird. (wobei „it“ für „intangible“ und „tangible“ steht und „pd“ für „physical“ und „digital“ aus der ursprünglichen Abkürzung ersetzt). Basierend auf der Trennung zwischen Input und Output eines interaktiven Systems (oder *Control* und *Representation*) und der auf dieser Trennung aufbauenden Abgrenzung zur GUIs, wird die Struktur einer TUI so konzeptualisiert, das eben diese Trennung nicht mehr vorhanden ist. Die grundlegenden Komponenten des MCRit-Modells sind (angeflehnt an das MCRpd-Modell) die *digitale Information*, die repräsentiert und manipuliert werden muss, sowie die *tangible Repräsentation* des Systemzustandes (siehe Abbildung 5.4). An die tangible Repräsentation sind die Eingabekanäle des Systems – die *Control*-Elemente – gekoppelt. Die tangible Repräsentation wird durch die in-

¹⁰Model-Control-Representation (intangible and tangible)

5. Grundlagen der Implementierung

tangible Repräsentation ergänzt, mittels der zusätzliche, dynamisch veränderliche Information abhängig vom Systemzustand ausgegeben werden kann.

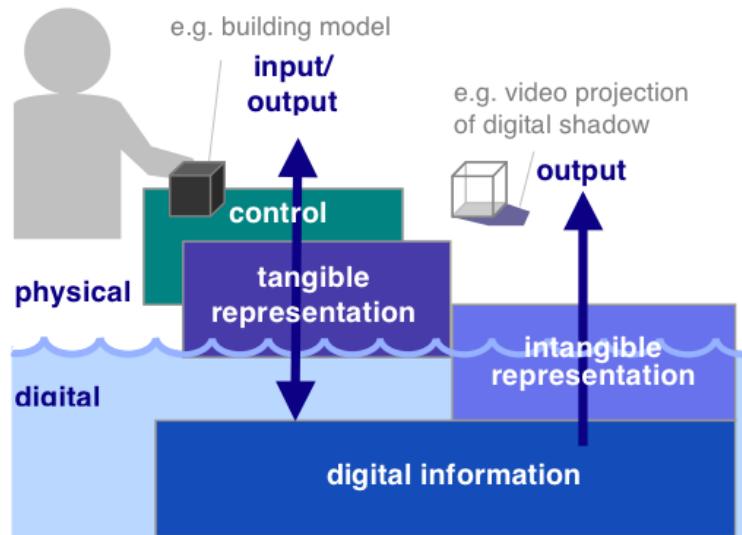


Abbildung 5.4.: Überblick über das MCRit-Modell (entnommen aus Ishii (2008))

Basierend auf der konzeptionellen Grundlage gibt Ishii die grundlegenden Eigenschaften von TUIs an:

Computational Coupling of tangible representations to underlying digital information and computation. Der zentrale Aspekt jedes TUI ist die Kopplung von realen Artefakten mit digitaler Information und diese Information manipulierende Funktionen. Zu berücksichtigen ist dabei die Art der Abbildung, also welche Prinzipien der Übersetzung und welche Metaphern (bei Ishii: „Embodiment“) verwendet werden.

Embodiment of mechanisms for interactive control with tangible representations. In TUIs werden die physischen Repräsentationen des Systemzustandes gleichzeitig auch zur Steuerung des Systems verwendet. Von Interesse sind hier die Fähigkeiten des Tangibles selbst (“inert”, also passive Objekt, die durch Benutzer manipuliert werden müssen oder „actuated“, also aktive Elemente, die selbstständig Änderungen des Systemzustandes ausgeben können), die Beschränkung des Interaktionsraums (“unconstrained” – unbeschränkt, „weakly constrained“ – nur schwach eingeschränkt, wie bei der Interaktion auf Oberflächen, oder „tightly constrained“ – stark eingeschränkt, z.B. bei Tokens die nur entlang einer Achse bewegt werden können) sowie die durch die physische Form der Tokens und die Constraints vorgegebenen bzw. ange-

deuteten Interaktionen und deren Kohärenz mit der dadurch manipulierten digitalen Information.

Perceptual Coupling *of tangible representations to dynamic intangible representations.* Die wahrgenommene Kopplung zwischen dem physischen Anteil des Systemzustandes und dem intangiblen (z.B. projizierten oder akustischen) Anteil ist der dritte wichtige Aspekt, der beim Design von Tangible Interfaces zu berücksichtigen ist. Der kritische Faktor ist dabei nach Ishii die Reaktion des intangiblen Anteil auf Änderung des tangiblen Anteils in Echtzeit. Zwischen der Manipulation von physischen Elementen und einer adäquaten Reaktion des Systems, die sich auch auf die intangible Repräsentation auswirkt, darf nur minimale, idealerweise nicht als solche wahrgenommene Verzögerung liegen.

Je nach Ausprägung dieser Eigenschaften können sich TUIs in unterschiedliche Kategorien von Systemen manifestieren. Basierend auf den bereits in (Ullmer et al., 2005) angegebenen Kategorien (siehe Abschnitt 5.2.7) erweitert Ishii das Schema und identifiziert acht Kategorien von TUIs:

Tangible Telepresence Systeme, bei denen entfernte physische Objekte digital miteinander gekoppelt werden, so dass Manipulationen eines Objekts an den gekoppelten Objekten physisches Feedback auslösen können. Durch die Kopplung zwischen entfernter Ein- und Ausgabe können Präsenz-Aspekte auch in dissozierten Anwendungsfällen übermittelt werden.

Tangibles with kinetic Memory Systeme, die Manipulation physischer Objekte aufnehmen und diese Manipulationen auch wieder physisch wiedergeben können.

Constructive Assembly Entspricht der in (Ullmer et al., 2005) angegebenen Kategorie und umfasst Systeme, bei denen der Systemzustand durch die (physische) Verknüpfung von physischen Elementen repräsentiert und manipuliert wird.

Tokens and Constraints Entspricht der in (Ullmer et al., 2005) angegebenen Kategorie und umfasst Systeme, die informationstragende physische Elemente zur Repräsentation des Systemzustandes benutzen und die möglichen Interaktionen mittels ebenfalls physischer Constraints beschränken bzw. vorgeben.

Interactive Surfaces – Tabletop UI Entspricht der in (Ullmer et al., 2005) angegebenen Kategorie und umfasst Systeme, bei denen physische Objekte auf einer Oberfläche manipuliert werden, um den Systemzustand zu verändern. Dabei wird meist zusätzlich visuelles Feedback auf der Oberfläche ausgegeben, wodurch Eingabe- und Ausgabemedium kohärent sind.

Continuous Plastic TUI Umfasst Systeme, die in der Lage sind, die äußere Form ihrer physischen Repräsentationen zu verändern bzw. auf derartige Formänderungen mit Änderungen des Systemzustandes reagieren.

Augmented Everyday Objects Systeme, bei denen physische Objekte des Alltagslebens mit Technologie ausgestattet werden, die einen wie auch immer gearbeiteten Mehrwert bei der Benutzung bieten kann.

Ambient Media Systeme, bei denen interaktive Systeme durch Zusatzinformation ergänzt werden, die nicht von der eigentlichen Arbeitsaufgabe ablenken. Im engeren Sinne handelt es sich dabei nach Ishii nicht um ein TUI, die Kategorie hat aber insofern Relevanz, als dass nicht die herkömmlichen Ausgabekanäle (wie Bildschirme) benutzt werden, um digitale Information zu repräsentieren.

Grundlegend gemein sind jedoch allen Arten von Systemen ein Satz von Merkmalen, die sie in ihrer Eigenschaft als TUIs mitbringen und die sich dem Autor zufolge durchwegs vorteilhaft auf die Interaktion mit den Benutzern auswirken können:

Double Interaction Loop – Immediate Tactile Feedback Tangible User Interfaces verfügen inhärent über zwei Feedbackschleifen über die den Benutzern Reaktionen auf deren Eingaben rückgespiegelt werden. Die erste, unmittelbar, Feedback-Schleife ist die haptische Erfahrung bei der Manipulation der physischen Element. Die Reaktion ist sofort sichtbar und muss nicht durch das System erfasst, interpretiert und ausgegeben werden. Die zweite Schleife wird über die intangible Repräsentation des Systemzustands realisiert, auf der sich ebenfalls Reaktionen auf Benutzereingaben manifestieren. Dabei muss die Benutzerinteraktion jedoch erfasst und interpretiert werden, bevor eine adäquate Reaktion erstellt und ausgegeben werden kann. Diese Reaktion kann umfassender als in der ersten Schleife ausfallen, benötigt jedoch mehr Zeit.

Persistency of Tangibles Bei TUIs wird ein wesentlicher Anteil des Systemzustandes durch die Konfiguration des physischen Elemente dargestellt. Diese sind ihrer Natur nach persistent, repräsentieren diesen Anteil des Systemzustandes also unabhängig von etwaiger Infrastruktur und sind auch verfügbar, wenn diese abgeschaltet ist.

Coincidence of Input and Output Spaces Ein grundlegendes Designkriterium von TUIs ist die Kohärenz (bei Ishii: Koinzidenz) von Eingabe- und Ausgabekanälen. Dies ermöglicht eine nahtlose Interaktion und Informationsrepräsentation im physischen und digitalen Raum.

Special Purpose vs. General Purpose In Abgrenzung zu GUIs, die im Normalfall zur Interaktion mit beliebigen Applikationen verwendet werden können, sind TUIs eher spezifisch auf einen bestimmten Anwendungsfall hin aus-

gerichtet und werden dementsprechend konzipiert. Wichtig ist hier vor allem die Berücksichtigung den Benutzern vertrauter Metaphern bei der Konzeption der Informationsrepräsentation und der Manipulations-Werkzeuge.

Space-Multiplexed Input Generell ermöglichen TUIs eine parallele Manipulation mehrerer räumlich verteilter physischer Objekte zur gleichen Zeit. Es ist damit möglich, kollaborative Interaktion zu unterstützen, bei der mehrere Benutzer den Systemzustand simultan beeinflussen. Die Kollaboration kann dabei auch räumlich verteilt stattfinden, wenn Mechanismen existieren, die den Systemzustand der einzelnen TUI-Instanzen synchron hält (z.B. durch Aktuatoren).

Im seinen Schlussbetrachtungen führt Ishii die größten Mängel des noch jungen Forschungsgebiets der Tangible User Interfaces aus: es fehle an „Killer Applications“, an skalierbaren Toolkits und an verlässlichen und validierten Design Prinzipien.

Kategorien	Tangible Telepresence, Tangibles with kinetic Memory, Constructive Assembly, Tokens and Constraints, Interactive Surfaces – Tabletop UI, Continuous Plastic TUI, Augmented Everyday Objects, Ambient Media
Konzepte	Digital Information, Tangible Representation, Control, Intangible Representation
Eigenschaften	<i>Gesamtsystem</i> : Computational Coupling, Embodiment of Control Mechanisms, Perceptual Coupling
PD-Brücke	Abhängig von der Art des Systems, grundsätzlich aber immer Kopplung zwischen tangibler Repräsentation und Control

5.2.13. Zusammenfassung

Die in den vorhergehenden Abschnitten betrachteten Arbeiten sind in Ansatz, Ausgangspunkt und Vorgehensweise höchst unterschiedlich. Ihnen ist jedoch gemein, dass sie sich mit der Konzeptualisierung von Tangible User Interfaces beschäftigen. Die Arbeiten können dabei entlang zweier Dimensionen nach Ziel und Betrachtungsgegenstand klassifiziert werden. Hinsichtlich der Zielsetzung sind Arbeiten, die auf die Spezifikation eines TUI ausgerichtet sind von solchen zu unterscheiden, die auf die Evaluation existierender Systeme ausgelegt wurden (im Sinne von detailliert angegebenen Merkmalen, die ein System aufweisen muss, um eine bestimmte Eigenschaft zu haben). Als dritte Ausprägung sind noch Ansätze zu identifizieren, die eine Einordnung in einen Referenzrahmen ermöglichen sollen ohne die

5. Grundlagen der Implementierung

Eigenschaften des TUI im Detail zu betrachten. Hinsichtlich des Betrachtungsgegenstandes sind unterschiedliche Detaillierungsgrade zu erkennen, wobei hier die beiden Ausprägungen „Gesamtsystem“ und „einzelne physische Elemente“ als Extremwerte zur Klassifikation herangezogen werden. Hinsichtlich des Betrachtungsgegenstandes ist noch zu unterscheiden, ob sich die Arbeit auf die Struktur des TUI konzentriert oder sich darüber hinaus auch explizit mit dessen Verwendung beschäftigt. Ansätze der zweiten Art werden in der Tabelle kursiv gesetzt. Jeder Ansatz verfolgt über diese Einordnung hinaus noch spezifische Zielsetzungen, die der Übersichtlichkeit wegen in dieser ersten Einordnung nicht angegeben werden.

Die Einordnung in die Kategorien (siehe Tabelle 5.1) erfolgt aufgrund der von den jeweiligen Autoren in den Artikeln explizit genannten Zielsetzungen oder – wenn diese nicht vorhanden oder aussagekräftig sind – aufgrund der von Autoren gewählten Schwerpunktsetzungen und unter Berücksichtigung des jeweiligen Gesamtzusammenhangs (übergeordnetes Forschungsvorhaben). Ansätze, die zu mehreren Kategorien Beiträge liefern, werden in allen betreffenden Feldern angeführt.

Tabelle 5.1.: Kategorien von konzeptionellen Arbeiten im Gebiet Tangible User Interfaces (kursiv gesetzte Arbeiten gehen auch auf das Verhalten von TUIs ein)

	Gesamtsystem	einzelne physische Elemente
Spezifikation	Bricks MCRpd-Interaktions-Modell Tangible Bits: Beyond Pixels	Containers, Tokens und Tools Tokens+Constraints <i>Tokens and Constraints nach Shaer et al.</i>
Evaluation	Graspable User Interfaces	<i>Taxonomie für Tangible User Interfaces</i>
Einordnung	Tangible Bits Tokens+Constraints Kategorien von TUI Anwendungen Tangible Bits: Beyond Pixels	Tangible Objects Meaning Degree of Coherence

In dieser Aufstellung ist zu erkennen, dass eine Großteil der Ansätze nicht auf den Interaktionsaspekt des Anwendungsfalls eingeht, für den das betrachtete TUI konzipiert ist, sondern sich auf die strukturellen Aspekte des Systems beschränkt.

Den umfassendsten Ansatz zur Spezifikation bietet die Arbeit „Tokens and Constraints“ von (Shaer et al., 2004), der aufbauend auf der Arbeit von Ullmer (2002) einen modifizierten Token+Constraints-Ansatz vorstellt. Dieser eignet sich durch

ein breiteres Verständnis des Constraint-Begriffs für die allgemeine Spezifikation der Struktur eines TUI. Zusätzlich stellt er ein Schema zur Verfügung stellt, in dem – basierend auf der Struktur – auch das Verhalten des Systems spezifiziert werden kann.

Zur detaillierten Evaluation eines TUI bietet sich die Taxonomie von Fishkin (2004) an, der in seiner Arbeit in zwei Dimensionen sowohl die Betrachtung der Struktur als auch der Verwendung der Objekte eines TUI in den spezifizierten Interaktionsabläufen erlaubt.

Ausdrucksstärke der konzeptionellen Ansätze

Grundsätzlich sind jene Ansätze, die sich mit der detaillierten Beschreibung einer TUI befassen, ausdrucksstärker als jene Ansätze, die ein System lediglich global betrachten. Dies bezieht sich jedoch in erster Linie auf die Quantität der generierten Daten, qualitativ gesehen ergänzen beide Sichtweisen einander und sind sowohl bei Spezifikation als auch Evaluation komplementär anzuwenden. Der Fokus der Betrachtung der jeweiligen Ansätze ist in Tabelle 5.1 angeführt und wird hier nicht separat unterschieden.

Einige der vorgestellten Ansätze sind inhaltlich insofern als überholt anzusehen, als dass sie heute gängige Techniken und Interaktionsparadigmen nicht adäquat abbilden können. Dies gilt sowohl in struktureller Hinsicht als auch in Bezug auf das Verhalten eines Systems. Die strukturellen Konzepte haben sich konzeptionell von „werkzeug-zentrierten“ auf „informations-zentrierte“ Sichtweise weiter entwickelt. Ältere, „werkzeug-zentrierte“ Ansätze betrachten die physischen Elemente ausschließlich als Werkzeuge zur Manipulation digitaler Information, die nach wie vor herkömmlich visuell ausgegeben wird und keine physische Manifestation besitzt. Typische Vertreter dieser Sichtweise sind (Fitzmaurice et al., 1995) und (Fitzmaurice, 1996). Ab der Arbeit von Ishii und Ullmer (1997) wird der Aspekt der physischen Repräsentation von Information berücksichtigt, womit erstmals eine umfassende Beschreibung von Systemen ermöglicht wird, die heute als TUI bezeichnet werden.

In der Folge wurden unterschiedliche Ansätze vorgestellt, die auf verschiedene Aspekte in der Beschreibung der Struktur eingehen. Ein grundlegendes Unterscheidungsmerkmal der Ansätze ist ihre Herangehensweise an die Unterscheidung zwischen physischen Objekten, die Information repräsentieren und solchen, die Information manipulieren (in wenigen Fällen werden auch Werkzeuge zur Systemsteuerung separat betrachtet). Eine Herangehensweise ist die strikte konzeptionelle Trennung zwischen physischen Objekten, die zur Informationsrepräsentation verwendet werden und sochen, die als Werkzeug zur Manipulation eingesetzt werden. Typische Vertreter sind hier (Ishii und Ullmer, 1997) und (Holmquist et al., 1999). Dem hinge-

5. Grundlagen der Implementierung

gen steht die Herangehensweise, die Bedeutung eines physischen Objekts für eine bestimmte Anwendung auf einem Kontinuum einzuordnen und Objekte damit als eher „werkzeug-artig“ oder eher „repräsentations-artig“ (oder beides integrierend) einzuordnen. Typische Vertreter für diese Herangehensweise sind (Underkoffler und Ishii, 1999), (Koleva et al., 2003) und (Fishkin, 2004). Einen dritten Weg gehen die Ansätze von (Ullmer und Ishii, 2000) und darauf aufbauend (Ishii, 2008), die physische Elemente immer als Kombination eines Repräsentations- und Kontroll-Anteils sehen und diese konzeptionell separat behandeln. Implizit in dieser Tradition stehen auch die Ansätze von (Ullmer, 2002) und (Shaer et al., 2004), die mit dem „Tokens und Constraints“-Konzept einerseits eine dritte Art von physischen Objekten – die Constraints, die den Interaktionsraum beschränken – einführen, andererseits aber bei den eigentlich zur Interaktion verwendeten physischen Elementen nicht zwischen Repräsentationen und Werkzeugen unterscheiden. Funktionalität wird vielmehr durch die Manipulation eines (informationstragenden) Tokens im Kontext von Constraints ausgelöst – Tokens haben somit einen Kontroll-Aspekt im Sinne von (Ullmer und Ishii, 2000).

Weniger häufig anzutreffen sind Arbeiten, die explizit auf die Beschreibung oder Evaluation der Interaktionsabläufe eines TUI eingehen. TODO

Nomenklatur

Hinsichtlich der Bezeichnung der Elemente eines TUI existiert keine einheitliche Nomenklatur, die konsistent über mehrere Arbeiten hinweg verwendet wird. Tabelle 5.2 gibt eine Übersicht über die für die einzelnen konzeptionellen Elemente verwendeten Begriffe.

Tabelle 5.2.: Gegenüberstellung der Nomenklatur zur Beschreibung der Elemente eines TUI

Arbeit	physisches Objekt	physisches Objekt zur Informations-repräsentation	physisches Werkzeug zur Informations-manipulation	physische Beschränkung des Interaktionsraums	digitale Objekte
Bricks	—	—	Brick	—	—
Graspable User Interfaces	—	—	—	—	—

5.2. Konzeptualisierung und Klassifikation von Tangible Interfaces

Tangible Bits	—	Phicon	Phandle (Informationsmanipulation), Instrument (Systemsteuerung)	Tray	—
Containers, Tokens und Tools	—	Container (unspezifische Form), Token (spezifische Form)	Tool	—	—
Tangible Objects Meaning	Object	Object as pure object (unspezifische Form), Object as attribute (teilspezifisch), Object as noun (spezifische Form)	Object as verb (fix gebundene Funktionalität), Object as recon- figurable tool (konfi- gurierbare Funktionali- tät)	—	—
MCRpd Inter- aktions- Modell	Represen- tation	Rep-P	Control	—	Model, Rep-D (Manifes- tation)
Tokens+- Constraints	—	Tokens	Tokens+- Constraints	Constraints	—

5. Grundlagen der Implementierung

Degree of Coherence	—	Identifier (unspezifische Form), Proxy (von hier an: spezifische Form, Enge der Bindung ansteigend), Projection, Illusion of same objects	General purpose tools (konfigurierbare Funktionalität), Specialized tools (fix gebundene Funktionalität)	—	—
TAC	Pyfo	Token	Token	Constraint	Variable
Kategorien von TUI-Anwendungen	—	—	—	—	—
Taxonomie für TUIs	—	—	—	—	—
Tangible Bits: Beyond Pixels	Representation	tangible Representation	Control	—	digital information, intangible representation (Manifestation)

Diese Arbeit folgt in der Bezeichnung der physischen Elemente dem TAC-Ansatz von (Shaer et al., 2004) und verwendet generell der Begriff des „*Tokens*“. Zur Abgrenzung wird, wo nötig, von „Modellierungstokens“ (jene Tokens, die Information repräsentieren) und „Werkzeugtokens“ (jene Tokens, die Funktionalität auslösen) unterschieden. Der digitale Aspekt eines TUI wird selten explizit benannt, der von (Shaer et al., 2004) gewählte Begriff der „Variable“ erscheint im Kontext der Repräsentation von Modellen aber als zu spezifisch bzw. unpassend vorbelegt. Deshalb wird im Allgemeinen nach (Ishii, 2008) von „*digitaler Information*“ gesprochen, wenn explizit auf die digitale Repräsentation des Modells Bezug genommen wird, wird der Begriff „*Modell-Elemente*“ verwendet. Eine weitere Ausdifferenzierung der Nomenklatur zur Bezeichnung von anwendungsspezifischen Eigenschaften des hier vorgestellten Werkzeugs erfolgt im Rahmen der folgenden Kapitel.

Die hier vorgestellten Ansätze werden nach der Beschreibung des Werkzeugs in Kapitel 9 wieder aufgegriffen und auf das hier entwickelte System angewandt. Damit werden zwei Ziele verfolgt. Einerseits soll die praktische Anwendbarkeit der Ansätze und deren Verwendbarkeit für ein konkretes, im Vergleich zu den in den Artikeln vorgestellten Beispielen komplexes und flexibles System überprüft werden. Andererseits soll versucht werden, aus der theoretisch-konzeptionellen Betrachtung des Werkzeugs Inkonsistenzen im Design zu erkennen und potentiell verbessерungswürdige Aspekte des Werkzeugs zu identifizieren. Eine Gegenüberstellung mit den Ergebnissen der praktischen Evaluierung erlaubt in der Folge auch die Überprüfung des Aussagekraft derartiger auf theoretischen Konzepten basierenden Betrachtungen bzw. Spezifikationen.

5.3. Tangible Interfaces in kooperativer Verwendung

(Hornecker, 2004)

5.4. Tabletop Interfaces

Grundlagen

5.4.1. Historische Entwicklung

Sensetable

Der Sensetable (Patten et al., 2001)

BUILD-IT

(Fjeld, 2001)

5.5. Tangible Interfaces zur Erstellung diagrammatische Modelle

(Blackwell et al., 2007)

5.5.1. Aktuelle verwandte Ansätze

- Historische Entwicklung von Tabletop Interfaces
 - Build IT (Fjeld et al., 1997)
 - Senstable (Patten et al., 2001)
 - Eva Hornecker (Hornecker, 2004)
 - ReacTable (Kaltenbrunner et al., 2006)
- Historische Entwicklung von Tangible Interfaces zur Modellbildung
 - Senstable Modeling Application
 - Designer's Outpost (Klemmer)
- Aktuelle verwandte Ansätze
 - (Tanenbaum und Antle, 2009)
 - (Do-Lenh et al., 2009)

5.6. Fazit

6. Eingabe und Interpretation

In diesem Kapitel wird jener Teil des Werkzeugs beschrieben, in dem die Interaktion der Benutzer mit dem System erfasst und interpretiert wird. Der erste Abschnitt behandelt grundlegende Möglichkeiten zur Erfassung der Benutzerinteraktion auf Tabletop Interfaces und endet mit der Identifikation einer für den Anwendungsfalls geeignet erscheinenden Technologie. Diese wird durch die Beschreibung von dafür verfügbaren Frameworks konkretisiert, was letztendlich in der Entscheidung für ein konkretes Produkt mündet.

- Einordnungsgrafik (Kontext - Vorgelagerte und nachfolgende Kapitel)

Basierend auf dieser Entscheidung wird in den darauf folgenden beiden Abschnitten auf das Design der Hard- und Softwarekomponenten eingegangen, die unmittelbar der Eingabe von Information durch Benutzer dienen. Der Ausgabeaspekt wird hier bewusst ausgeklammert und im nächsten Kapitel beschrieben. Dieses Kapitel endet mit einer Beschreibung der Interpretationsroutinen, die aus den durch das Framework gelieferten Rohdaten höherwertige, anwendungsspezifische Information extrahieren und diese den nachgeordneten Software-Modulen zur Verfügung stellen.

6.1. Möglichkeiten zur Erfassung von Benutzerinteraktion

Im Gegensatz zu Systemen mit dezidierten Eingabegeräten (wie Tastatur oder Maus) ist die Informationseingabe bei Tangible Interfaces unmittelbar an physische Tokens gebunden, die unabhängig voneinander und gegebenenfalls auch simultan manipuliert werden können. Diese Manipulation muss von einer vorhandenen Infrastruktur erfasst und interpretiert werden. Der wesentliche Unterschied zu dezidierten Eingabegeräten besteht darin, dass die Manipulation des physischen Artefakts selbst für den Benutzer bedeutungstragend ist und nicht nur dem Zweck einer Zustandsänderung im digitalen Informationsraum dient. Dies impliziert, dass der Zustand der verwendeten Tokens bzw. der aktuelle Wert der relevanten Parameter (z.B. Position, Rotation, Form, ...) erfasst werden können, ohne die Bedeutung der Tokens noch deren Manipulierbarkeit in der realen Welt zu beeinträchtigen. Je nach

6. Eingabe und Interpretation

Anwendungsfall kommen dafür mehrere unterschiedliche technologische Ansätze in Frage. Die Beurteilungskriterien die dabei zu berücksichtigen sind, umfassen die zu erhebenden Parametern und die notwendige Erfassungsrate des Zustandes der Tokens sowie die Anzahl der simultan zu erfassenden Tokens bzw. Parameter.

Im konkreten System muss – wie in Kapitel XY beschrieben – die planare Position von mehreren Tokens in Echtzeit (d.h. mehrmals pro Sekunde mit für den Benutzer nicht wahrnehmbaren Verzögerungen) erfasst werden. Neben der Position ist noch die Rotation eines Tokens als Raumparameter von Interesse. Bezuglich des Zustands eines Tokens muss erfasst werden können, ob es geöffnet oder geschlossen ist und ob es eingebettete Objekte enthält oder nicht. Mit diesen Anforderungen wird in den folgenden Unterabschnitten ein technologischer Ansatz zur Umsetzung der Interaktionserkennung ausgewählt.

6.1.1. In Frage kommende technologische Ansätze

Bei der Auswahl möglicher technologischer Ansätze zur Erfassung der Benutzerinteraktion müssen die zu erfassenden Parameter unterschiedlich behandelt werden. Konkret werden hier Ansätze zur Erfassung der Raumparameter (Position, Rotation) und Ansätze die Zustandsänderungen des Tokens erfassbar machen unterschieden.

Raumparameter

Zur Erfassung von Raumparametern von Tokens bieten sich mehrere technologische Ansätze an. In Frage kommen für das konkrete Tabletop Interface nur Technologien, die eine Erfassung dieser Parameter mit einer Genauigkeit im Zentimeter bis Millimeter-Bereich ermöglichen, da eine niedrigere Raumauflösung zu zu großen Ungenauigkeiten in der Positionsbestimmung führen würden, die einen Einsatz für das hier vorgestellte Werkzeug nicht erlauben würden. Im Folgenden werden die in Frage kommenden Technologien in ihren Grundzügen beschrieben und hinsichtlich ihrer Eignung für das konkrete System bewertet.

Optisch Optische Positionsbestimmung erfolgt mit Hilfe von Kamera-Systemen und Methoden der digitalen Bildverarbeitung. Die Kamera erfasst dabei die zu identifizierenden Tokens. Das resultierende Bild wird mit in Software umgesetzten Algorithmen ausgewertet. Dadurch können zumindest Identität und Position, zumeist aber auch weitere Raumparameter (wie Rotation) aller im Erfassungsbereich der Kamera befindlichen Tokens ermittelt werden. Neben dem optischen Erfassungsbereich bestimmen zudem die Auflösung der Kamera sowie die Größe der Tokens

die letztendlich erfassbare Fläche. Optische Systeme sind bei schlechten oder wechselnden Lichtverhältnissen generell eher fehleranfällig und nicht robust gegen Verdeckungen von Tokens (etwa durch Gliedmaßen oder andere Tokens).

Hinsichtlich des Identifikationsansatzes können zwei Arten von Systemen unterschieden werden. *Codebasierte* Systeme verwenden zur Identifikation eines Tokens einen von der Kamera erfassbaren Code (etwa einen „Barcode“), der eindeutig einem Token zugeordnet werden kann. *Featurebasierte* Systeme identifizieren ein Token aufgrund seiner äußereren Eigenschaften, zumeist über dessen Form (Schattenriss). Letztere bieten den Vorteil, dass ein Token nicht durch das Anbringen eines zusätzlichen Codes optisch verändert werden muss. Der größte Nachteil besteht in der Eigenschaft, dass nur Token mit unterschiedlichen Formen eindeutig identifiziert werden können. Die eindeutige Identifikation von mehreren Tokens einer Bauart ist bei featurebasierten Systemen nicht möglich. Codebasierte Systeme verwenden zumeist nicht herkömmliche EAN¹-Barcodes sondern robustere Systeme, bei denen eine Erkennung auch unter widrigen Beleuchtungsbedingungen oder niedrigen Bildauflösungen möglich ist und die zum Teil auch die Extraktion zusätzliche Information über weitere Raumparameter (wie Rotation, teilweise auch Parameter der dritten Dimension wie Neigung oder Entfernung) ermöglichen.

Codebasierte Systeme können hinsichtlich der Art der Codierung der Identitätsinformation wiederum in zwei Klassen unterschieden werden. Eine Gruppe von Ansätzen integriert die eigentliche Nutzinformation, also im Wesentlichen die tokenspezifische Identifikationsnummer, direkt in den Code und ermöglicht so ein direktes Auslesen der Information (z.B. bei QRCode (ISO JTC1/SC31, 2006)). Die zweite Gruppe verwendet eine indirekte Zuordnung zwischen Token-ID und Code. Bei derartigen Ansätzen muss die Identität eines Tokens in einem Zwischenschritt über eine Mapping-Tabelle abgebildet werden, im Gegenzug ist die Ausgestaltung des Codes flexibler, im Allgemeinen kann dabei eine höhere Robustheit bei der Erkennung erreicht werden (z.B. ARToolkit (Kato et al., 2000)).

Kapazitiv Kapazitive Ansätze basieren auf der Änderung der Kapazität von Leiterbahnen, die durch deren Berührung mit leitfähigem Material verursacht wird. Ursprünglich wurde die Technologie zur Umsetzung von berührungssensitiven Oberflächen entwickelt, kann jedoch auch zum Tracking von Tokens verwendet werden. Im Gegensatz zu druckempfindlichen Oberflächen (klassischen „Touchscreens“) ist keine Druckausübung zur Erkennung notwendig, es können außerdem auch mehrere Tokens (bzw. Finger) gleichzeitig erkannt werden.

Technologisch bedingt müssen bei kapazitiven Ansätzen alle zu identifizierenden Objekte die Oberfläche des Systems berühren. In dieser Oberfläche ist ein Metall-

¹European Article Number

gitter eingebettet, zwischen dessen Adern eine elektrische Kapazität gemessen werden kann. Diese Kapazität verändert sich, sobald die Adern berührt werden (wobei die Token in einen entsprechend geeigneten Material ausgeführt sein müssen). Durch die lokale Änderung der Kapazität kann die Position einer Berührung festgestellt werden. Die Genauigkeit ist dabei durch die Rasterweite des Metallgitters beschränkt. Der größte Nachteil eines kapazitiven Ansatzes ist in diesem Kontext aber, dass die Identität einer Tokens nicht direkt festgestellt werden kann (die Kapazitätsänderung ist für alle Token identisch). Zudem ist die Extraktion weiterer Raumparameter (wie Rotation) nicht bzw. nur mit zusätzlichen Aufwand möglich. Die Vorteile von kapazitiven Systemen liegen in der hohen Robustheit der Erkennung auch bei widrigen Umgebungseinflüssen (Lichtverhältnisse, Schmutz) sowie der prinzipiell beliebig großen und beliebig geformten Oberfläche, die zur Erkennung verwendet werden kann.

Kapazitive Systeme eignen sich also zur Positionsbestimmung, nicht aber zur Identifikation von Tokens. Dies macht sie für den konkreten Anwendungsfall nur in Kombination mit einer anderen Technologie geeignet.

Elektromagnetisch Die Ausstattung von Tokens mit elektromagnetisch erfassbaren Einheiten (z.B. RFID-Chips) ermöglicht ebenfalls die Erfassung von Raumparametern. Vorrangig eignet sich diese Technologie jedoch zur Identifikation von Tokens, die Positionsbestimmung kann nur mit erheblichem technischen Aufwand durchgeführt werden.

RFID-Chips (als Beispiel für einen elektromagnetischen Ansatz) sind passive Bauteile, die bei Energieversorgung durch ein elektrisches Feld aktiv werden und ihrerseits eine eindeutige Identifikationsnummer senden (im einfachsten Fall, komplexere Varianten sind möglich, werden hier aber nicht betrachtet). Historisch stammt die Technologie aus der Logistik und Warenwirtschaft und dient der Identifikation von Gütern und nicht der exakten Positionsbestimmung. Diese ist somit auch nur mittels erweiterter Infrastruktur möglich. Zum Auslesen eines RFID-Chips wird ein Lesegerät mit Antenne benötigt. Aus der Feldstärke, mit der diese Antenne die Antwort des Chips empfängt, kann auf die Entfernung des Chips von der Antenne geschlossen werden. Durch Kreuzpeilung mit mindestens zwei Antennen, deren Position bekannt ist, kann somit auf die ungefähre Position des Chips (und damit des Tokens, in das dieser eingebaut ist) geschlossen werden. Durch den Einsatz von „Antennenarrays“ (matrixförmig angeordneten Antennen) mit geringer Reichweite ist so eine verhältnismäßig exakte (Größenordnung einige cm) Positionsbestimmung möglich. Die Feststellung der Ausrichtung eines Tokens (Rotation) ist auf diesem Wege allerdings nicht möglich. Die Identifikation eines Tokens ist jedoch unabhängig von Sichtkontakt und unmittelbarer Berührung und somit äußerst robust gegen Umgebungseinflüsse.

Elektromagnetische Systeme eignen sich wegen des hohen technischen Aufwandes bei gleichzeitig beschränkter Genauigkeit nur bedingt zur Feststellung von Raumparametern. Durch die Ausrichtung auf Extraktion der Identitätsinformation ist der Ansatz jedoch gut zur Kombination mit anderen Technologien wie kapazitiven Ansätzen geeignet, die ihre Stärken in der Bestimmung der Raumparameter haben.

Akustisch Akustische Ansätze zur Positionsbestimmung basieren im Allgemeinen auf der Laufzeitmessung von Ultraschallwellen im Raum. Mit entsprechender Infrastruktur ist damit in einem begrenzten Bereich eine hochexakte Feststellung der Raumparameter in drei Dimensionen (Genauigkeit im mm-Bereich) sowie die Identifikation von Tokens möglich.

Ultraschallbasierte Techniken zur Positionsbestimmung basieren auf dem Einsatz von Bakensendern an bekannten Positionen. Diese Sender werden zumeist an der Zimmerdecke montiert und senden periodisch einen Ultraschallimpuls aus. Dieser Impuls wird von den Tokens (die in diesem Fall aktive Bauteile mit Stromversorgung sind) empfangen, die daraufhin einen sie identifizierenden Impuls zurücksenden. Aus der Laufzeit zwischen Absetzen des Sendeimpuls und Empfangen des Antwortimpulses bei verschiedenen Baken lässt sich so die Position des Tokens im Raum feststellen. Problematisch ist hierbei jedoch die durch den auf sequentieller Zeitmessung basierenden Ansatz beschränkte Anzahl von verfolgbaren Tokens, wenn Echtzeit-Ansprüche gestellt werden. Zudem ist der Ansatz nicht robust gegen (akustisch) verdeckte Tokens. Eine Anfälligkeit gegenüber anderen Störeinflüssen besteht nicht.

Für die Feststellung von Raumparametern sind ultraschall-basierte Systeme generell ausgezeichnet geeignet. Auch die Identifikation von Tokens ist prinzipiell möglich. Bei der Bewertung hinsichtlich des Einsatzes für Tabletop Interfaces ist jedoch zu bedenken, dass eine drei-dimensionale Positionierung nicht zu den allgemeinen Anforderungen zählt und nur in speziellen Anwendungsfällen sinnvoll sein kann. Zudem kann die Notwendigkeit von stromversorgten Tokens einen Nachteil bzw. ein Hindernis beim Einsatz darstellen.

Bewertung Im konkreten Anwendungsfällen ist die Feststellung der Identität sowie der planaren Position und Rotation von mehreren Tokens in hoher Genauigkeit sowie in Echtzeit gefordert. Aus oben genannten Gründen sind kapazitive und elektromagnetische Systeme im Einzeleinsatz nur bedingt geeignet. Akustische Systeme erscheinen für den Anwendungsfällen als zu aufwändig und unflexibel und stoßen außerdem bei der Anzahl der simultan zu verfolgenden Tokens an ihre Grenzen.

Die Kombination von kapazitiven und elektromagnetischen Systemen ist grundsätzlich eine Möglichkeit, die in Betracht gezogen werden könnte. Auch optische

Systeme genügen den Anforderungen und kommen damit in Frage. Der kombinierte Ansatz ist im Vergleich mit optischen Systemen als robuster gegen Störeinflüsse aus der Umgebung zu betrachten. Für optische Systeme sprechen hingegen die weitaus geringeren Aufwände für Infrastruktur und Tokens sowohl bei Anschaffung als auch bei Wartung und Betrieb. Durch die geringere Komplexität des Systems sind auch weniger potentielle Fehlerquellen vorhanden, was bei der Erstellung des Werkzeug-Prototypen hilfreich ist. Aufgrund dieser Aspekte und einer vergleichbaren zur erwartenden Erkennungsleistung wurde für die hier vorgestellten Anwendungsfalls die Entscheidung getroffen, ein optisches System zur Bestimmung der Positionsparameter sowie der Identität der Tokens einzusetzen.

Tokenzustand

Hinsichtlich des Tokenzustands sind im Kontext des hier vorgestellten Anwendungsfalls Informationen zu erheben, die den Inhalt des Tokens betreffen. Wie in Kapitel XY beschrieben sind die Modellierungs-Tokens als Container ausgeführt, die geöffnet und geschlossen werden können und in die kleinere Tokens als Träger von Zusatzinformation hineingelegt werden können. Die Auswahl eines Ansatzes, der die Identifikation des Öffnungs-Zustandes eines Tokens sowie dessen Inhalt erlaubt, ist Gegenstand dieses Abschnitts. Dazu wird grundlegend zwischen dem Einsatz von passiven Tokens und aktiven Tokens unterschieden. Passive Tokens besitzen keine zusätzliche Elektronik, die geforderten Informationen können lediglich durch die bereits vorhandene (optische) Infrastruktur festgestellt werden. Aktive Tokens werden hingegen mit zusätzlicher Elektronik zur Zustandsbestimmung ausgestattet, was allerdings eine Energieversorgung jedes Tokens bedingt.

Passive Token Bei passiven Tokens muss sichergestellt werden, dass die bereits vorhandene Infrastruktur die Zustandsänderungen eines Tokens erfassen kann. Da die vorhandene Infrastruktur auf optischen Technologien basiert, müssen sich alle Zustandsänderungen im äußeren – durch die Kamera erfassbaren – Erscheinungsbild eines Tokens wieder spiegeln.

Der Öffnungszustand eines Tokens kann durch Kameras einfach erfasst werden, wenn sich – je nach eingesetzter Technologie – durch das Öffnen der Umriss des Tokens verändert oder ein weiterer Code sichtbar wird bzw. der bestehende Code modifiziert wird. Diese Anforderung kann also durch passive Tokens erfüllt werden.

Zur Erfassung des Inhalts eines Container-Tokens sind zwei Ansätze denkbar. Einseitig kann der Inhalt eines Tokens zu einem bestimmten Zeitpunkt erfasst werden, andererseits ist auch eine Erfassung der Änderung des Tokeninhalts möglich (Erfassung des Vorgangs von Hineinlegen und Herausnehmen). Diese beiden Mög-

lichkeiten sind hinsichtlich der Umsetzbarkeit mit passiven Token unterschiedlich zu beurteilen. Eine Erfassung des aktuellen Tokeninhalts ist mit optischen Systemen nur schwer möglich. Die einzige sich bietende Möglichkeit ist die Verwendung von transparenten Teilbereichen der Außenfläche eines Tokens. Damit ist es grundsätzlich möglich, den Inhalt eines Tokens mit einer externen Kamera zu erfassen. Sowohl bei feature- als auch code-basierten Ansätzen sind jedoch Verdeckungen, Verzerrungen oder zu geringe Kameraauflösung potentiell problematisch und lassen diesen Ansatz für den praktischen Einsatz als ungeeignet erscheinen.

Die Erfassung der Änderung des Tokeninhalts lässt sich mit optischen Systemen einfach implementieren. So kann der Vorgang des Hineinlegens als auch des Herausnehmens von einer Kamera erfasst werden. Die größte Herausforderung hierbei ist die Identifikation des Tokens, das eingebettet wird. Hier kann es wiederum durch Verdeckungen zu Erkennungsschwierigkeiten führen, was in diesem Fall einen permanent fehlerhaften Modellzustand zur Folge hat, der sich im Falle wiederholter Fehlerkennungen sogar inkrementell zu größeren Abweichungen führen kann. Diesem Umstand kann lediglich durch eine explizite Aktion des Benutzers Rechnung getragen werden, der das betreffende einzubettende Token ins Sichtfeld der Kamera halten muss, bis das System Feedback über eine erfolgreiche Erkennung gibt. Diese Lösung erscheint allerdings hinsichtlich der Anforderung, die Technologie für den Benutzer vollkommen in den Hintergrund treten zu lassen, als eher suboptimal.

Aktive Token Aktive Tokens beinhalten zusätzliche Sensorik, die die Erfassung des Tokenzustands ermöglicht. Derartige Tokens benötigen allerdings eine Energieversorgung und müssen über eine Möglichkeit zur Datenübertragung verfügen, um den Tokenzustand an das System zu übermitteln. Weiters ist im Allgemeinen eine Steuereinheit notwendig, um die Sensoren zu kontrollieren, die Daten zu aggregieren und letztendlich zu übertragen.

Im konkreten Fall einer optisch arbeitenden Infrastruktur bietet sich eine (ggf. aufladbare) Batterie als Energiequelle an, um im Kamerabild Verdeckungen durch ansonsten eventuell zu verwendende Kabel zu vermeiden. Eine Stromversorgung über die Oberfläche (wie z.B. im Pin & Play System (Van Laerhoven et al., 2003) vorgestellt) scheidet hier aus, da die Blöcke dann mit Krafteinsatz auf die Oberfläche gesetzt werden müssten und nicht verschoben werden können.

Als Steuerungseinheit bietet sich neben selbst auf der Basis von Mikrocontrollern wie dem PIC oder 8051 (James, 1997) konzipierten Systemen auch Plattformen an, die explizit für den Anwendungszweck der Ansteuerung von Sensoren oder Aktuatoren und der Kommunikation mit einem Basissystem gefertigt werden. Exemplarisch kann hier die Smart-ITs-Plattform (Gellersen et al., 2004) angeführt werden,

6. Eingabe und Interpretation

die neben der flexiblen Ansteuerbarkeiten von unterschiedlichen Sensoren bereits Module zur Vernetzung untereinander und mit zentralen Diensten in der Infrastruktur anbietet.

Aus den eben angeführten Gründen erscheint zur Datenübertragung eine drahtlos arbeitende Technologie am geeignetsten. Aufgrund der geringen benötigten Reichweite und der Anforderung, möglichst energieeffizient zu arbeiten, bieten sich die Technologien „Bluetooth“ (Bluetooth SIG, 2007) und „ZigBee“ (ZigBee Alliance, 2007) an. Bluetooth erreicht höhere Übertragungsraten, ist aber in der Anzahl der gleichzeitig verwendbaren Geräte (max. 7) für den hier vorgestellten Anwendungsfall zu beschränkt. Ein ZigBee-Netz kann mit bis zu 255 Geräten gleichzeitig arbeiten und ist außerdem im Einsatz energiesparender. Für den gegebenen Anwendungsfall erschien also ZigBee als geeignete Technologie (und wurde auch bereits in (Ferscha et al., 2008) in einem ähnlichen Anwendungsfall erfolgreich eingesetzt).

Zur Feststellung des Öffnungsstatus eines Container-Tokens bieten sich bei aktiven Sensortechnologien mehrere Möglichkeiten an. Der Einsatz eines Schaltelements, das beim Öffnen den Kontakt herstellt oder unterbricht, erscheint als eine nahe liegende Lösung. Auch der Einsatz eines Drehelements am Angelpunkt des Öffnungsschaniers, dessen elektrische Eigenschaften (z.B. Widerstand oder Kapazität) mit dem Öffnungswinkel ändern, kann angedacht werden. Damit ist nicht nur eine Unterscheidung zwischen den Zuständen „offen“ oder „geschlossen“ sondern auch die Identifikation von Zwischenzuständen möglich.

Der Inhalt eines Container-Tokens kann ebenfalls mit unterschiedlichen Technologien erfasst werden. Die Zielsetzung ist hier nicht der Positionsbestimmung der eingebetteten Tokens sondern lediglich die Feststellung der Identität. Naheliegend ist hierzu der Einsatz von elektromagnetischen Ansätzen wie oben beschrieben. Durch das Anbringen von z.B. RFID-Chips an den einzubettenden Tokens sowie eines Lesegeräts im Container-Token kann die Identifikation robust durchgeführt werden. Alternativ bieten sich Systeme an, die auf Gewichtsmessung basieren. Über einen in das Containertoken eingebauten Sensor wird dabei das Gesamtgewicht der eingebetteten Tokens bestimmt. Bei entsprechender Konzeption der einzubettenden Tokens (unterschiedliche Gewichte) kann aus dem Gesamtgewicht auf die tatsächlich enthaltenen Tokens geschlossen werden. Ein Nachteil dieses Ansatzes ist die beschränkte Anzahl von erfassbaren Tokens und die notwendige exakte Fertigung jedes einzelnen Tokens, da es bei Gewichtsabweichungen zu Fehlerkennungen kommt.

Bewertung Hinsichtlich der erreichbaren Flexibilität und zu erwartenden Servicequalität wäre in diesem Abschnitt eine Entscheidung zugunsten aktiver Tokens zu treffen. Im Gesamtkontext betrachtet und unter Berücksichtigung der Entschei-

dung für optische Systeme zur Bestimmung der Positionsparameter ist diese Wahl jedoch zu relativieren. Wie oben beschrieben, erlaubt eine auf optischen Systemen beruhende Infrastruktur grundlegend die Umsetzung der geforderten Funktionalität. Gleichzeitig wird die Komplexität des Systems massiv reduziert und die Erstellung zusätzlicher Tokens vereinfacht (da keine zusätzliche Elektronik notwendig ist). Der Wegfall von Energieversorgung und Sensorlogik in den Token reduziert deren Gewicht und ermöglicht gleichzeitig mehr Platz für einzubettende Tokens.

Für den hier beschriebenen Anwendungsfalls bzw. die prototypische Umsetzung des Werkzeugs wird deshalb auf aktive Tokens verzichtet und der Einsatz von passiven Tokens bevorzugt. Der Mehraufwand in Erstellung und Wartung des Systems beim Einsatz aktiver Tokens wiegt in der Gesamtheit betrachtet die zu erwartende höhere Erkennungsqualität nicht auf.

6.1.2. In Frage kommende Frameworks

Unter Anbetracht der im vorherigen Abschnitt getroffenen grundlegender Technologieentscheidung zugunsten optischer Erkennungstechnologie mit passiven Tokens werden nun unterschiedliche Frameworks betrachtet, die die Umsetzung dieses Ansatzes erlauben. Dabei sind zwei Klassen von Frameworks zu unterscheiden:

Generische Frameworks für Tangible Interfaces ermöglichen generell die Koordination von Services, die die Kopplung von Sensoren und Aktuatoren mit Interpretations-Logik und letztendlich konkreten Applikationen erlauben. Sie gehen dabei nicht auf konkrete Sensortechnologien (wie die hier verwendeten optischen Ansätze) ein sondern versuchen eine Abstraktionsebene einzuführen, die die Applikationen von der konkreten Technologie entkoppelt und damit flexibler macht.

Frameworks für video-basierten Input für Tangible Interfaces sind spezialisierte Produkte, die konkret für die Umsetzung von optischen Ansätzen zur Eingabe von Daten bei Tangible Interfaces ausgelegt sind. Durch die Spezialisierung auf optische Identifikation liegt ihr Vorteil im geringeren technischen Aufwand bei der Einrichtung und auch während des Betriebs. Echtzeit-Anforderungen sind oft nur mit spezialisierten Frameworks zu erfüllen. Eine Kombinationsmöglichkeit zwischen Produkten der beiden Kategorien ergibt sich beim Einsatz eines spezialisierten Frameworks als Eingabe-Modul für eine generisches Framework. Durch diesen Ansatz kann die einfache Inbetriebnahme spezialisierter Frameworks mit der Flexibilität generischer Frameworks zusammengeführt werden.

Generische Frameworks

6. Eingabe und Interpretation

Generische Frameworks zur Behandlung von Input und Output bei Tangible Interfaces sind historisch nicht exakt von anderen Frameworks abzugrenzen, die im Umfeld des Ubiquitous bzw. Pervasive Computing (Weiser, 1991) entwickelt wurden. Von konkreter Technologie abstrahierende Frameworks wurden erstmals im Zusammenhang mit „Context-aware Computing“ (Schilit et al., 1994) erwähnt, um Applikationen die Möglichkeit zu bieten, Information aus der Umgebung über beliebige Sensoren zu erfassen, diese zu aggregieren und zu interpretieren. Aufbauend auf dieser Interpretation sollen Aussagen über den aktuellen Zustand der Umgebung (den „Kontext“) getroffen werden können, die die Applikationen wiederum zur Adaption benutzen können. Der Rückkanal, also die Ansteuerung von Aktuatoren, wurde erst in späteren Entwicklungen berücksichtigt. Ein Großteil der Systeme dient explizit nicht der Erstellung von marktreifen Applikationen sondern widmet sich eher der Umsetzung von „Rapid Prototyping“-Ansätzen im Bereich der Tangible Interfaces. Begründet wird dies mit der oft suboptimalen Ressourcen-Ausnutzung, die mit der Generalisierung und Flexibilisierung des Frameworks einhergeht.

Die Aufzählung der hier beschriebenen Frameworks erhebt keinen Anspruch auf Vollständigkeit. Bei der Auswahl wurde darauf geachtet, historische bzw. für den konkreten Anwendungsfall geeignete Ansätze aufzunehmen und in ihren wesentlichen Eigenschaften zu beschreiben.

Context Toolkit Das Context Toolkit (Dey et al., 2001) ist das historisch erste Framework, das versucht, die starre Verbindung zwischen Sensoren und Applikationslogik aufzubrechen. Es führt eine konfigurierbare Schicht ein, die eine schnellere, generischere Applikationsentwicklung ermöglicht und die Wiederverwendung einmal entwickelter Komponenten ermöglicht.

Konzeptuell existieren im Framework drei Arten von Komponenten: „Context Widgets“, „Context Interpreters“ und „Context Aggregators“. „Context Widgets“ implementieren die Ansteuerung beliebiger Software- und Hardwaresensoren und sind für das Sammeln von Information über die Umgebung zuständig. Sie vermitteln zwischen der physischen Umgebung und den konzeptionell höher liegenden Komponenten indem sie die unverarbeiteten Kontextdaten mittels einer geeigneten Schnittstelle kapseln und bestimmte Funktionen zur ersten Auswertung der Rohdaten ausführen. Eine Anwendung kann diese Daten verwenden, ohne dass sie Detailkenntnisse über die zugrunde liegenden Sensortechnologien haben muss. „Context Interpreter“ aggregieren die Sensordaten zu komplexeren Kontextinformationen d.h. sie konvertieren und interpretieren die Daten mehrerer „Context Widgets“ und versuchen diese zu einheitlichen Clustern zusammenzufassen. „Context Aggregators“ dient der Zusammenführung verschiedener Kontextinformationen die für bestimmte Anwendungen relevant sind. „Context Aggregators“ bilden damit die Schnittstelle zu den eigentlichen Applikationen.

Das Context Toolkit bietet nicht nur eine Softwareschnittstelle zu physischen Sensoren, es trennt auch die Akquisition und Repräsentation von der Auslieferung der Daten an kontextsensitive Applikationen.

SiLiCon Context Framework Das SiLiCon Context Framework (Beer et al., 2003) ist ein Vertreter jener Klasse von Frameworks, deren Verhalten zur Laufzeit dynamisch konfigurierbar ist. Dies bedeutet im konkreten Fall, dass Applikationen, die auf Basis des SiLiCon Context Framework erstellt wurden, ihr Verhalten und ihren Aufbau aufgrund eintretender Ereignisse verändern können. Dies betrifft sowohl das Aktivieren und Deaktivieren von Input- und Output-Kanälen also auch die Veränderung der Interpretation der eingehenden Information und die Reaktion darauf. Das Framework wurde entworfen, um Szenarien beschreiben zu können, in denen interaktive Systeme kontextsensitiv – d.h. abhängig vom aktuellen Zustand ihrer Umwelt – reagieren müssen.

Die grundlegenden Bausteine des SiLiCon Context Framework sind „Entitäten“, die Objekte der realen Welt konzeptionell abbilden. Diese „Entitäten“ besitzen „Attributes“, also Eigenschaften, mit Hilfe derer die Entität näher beschrieben wird. Über „Attributes“ kann die Wahrnehmung einer „Entität“ von deren Umwelt sowie deren Interaktionsmöglichkeiten mit derselben beschrieben werden. Mit Hilfe von ECA-Regeln wird beschrieben, auf welche Wahrnehmung der Umwelt („Event“) eine Entität unter welchen Bedingungen („Condition“, formuliert auf Basis des internen Zustands der Entität) mit welchen Aktivitäten („Action“) reagiert. Diese Regeln können zur Laufzeit dynamisch verändert und nachgeladen werden. Außerdem ist es möglich, in „Actions“ das Nachladen von Entitäten oder das Hinzufügen oder Entfernen einzelner Attribute durchzuführen (Oppl, 2004, S. 90).

Das SiLiCon Context Framework abstrahiert durch seine konzeptionelle Struktur mit dem Einsatz von „Entitäten“ und „Attributen“ nicht so stark von der realen Welt wie das zuvor beschriebene Context Toolkit – die Abbildung ist im ersten Schritt „direkter“ und muss erst im zweiten Schritt technisch konkretisiert werden, ein klassischer Softwareengineeringprozess (im Sinne von „Analyse – Design – Implementierung“) wird damit vollständiger (auch in den ersten Phasen) durch das Framework abgebildet und unterstützt.

Papiermaché Papiermaché (Klemmer et al., 2004) ist eines der ersten Rapid Prototyping Frameworks, die dezidiert zur Entwicklung von Tangible Interfaces entworfen wurden. Es fokussiert auf die Anbindung von Inputtechnologien für passive Objekte (RFID oder optische Ansätze) und stellt Applikationen eine abstrahierte Sicht auf die reale Welt zur Verfügung (in Form einer Repräsentation der erfassten Objekte als „Phobs“ – Physical Objects –, die für die jeweilige Applikation

6. Eingabe und Interpretation

relevanten Daten kapseln). Andere Eingabekanäle sind in der zur Verfügung stehenden Implementierung nicht vorgesehen – die Einordnung als generisches Framework ist aufgrund der übrigen Architektur trotzdem gerechtfertigt. Der Benachrichtigungsmechanismus erfolgt wie im Context Toolkit ereignisgesteuert – d.h. jede Änderung der erfassten Umgebung generiert ein Ereignis im Framework, das an die weiterverarbeitende Applikation weitergeleitet wird.

Im Sinne des Rapid Prototyping stellt Papiermaché als einziges hier betrachtetes Framework eine Art Entwicklungs- und Laufzeitumgebung für Tangible Interfaces zur Verfügung. Mittels dieser können Applikationsentwickler etwa Eingabekanäle auswählen und die Interpretation der Rohdaten zur Laufzeit konfigurieren. Die Entwicklungsumgebung macht die zugrundeliegende Hardware für den Entwickler dabei transparent, er muss sich also nicht um die Details der technischen Anbindung kümmern, ist dafür aber auf die bereits implementierten Hardware-Schnittstellen beschränkt. Die Interpretation der Daten beschreibt der Entwickler in Source Code, der über definierte Schnittstellen in das Framework eingebunden wird. Über ein Monitoring-Interface kann die Arbeit des Frameworks zur Laufzeit beobachtet werden, um so z.B. Fehlverhalten oder ineffiziente Konfigurationen aufzudecken.

Papiermaché versucht, mittels der Integration von vorgegebenen Eingabekanälen, die konfiguriert werden können, einen Teil der Komplexität bei der Entwicklung von Tangible Interface abzumildern. Der übrige konzeptionelle Aufbau ähnelt dem Context Toolkit – Interpretation der Daten und Verwendung in einer Applikation sind konzeptionell getrennt. Papiermaché ist in seiner Ausrichtung auf den Einsatz für Tangible Interfaces ausgelegt, die auf der Manipulation von einzelnen physischen Objekten und der Beziehungen zwischen diesen beruht. Es ist damit nicht so flexibel einsetzbar wie die anderen betrachteten Frameworks, erscheint aber für den hier vorgestellten Anwendungsfall grundsätzlich geeignet.

TUIpist Das TUIpist-Framework (Furtmüller, 2007) wurde im Zusammenhang mit der hier vorgestellten Arbeit entwickelt (Furtmüller und Oppl, 2007). TUIpist verfolgt einen ähnlich modularen Ansatz wie die anderen hier vorgestellten Frameworks, setzt jedoch zur Koordination der Module untereinander einen datenzentrierten Ansatz – auf dem LINDA-Konzept (Carriero und Gelernter, 1989) beruhende TupleSpaces – ein. Die konzeptionelle Modulstruktur ist ähnlich der Aufteilung, die bereits von Dey et al. (2001) im Context Toolkit vorgeschlagen wurde.

Die grundlegenden Module, die im Framework verwendet werden, sind „Sensoren“, „Aggratoren“ und „Anwendungen / Aktuatoren“ (siehe Abbildung 6.1). „Sensor“-Module binden externe Datenquellen an das Framework an. Sie enthalten dazu eine sensor-spezifische Komponente, die die Schnittstelle zur jeweiligen Hard- bzw.

6.1. Möglichkeiten zur Erfassung von Benutzerinteraktion

Software bildet. Über diese Schnittstelle gelieferte Daten werden in einer zweiten Komponente vorverarbeitet und soweit abstrahiert, dass die Datenrepräsentation unabhängig von der die Daten liefernden Sensorotechnologie ist (z.B. Abbildung von GPS²-Koordinaten auf logische Positionsinformation, die auch aus anderen Quellen stammen könnte). „Aggregatoren“ fassen die Information mehrerer „Sensor“-Module zusammen und interpretieren ggf. einander ergänzende oder auch widersprechende Information. Sie werden immer aktiv, wenn neue Sensordaten zur Verfügung stehen und aktualisieren dabei die Information über den Gesamtzustand der den Framework bekannten Umwelt. „Anwendungen / Aktuatoren“ bilden letztendlich die Schnittstelle zu konkreten Applikationen, die ihr Verhalten an den aktuellen Umweltzustand anpassen bzw. diesen darstellen oder Aktuatoren, die basierend auf dem aktuellen Umweltzustand Aktionen in dieser setzen. „Anwendungen / Aktuatoren“ filtern dabei wieder den von „Aggregatoren“ gelieferten Gesamtzustand der bekannten Umwelt und liefern nur die für die Applikation relevanten Daten aus. Dabei kann erneut eine Nachverarbeitung der Daten, z.B. im Sinne einer Anpassung an eine externe Schnittstelle, erfolgen.

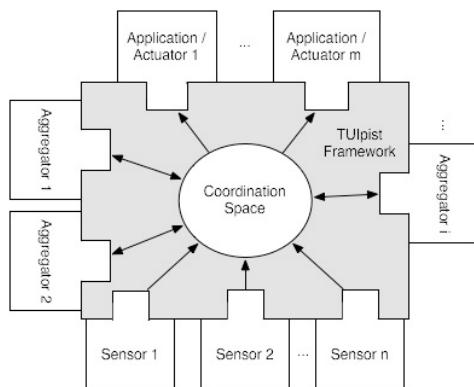


Abbildung 6.1.: Architektur des TUIpist-Framework (Furtmüller und Oppl, 2007)

Die Verbindung der Komponenten erfolgt wie erwähnt datenzentriert. Eine wesentliche Eigenschaft dieses Ansatzes ist, dass keine explizite Verknüpfungen zwischen den Modulen definiert werden. Die Zuordnung erfolgt vielmehr indirekt durch die Daten selbst. Jedes Modul kann Datensätze („Tuple“) in einem definierten Format generieren und in einen gemeinsam genutzten Datenraum – den „Tuplespace“ – stellen. Andere Module können nun Anfragen an den „Tuplespace“ stellen, ob Daten, deren Struktur oder Inhalt gewissen Kriterien entspricht, vorhanden sind. Ist dies der Fall, können diese Daten aus dem „Tuplespace“ entnommen werden und nach erfolgter Verarbeitung in modifizierter Form wieder eingestellt werden (siehe Abbildung 6.2). Das „Tupel“space-Konzept erlaubt auch eine dynamische Erweiterung

²Global Positioning System

6. Eingabe und Interpretation

bzw. Veränderung sowohl der Ein- und Ausgabekanäle als auch der internen Datenverarbeitung zur Laufzeit, indem zusätzlich Module am „Tuplespace“ registriert werden. Durch die lose Koppelung der Module muss keine zusätzliche Konfiguration an anderen Modulen oder am „Tuplespace“ selbst vorgenommen werden.

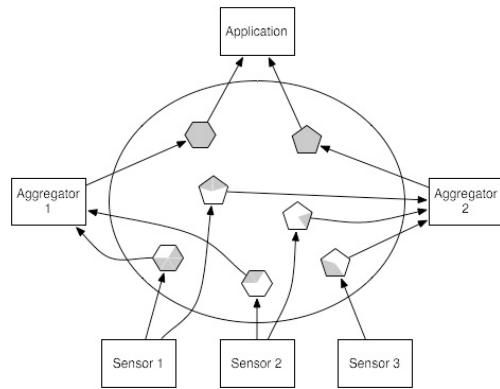


Abbildung 6.2.: Zusammenspiel der Komponenten in TUIpist (Furtmüller und Oppl, 2007)

Die Implementierung von TUIpist auf Basis des Java Jini-Frameworks (Arnold et al., 1999) ermöglicht eine Verteilung der einzelnen Module einer Applikation auf unterschiedliche Rechner (im Sinne eines „verteilten Systems“ (Stary, 1994)) ohne zusätzlich vom Implementierer zu berücksichtigenden Koordinationsaufwand. Es können damit auch (räumlich) entfernte Sensoren oder Webapplikationen angebunden werden und der ggf. auftretende Rechenaufwand zur Aggregation oder Interpretation von Daten auf mehrere Rechner verteilt werden.

Frameworks für video-basierten Input

Im Gegensatz zu den eben beschriebenen generischen Frameworks wurden die hier vorgestellten Frameworks explizit für die Behandlung von video-basiertem Input entwickelt. Wie oben bereits erwähnt, stehen die hier beschriebenen Frameworks mit den eben angeführten generischen Frameworks insofern in Zusammenhang, also dass sie zumeist als Sensor-Komponente in generischen Ansätzen eingesetzt werden können. In ihrer grundlegenden Ausrichtung sind sie jedoch zumeist für den unmittelbaren Einsatz in einer Endanwendung konzipiert.

Die hier vorgestellten Systeme implementieren den Ansatz des code-basierten optischen Trackings. Feature-basierte Ansätze kommen im konkreten Anwendungsfall nicht in Frage, da zur Modellierung eine Vielzahl von gleichartigen Objekten eingesetzt werden, die sich in ihrem äußeren Erscheinungsbild nicht unterscheiden und damit in feature-basierten Ansätzen nicht eindeutig identifiziert werden können.

Wie bereits im letzten Abschnitt erhebt auch die hier angeführte Aufzählung keinen Anspruch auf Vollständigkeit. Neben der Darstellung der historischen Entwicklung des Feldes und der Beschreibung von in Forschung und Praxis relevanten Ansätzen wurde bei der Auswahl vor allem auf freie Verwendbarkeit und Zugriff auf den Source-Code der Erkennungsroutine geachtet, da die Notwendigkeit applikationsspezifischer Anpassungen nicht auszuschließen war (etwa um benötigte aber nicht direkt unterstützte Parameter zu extrahieren).

AR Toolkit Das AR Toolkit („Augmented Reality Toolkit“) (Kato et al., 2000) ist historisch das erste in breitem Umfang eingesetzte Framework zur optischen Identifikation von Elementen in der realen Welt. Das AR Toolkit verwendet zur Identifikation Marker, in die selbst keine Information codiert ist, und eine Mapping-Tabelle zur Verknüpfung mit digitaler Information. Die Marker des AR Toolkit (siehe Abbildung 6.3) werden durch Mustererkennung identifiziert und weisen bis auf den schwarzen Rahmen, der einen Marker begrenzt, keine verpflichtenden Elemente auf. Die Form der Marker ist also vorgegeben, ihr Inhalt kann frei gestaltet werden, zur Identifikation ist es aber notwendig, dass die zu erkennenden Marker im Vorhinein beim Framework registriert werden.



Abbildung 6.3.: AR Toolkit Marker (Quelle: www.hitl.washington.edu/artoolkit/)

Das AR Toolkit ist auf die Verwendung in Anwendungen im Bereich der Augmented Reality ausgelegt und liefert die aus dem Bild extrahierte Information in einer Form, die eine Weiterverarbeitung mittels Frameworks aus dieser Domäne möglich macht. Konkret werden Raumparameter als Transformationsmatrizen ausgegeben, die eine direkte Umsetzung der Information in eine Darstellung mittels OpenGL ermöglichen. Die Anbindung von Kameras an das Framework wird über plattformspezifische Treiber realisiert, das AR Toolkit selbst empfängt einen Bildstrom, den es auswertet. Die Erkennungssoftware ist in C implementiert, bietet aber Wrapper für andere Programmiersprachen (unter anderem Java). Sie wird als Bibliothek in die eigentliche Applikation eingebunden.

Visual Codes Das Visual Codes System (Rohs, 2005) ist ein Vertreter der direkt codierenden Ansätze, d.h. dass die Nutzinformation ohne Zwischenschritt direkt

6. Eingabe und Interpretation

aus dem Code extrahiert werden kann. Im Gegensatz zu den standardisierten und kommerziell genutzten Code-Formate QR-Code (ISO JTC1/SC31, 2006) oder Datamatrix (GS1, 2008), die eine Kapazität von bis zu einigen hundert Byte haben, bietet das Visual Code System lediglich 83 Bits an Nutzinformation an (siehe Abbildung 6.4), was dazu führt, dass in vielen Anwendungsfällen ein Mapping durch die Anwendung durchgeführt wird, um die zu verwaltende Information abbilden zu können.

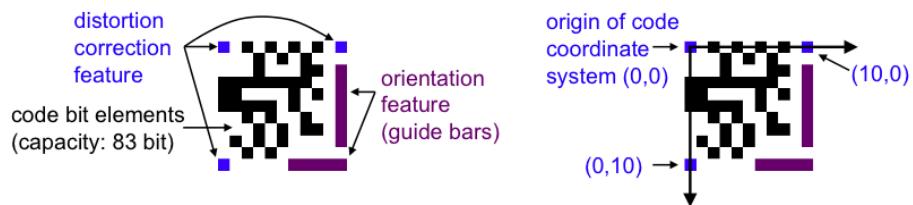


Abbildung 6.4.: Visual Codes – Aufbau und Features (übernommen von Rohs und Gfeller (2004))

Der Vorteil des Visual Code Systems liegt in der Auswertbarkeit einer Vielzahl von Raumparametern bei gleichzeitig geringem Bedarf an Rechenkapazität. In der Standardimplementierung werden neben der Position und der Rotation in der Ebene auch die Neigungsparameter im Raum extrahiert. Der Erkennungsalgorismus läuft dabei in Echtzeit und kann unverändert sogar auf Java-fähigen Mobiltelefonen ausgeführt werden. Durch die relativ geringe Datendichte der Codes reichen dementsprechend auch verhältnismäßig niedrig auflösende Bilder (z.B. 320 x 240 Bildpunkte) für eine Erkennung aus. Wie alle anderen optischen Ansätzen leidet das Visual Code System unter Erkennungsproblemen bei wechselnden Lichtverhältnissen und insbesondere bei schlechtem Kontrast. Diese Verhalten kann in der vorliegenden Implementierung auch nicht durch Anpassung der Kameraparameter (Bildverstärkung o.ä.) korrigiert werden.

Das Visual Code System muss von auf ihm aufbauenden Applikationen direkt auf Source-Code-Ebene eingebunden werden, eine vorgegebene externe Schnittstelle existiert nicht. Anbindungsrouterien für Kameras existieren nur für Mobiltelefone und müssen für andere Plattformen ggf. neu erstellt werden.

ReacTIVision ReacTIVision (Kaltenbrunner und Bencina, 2007) ist ein frei verfügbares Framework zu optischen Erkennung von Codes in Echtzeit. ReacTIVision arbeitet mit proprietären Codes (siehe Abbildung 6.5), in denen die Information nicht an bestimmte Positionen sondern im Wesentlichen in die Anzahl und Schachtelung der Schwarz-Weiß-Übergänge codiert ist. Der Umfang der direkt in einen

Marker codierbaren Information ist damit beschränkt und nicht direkt auf ein binäres Codierungsschema abbildbar. Deswegen wird ein Mapping-Ansatz verwendet, um die eigentliche Nutzinformation auf Codes abzubilden. Grundsätzlich wäre jedoch die direkte Codierung eine beschränkten Anzahl von Bits möglich, ist aber nicht unmittelbar vorgesehen.

Durch die Art der Informationscodierung ist die Erkennungsleistung von ReacTIVision auch unter schlechten Lichtbedingungen oder bei verzerrtem Eingangsbildern akzeptabel bis sehr gut. Die Form der Codes ist außerdem nicht vorgegeben, sie kann frei gewählt werden. Sogar händisch gezeichnete Codes können erkannt werden, da ausschließlich eine geschlossene Außenlinie und entsprechende Schwarz-Weiß-Übergänge innerhalb dieser Linie erfassbar sein müssen.



Abbildung 6.5.: ReacTIVision Code

Die ReacTIVision-Software ist plattformübergreifend für Windows, Linux und Mac OS X verfügbar. Sie greift über plattformspezifische Schnittstellen auf eine angeschlossene Kamera zu und wertet das empfangene Bild in Echtzeit aus. Dabei sind bei einer Kameraauflösung von 1024 x 768 Bildpunkten Bildraten von 15-20 Bildern pro Sekunde erreichbar. Diese Leistung wird auch durch eine höhere Anzahl von gleichzeitig im Bild vorhandenen Codes nicht merklich geringer.

Neben der Position der Codes wird auch deren aktuelle Rotation sowie die erste Ableitung dieser drei Parameter (also ein Maß für die Bewegung) extrahiert. Zudem können Finger, die die Oberfläche berühren, erkannt und deren Position extrahiert werden. Dies ist auch für mehrere Finger möglich, wobei eine eindeutige Zuordnung über die Zeit erhalten bleibt und so rudimentäre Multitouch-Funktionen umgesetzt werden können.

Als problematisch stellt sich wie auch bei allen anderen betrachteten optischen Ansätzen die Abhängigkeit der Erkennungsqualität von der Umgebungsbeleuchtung bzw. deren Änderung über die Zeit dar. ReacTIVision arbeitet mit adaptiven Filteralgorithmen zur Aufbereitung des Bildes, was jedoch nur leichte Beleuchtungsschwankungen ausgleichen kann. Die Software kann manuell an die Beleuchtungsverhältnisse angepasst werden (Einstellung der Blendenöffnung und der Bildverstärkung), bei sich ändernden Lichtverhältnissen muss jedoch regelmäßig eine Nachführung der Parameter vorgenommen werden.

Die aus dem Bilderstrom gewonnenen Daten werden im Falle einer Änderung zu mindest eines Wertes über eine Netzwerkschnittstelle (UDP-basiert) in einem proprietären Protokoll zur Verfügung gestellt. Zu diesem Protokoll werden Schnittstellen und Referenzimplementierungen in unterschiedlichen Programmiersprachen – unter anderem C(++) und Java – zur Verfügung gestellt. Applikationen können diese Schnittstelle implementieren und werden mittels sechs zu implementierenden Methoden an die Erkennungsroutinen angebunden (je 3 für Code- und für Fingertracking).

6.1.3. Technologieentscheidung

Auf Basis der Anforderungen, die im Anwendungsfall an das Werkzeug gestellt werden, ist nun nach der grundsätzlichen Entscheidung für ein auf optischen Ansätzen basierenden System die Entscheidung für ein konkretes Framework zu treffen.

Vergleich der Frameworks für videobasierten Input

Für die Anbindung von videobasiertem Input wurde drei Frameworks vorgestellt. Diese Frameworks sind nun hinsichtlich mehrere Aspekte zu vergleichen, die sich aus den Anforderungen der zu erstellenden Applikation sowie der Forderung nach möglichst einfacher (im Sinne von weniger aufwändig) Einbindung des Frameworks ergeben. Im Einzelnen sind dies

- die Unterstützung bei der Einbindung von Kameras und eine Schnittstelle zur Programmiersprache Java,
- eine stabile und in Echtzeit ablaufende Bilderkennung,
- eine ausreichende Anzahl von Codes (Größenordnung 100),
- die Extraktion von Position und Rotationsinformation für Codes im Erfassungsbereich der Kamera,
- die Art der Kopplung von Framework und darauf aufbauender Applikation, sowie
- die technische und lizenzierte Möglichkeit, die Frameworks an die eigenen Anforderungen anzupassen.

ReacTIVision bietet die umfassendste Unterstützung zur Einbindung unterschiedlicher Videoquellen und ist zur Anwendungsseite hin durch die Netzwerkschnittstelle am flexibelsten einsetzbar. Alle drei Ansätze bieten Schnittstellen bzw. Konnektoren für die Programmiersprache Java an. Die Frameworks selbst sind in C(++) oder Java erstellt und können auf den gängigen Plattformen (Windows, Linux, Mac

OS x) ausgeführt werden. Eine Verteilung der Applikation auf mehrere Rechner wird nur von ReacTIVision explizit unterstützt.

Hinsichtlich der eigentlichen Bilderkennungs-Routinen sind die Ansätze in ihrer Leistung und Geschwindigkeit vergleichbar, leiden aber alle unter der Abhängigkeit von der Umgebungsbeleuchtung. ReacTIVision bietet hier als einziger Ansatz die Möglichkeit, Kameraparameter zur Laufzeit nachzuführen und so Schwankungen der Umgebungshelligkeit auszugleichen.

Durch den eingesetzten Mapping-Ansatz sind AR Toolkit und ReacTIVision hinsichtlich Form und Inhalt der Codes flexibler als direkt codierende Ansätze wie Visual Codes. Dies ist im konkreten Anwendungsfall relevant, da aufgrund der Token-Form und deren beschränkter Größe eine ideale Platzausnutzung erfolgen muss, um ausreichende Erkennungsleistung zu gewährleisten.

Die Extraktion der Raumparameter (Neigungsinformation,...), die von AR Toolkit und Visual Codes ermöglicht wird, ist bei tisch-basierten Werkzeugen wie dem hier vorgestellten nicht notwendig – die Erhebung planarer Parameter (Position und Rotation) ist ausreichend. AR Toolkit liefert aufgrund seiner Ausrichtung auf Augmented Reality-Anwendung die Parameter in Form einer Transformationsmatrix, die für die Verarbeitung in 3D-Umgebungen wie OpenGL direkt geeignet ist, im hier verfolgten Ansatz jedoch einer zusätzlich Umrechnung auf das benötigte Parameterformat unterzogen werden müsste.

Die Anzahl der verfügbaren und zuverlässig unterscheidbaren Codes muss für die hier vorgeschlagene Anwendung in der Größenordnung 100 liegen. Visual Codes und ReacTIVision erfüllen diese Anforderung, AR Toolkit bietet im Lieferumfang weniger Codes an, diese können jedoch erweitert werden und bieten auch in der geforderten Anzahl noch ausreichend Unterscheidungsmerkmale für eine zuverlässige Unterscheidung (Wagner und Schmalstieg, 2003).

Das AR Toolkit und Visual Codes müssen direkt über eine Einbindung von Bibliotheken in die eigene Applikation integriert werden. ReacTIVision ist hier flexibler und wird über einen Netzwerkport mit der Zielapplikation verbunden. Dies kommt dem hier verfolgten Ansatz entgegen, da ein verteilter Betrieb aufgrund der hohen Rechenanforderungen oder im Betrieb mit mehreren Ausgabegeräten notwendig werden kann.

Von AR Toolkit und ReacTIVision steht der Source-Code zur Verfügung. Lizenzrechtlich erlauben das AR Toolkit und ReacTIVision explizit eine Veränderung des Sourcecodes unter der GNU Public Licence³ bzw. der Lesser GNU Public Licence⁴. AR Toolkit verfolgt dabei einen Dual Licensing Ansatz, der einen kommerziellen

³<http://www.gnu.org/licenses/gpl-3.0.html>

⁴<http://www.gnu.org/licenses/lGPL-3.0.html>

6. Eingabe und Interpretation

Einsatz kostenpflichtig macht. Zu Visual Codes sind keine Lizenzinformationen erhältlich, auch der Sourcecode steht nicht zum Download bereit.

Auf Basis der in Tabelle 6.1 angegeben Gegenüberstellung ist erkennbar, dass das ReacTIVision-Framework für den hier verfolgten Ansatz am besten geeignet erscheint. Betrachtet man die funktionalen Anforderungen, erscheinen zwar alle Frameworks grundsätzlich geeignet, die nicht-funktionalen Anforderungen – vor allem die plattform-übergreifende Einsetzbarkeit, die Möglichkeit zum verteilten Betrieb und die gute Unterstützung beliebiger Kameras sowie deren Konfigurierbarkeit – sprechen für die ReacTIVision-Plattform. Diese wird deshalb zur Umsetzung des Werkzeugs verwendet. Der zusätzliche Einsatz eines generischen Frameworks zur Flexibilisierung der Applikationsstruktur ist damit nach wie vor möglich und wird im nächsten Abschnitt diskutiert.

Vergleich der generischen Frameworks

Der Einsatz eines generischen Frameworks ist im konkreten Anwendungsfall für die Modularisierung der Interpretations- und Output-Module denkbar. Eine weitere Anwendung von generischen Frameworks ist die Sicherstellung der Skalierbarkeit der Anzahl der Ausgabemodule ggf. auch während des Betriebs der Applikation (um z.B. weitere Viewer-Module zur Beobachtung des Modellierungsvorganges während der Laufzeit zuschalten zu können). Die Aspekte die beim Vergleich der vorgestellten Ansätze berücksichtigt werden müssen, sind deshalb

- die Unterstützung von modularen funktionalen Einheiten sowie
- die Möglichkeit diese zur Laufzeit zu laden und entfernen und
- die Konfigurierbarkeit der Verknüpfungen dieser Module.

Daneben sind nicht-funktionale Anforderungen wie

- Skalierbarkeit,
- Effizienz bei der Weiterleitung und Verteilung der Anwendungsdaten und
- die Lizenzbestimmungen des Frameworks

zu berücksichtigen.

Die Modularisierung der Applikation wird von allen vier vorgestellten Frameworks unterstützt. Das Context Toolkit, Papiermaché und TUIpist unterscheiden dabei konzeptionell zwischen unterschiedlichen Arten von Modulen (im Wesentlichen „Input“, „Verarbeitung“ und „Output“), das SiLiCon Context Framework unterscheidet nicht zwischen unterschiedlichen Modultypen, dort wird die Rolle eines Moduls ausschließlich über seine Attribute (also die nach außen sichtbaren funktionalen Einheiten) definiert.

6.1. Möglichkeiten zur Erfassung von Benutzerinteraktion

Tabelle 6.1.: Gegenüberstellung der Frameworks für video-basierten Input

	AR Toolkit	Visual Codes	ReacTIVision
Kamera-unterstützung	nativ auf allen unterstützten Plattformen (Interfaces & Features treiber-abhängig)	nativ nur für Mobiltelefone	nativ auf allen unterstützten Plattformen
Plattformen	Windows, Linux, Mac OS X	Java-basiert auf allen Plattformen	Windows, Linux, Mac OS X
Schnittstelle zu Java	via Drittanbieter-Software	nativ	ja
Stabilität der Bilderkennung	abhängig von den verwendeten Markern	eher hoch	hoch
Kompensations-möglichkeit bei schlechten Lichtbedingun-gen	nein	nein	ja
Geschwindigkeit der Bilderkennung	> 10 fps	> 10 fps	> 10 fps
Anzahl von gleichzeitig erkennbaren Codes	keine Einschränkung	keine Einschränkung	keine Einschränkung
Erkennung von Position	implizit	ja	ja
Erkennung von Rotation	implizit	ja	ja
Kopplung von Framework und Applikation	eng	eng	lose
Source-Code verfügbar	ja	nein	ja
Lizenz	GPL, alternativ kommerziell	unbekannt	GPL, LGPL

6. Eingabe und Interpretation

Eine Erweiterbarkeit der Applikation zur Laufzeit ist nur beim SiLiCon Context Framework und bei TUIpist möglich. Das SiLiCon Context Framework arbeitet hier mit eigens erstellten Java-Classloadern, die das Nachladen von Klassen (Modulen) und deren Integration in die Infrastruktur erlauben. TUIpist verwendet die inhärent dynamische Erweiterbarkeit des Java Jini Frameworks (Arnold et al., 1999), auf Basis dessen es implementiert wurde. Das Context Toolkit und Papiermaché erlauben kein Nachladen oder Entfernen funktionaler Einheiten während der Laufzeit.

Die Konfiguration der Verbindungen zwischen den Modulen ist bei allen Frameworks möglich, konzeptionell jedoch unterschiedlich ausgeführt. Im Context Toolkit und in Papiermaché muss die Verbindung direkt im Programmcode codiert werden und wird im wesentlichen auf Methodenaufrufe abgebildet. Das SiLiCon Context Framework verwendet ereignisbasierte Regeln, um Module zu verknüpfen. Ein von einem Modul ausgelöstes Ereignis kann hier (ggf. durch Bedingungen eingeschränkt) Aktionen in anderen Modulen auslösen. Eine Besonderheit dieses Ansatzes ist, dass Teile der Interpretationslogik (also der Erkennung von Benutzerinteraktion aus den Rohdaten) in die Regeln ausgelagert werden und damit jederzeit dynamisch verändert werden können. So kann zum Beispiel die Interpretation der Nähe zwischen zwei Token in einer Regel codiert werden und so in die Reihe der zulässigen (bzw. erkennbaren) Interaktionen aufgenommen werden. TUIpist benötigt hingegen keinerlei explizite Verbindung der Module, da sämtliche Koordination über den geteilten Datenraum ausgeführt wird. Die Verknüpfungslogik wird hier in die Module verschoben, die selbst wissen müssen, für welche Daten für sie ggf. relevant sind und welche sie deshalb dem Tuplespace entnehmen.

Hinsichtlich der Skalierbarkeit der Frameworks können aufgrund mangelnder Vergleichsdaten keine fundierten Aussagen getroffen werden. Hinzuweisen ist jedoch auf die Unterstützung von verteiltem Betrieb einer Applikation durch das SiLiCon Context Framework (via SOAP-Aufrufe (Curbera et al., 2002) und das TUIpist Framework (via Jini und Java RMI⁵ (Downing, 1998)). Durch diese Verteilbarkeit ist kann die Problematik mangelnder Rechenressourcen umgangen werden, die vor allem bei rechenintensiven Anwendungen wie der eingesetzten Bildanalyse im optischen Tracking auftritt.

Die Beurteilung der Effizienz der Frameworks ist hier in Hinblick auf die geforderte Echtzeit-Interaktion mit dem System als relevant zu erachten. Das Haupt-Kriterium für Effizienz ist dementsprechend die Verzögerung zwischen Eingabe-Ereignissen und dem Feedback auf Applikationsebene, die beim Einsatz der Frameworks auftritt. Eine Beurteilung in absoluten Zahlen kann hier mangels entsprechenden Datenquellen ebenfalls nicht erfolgen, konzeptionell sind aber bei den Frame-

⁵Remote Methode Invocation

6.1. Möglichkeiten zur Erfassung von Benutzerinteraktion

works, in denen Module direkt durch Programm-Code verknüpft werden (Context Toolkit und Papiermaché), eher geringe Verzögerungen zu erwarten. Die beiden Frameworks, die mit expliziter und konfigurierbarer Middleware zur Datenvermittlung arbeiten (SiLiCon Context Framework und TUIpist) lassen eher höhere Verzögerungen erwarten, wobei der ereignisbasierte Ansatz des SiLiCon Context Framework dem daten-basierten Ansatz von TUIpist insofern überlegen ist, als das bei der datenbasierten Interaktion die Module in regelmäßigen Intervallen nach neuen Daten suchen müssen (Polling, „Information Pull“-Ansatz), während bei ereignisbasierten Ansätzen die Benachrichtigung der betroffenen Module automatisch und zum Zeitpunkt des Auftretens des Ereignisses erfolgt („Information Push“-Ansatz). Der Information Pull-Ansatz sorgt dabei einerseits für erhöhten Kommunikationsaufwand und potentiell für Verzögerungen bei Ereignissen, die innerhalb eines Polling-Intervalls auftreten.

Tabelle 6.2.: Gegenüberstellung der generischen Frameworks

	Context Toolkit	SiLiCon Context Framework	Papiermaché	TUIpist
Modularität	ja	ja	ja	ja
Erweiterbarkeit zur Laufzeit	nein	ja	nein	ja
dynamisch konfigurierbare Orchestrierung	nein	ja	nein	ja
Skalierbarkeit	schlecht	gut	schlecht	sehr gut
Effizienz bei der Weiterleitung von Daten	sehr gut	gut	sehr gut	ausreichend
Lizenz	unbekannt	unbekannt	Open-Source (nicht näher bestimmt)	Eigenentwicklung

Auf Basis der in Tabelle 6.2 angegebenen Gegenüberstellung erscheinen das SiLiCon Context Framework und TUIpist als die beiden geeignetsten Kandidaten für den Einsatz als generisches Framework im hier vorgestellten Anwendungsfall. Das Context Toolkit und Papiermaché scheiden aus verschiedenen Gründen, vor allem aufgrund der mangelnden Flexibilität aus.

Stellt man die beiden verbleibenden Frameworks gegenüber, so sind hinsichtlich der funktionalen Anforderungen keine Vorteile für einen der beiden Kandidaten zu identifizieren. Hinsichtlich der nichtfunktionalen Anforderungen scheint TUIpist hinsichtlich der Skalierbarkeit leichte Vorteile zu haben, da das Nachladen von neuen Instanzen weniger Aufwand versucht als im Fall des SiLiCon Context Framework (lediglich Laden eines Moduls im ersten Fall gegenüber Laden und Einbinden über eine Änderung der Regelbasis im zweiten Fall). Außerdem ist die technologische Basis der Verteilungsarchitektur in TUIpist konzeptionell auf höherer Ebene angesiedelt und mächtiger in der Verwaltung der Applikationsstruktur (unter anderem werden neue Module bei TUIpist automatisch in die Infrastruktur eingebunden, sobald sie angemeldet werden, im SiLiCon Context Framework muss für jeden Kommunikationskanal die IP-Adresse des Empfängers bekannt sein und in die Regelbasis aufgenommen werden).

Hinsichtlich der Effizienz der Kommunikation bietet das SiLiCon Context Framework gegenüber TUIpist aus den oben beschriebenen Gründen (Benachrichtigung über Änderungen gegenüber Polling) Vorteile. Für die hier beschriebene Applikation fällt die Entscheidung trotzdem zugunsten TUIpist. Neben der generell weniger aufwändigen Konfiguration kommt die Struktur von TUIpist einem iterativen Software-Entwicklungsprozess insofern eher entgegen, als dass die Modularisierung von initialen, monolithischen Prototypen (z.B. basierend auf der Struktur des zuvor ausgewählten ReacTIVision-Frameworks) einfacher möglich ist. Dies liegt darin begründet, dass in der modularen und dynamischen Struktur von TUIpist die gesamte Applikationslogik in den Modulen enthalten ist und so Teile aus einer monolithischen Applikation herausgelöst und direkt übernommen werden können, wobei lediglich die Logik der Datenübergabe von direkten Methoden-Aufrufen auf die TUIpist-Routinen zum Zugriff auf den Tuplespace umgearbeitet bzw. erweitert werden muss. Beim korrekten Einsatz des SiLiCon Context Framework wandert wie oben beschrieben ein Teil der Applikationslogik in die Regelbasis, was erhöhten Aufwand bei der iterativen Entwicklung verursacht und außerdem das Zusammenspiel der Applikations-Module unübersichtlicher und schwerer fassbar macht.

6.2. Konzeption und Umsetzung der Hardwarekomponenten

Basierend auf der oben getroffenen Technologieentscheidung zugunsten eines optischen, video-basierten Input-Systems für das hier zu erstellende Tabletop Interface wird in diesem Abschnitt das konkrete Hardware-Design beschrieben. Dies umfasst die Beschreibung des Tabletop Interfaces im Überblick und detaillierte Betrachtungen des Token-Designs sowie der Tisch-Oberfläche, die als Input-Kanal dient.

6.2. Konzeption und Umsetzung der Hardwarekomponenten

Weiters werden spezifische Herausforderungen und der jeweilige hier verfolgte Lösungsansatz beschrieben. Nicht Gegenstand dieses Abschnitts sind jene Hardware-Komponenten, die für die Ausgabe von Information eingesetzt werden. Obwohl hier im Überblick angeführt, werden sie detailliert erst in Kapitel XY über die Visualisierung der Modellierungsinformation beschrieben.

6.2.1. Überblick

Das Tabletop Interface ist als Tisch mit einer Oberfläche von 100 cm x 80 cm ausgeführt. Die Höhe beträgt 110 cm. Die wesentlichen Hardwarekomponenten sind die Tischoberfläche, die in semi-transparentem Acrylglass ausgeführt ist und die Bodenplatte, in die die Kamera zum optischen Tracking der Tokens auf der Oberfläche sowie der Videoprojektor zur Ausgabe von Information auf der Oberfläche (Projektion von unten) eingebaut sind (siehe Abbildung 6.6). Der Tisch ist auf allen Seitenflächen mit Platten aus expandiertem Polystyrol (Styropor) verkleidet, um im Inneren kontrollierte Umgebungslichtbedingungen für die Bilderfassung mit der Kamera zu schaffen. Die kontrollierten Bedingungen werden durch den Einsatz von vier Beleuchtungsmodulen gewährleistet, die ebenfalls in der Bodenplatte integriert sind und über den Seitenflächen eingebaute Streuscheiben für eine einheitliche, diffuse Beleuchtung der Tischinnenraums sowie der Oberfläche sorgen.

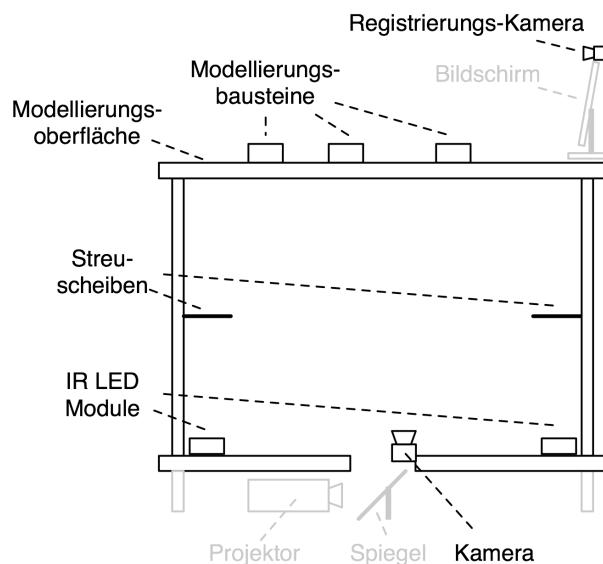


Abbildung 6.6.: Überblick über den Aufbau des Werkzeugs – Eingabekomponenten

Am Tisch befindet sich zusätzlich ein Bildschirm, auf dem entsprechend der Anforderungen zur Unterstützung der Modellierung Zusatzinformation ausgegeben

6. Eingabe und Interpretation

wird. Eine zweite Kamera, die am Bildschirm sitzt und der unterstützenden Erfassung von Information dient (zum Beispiel der bereits erwähnten Registrierung von geschachtelten Tokens).

Das gesamte System ist so zerlegbar, dass es im Kofferraum eines Mittelklassewagens transportiert werden kann. Es werden dazu die Verkleidungsplatten und Tischbeine entfernt, die Tischplatte kann dann direkt auf die Bodenplatte gesetzt und verbunden werden. Das Volumen reduziert sich dabei auf 100 cm x 80 cm x 20 cm, wobei die Tischbeine, die Verkleidungsplatten und der Videoprojektor separat transportiert werden müssen. Ein Zusammenbau des Systems inklusive Kalibrierung der Eingabe- und Ausgabe-Kanäle ist in etwa 30 Minuten möglich.

6.2.2. Tokens und Input-Werkzeuge

In diesem Abschnitt werden die einzelnen durch die Benutzer manipulierbaren Token-Arten beschrieben. Neben den eigentlichen Modellierungstokens sind dies die Tokens zur Einbettung in Container sowie die Werkzeugtokens, von denen es Ausführungen zur Manipulation des Modells und Tokens zur Auslösung bzw. Kontrolle spezifischer Funktionalitäten gibt.

Modellierungs-Tokens

Die eingesetzten Modellierungs-Tokens sind aus nicht transparentem Acrylglas gefertigt. Ihre Außenmaße betragen in etwa (je nach Form) 10 cm x 6 cm in der Grundfläche und 4 cm in der Höhe. Damit ist einerseits eine ausreichende Größe zur Anbringung der ReacTIVision-Codes gewährleistet, andererseits sind noch klein genug, um in einer Hand gehalten und manipuliert werden zu können. Die Codes werden auf der Unterseite der Tokens angebracht und auch von unten erfasst (siehe nächster Abschnitt), um eine Verdeckung der Codes während der Manipulation zu verhindern (siehe Abbildung 6.7).

Die Modellierungs-Tokens wurden in drei Ausführungen gefertigt (siehe Abbildung 6.8). Diese unterscheiden sich in Form und Farbe und können während der Modellierung von den Benutzern frei mit Bedeutung belegt werden. Die Auswahl der Formen und Farben erfolgte inspiriert von den Symbolen gängiger Modellierungsnotationen in Abstimmung mit fertigungstechnischen Einschränkungen. Eine Untersuchung geeigneter Token-Formen und eine auf diesen Ergebnissen basierende Umsetzung wurde nicht vorgenommen. Die liegt vor allem in der Tatsache begründet, dass sich der Fokus der hier vorgestellten Arbeit im Entwicklungsprozess von einem Werkzeug zur Geschäftsprozessmodellierung (mit vorgegebener Notation) hin zu einem allgemeiner einsetzbaren Werkzeug zur generischen Modellie-

6.2. Konzeption und Umsetzung der Hardwarekomponenten



Abbildung 6.7.: An Tokens angebrachte ReacTIVision-Codes zur Identifikation

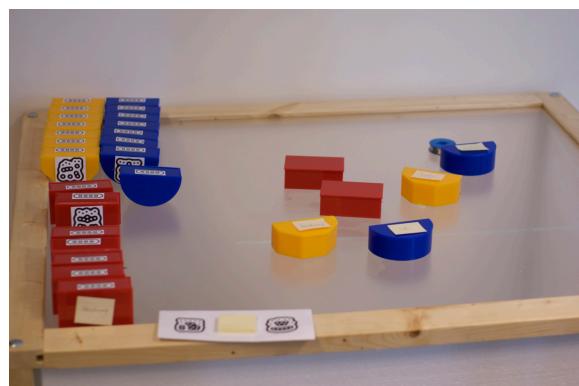


Abbildung 6.8.: Arten von Modellierungstokens

6. Eingabe und Interpretation

rung konzeptioneller Modelle (ohne vorgegebene Notation) entwickelte. Die Auswahl der Token-Formen fiel in die erste Phase, wodurch die Tokens im äußerem Erscheinungsbild an gängige Notationen zur Ablauf-Modellierung angelehnt sind. Die Token-Form scheint aber bei Benutzern ohne Modellierungs-Vorbildung geringen bis keinen Einfluss auf den Modellierungsprozess zu haben (siehe Kapitel XY – Evaluierung).

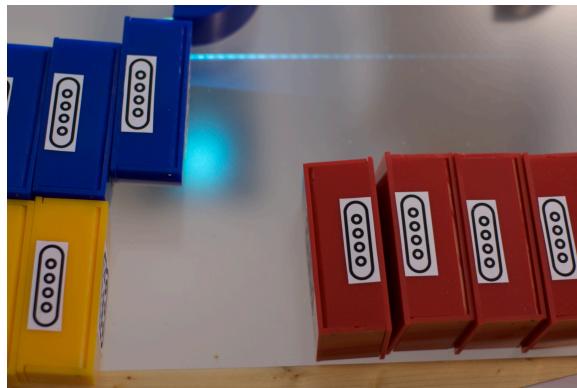


Abbildung 6.9.: Rückwand von Container Tokens

Wie in der Beschreibung der Anforderungen gefordert und oben bereits beschrieben, sind die Modellierungs-Tokens als Container ausgeführt. Im Abschnitt 6.1.1 über die Erkennung des Token-Zustands wurde beschrieben, das beim Einsatz von optischem Tracking eine Möglichkeit geschaffen werden muss, im Kamerabild zu erkennen, ob ein Token geöffnet ist oder nicht. Um den durchgängigen Einsatz des Erkennungsframeworks (ReacTIVision) zu gewährleisten, wird zu diesem Zweck ein zweiter Code eingesetzt (siehe Abbildung 6.9), der nur dann für die Kamera sichtbar wird, wenn das Token geöffnet ist. Hardwareseitig ist dies so umgesetzt, das die Modellierungstokens einen Öffnungsmechanismus besitzen, dessen Schanier nicht am Deckel sitzt (und damit nur diesen beweglich machen würde), sondern an der Bodenplatte, wodurch sich beim Öffnen eines Containers nicht nur der Deckel sondern auch die Hinterwand des Tokens bewegt (siehe Abbildung 6.10). Die Hinterwand kommt im geöffneten Zustand auf der Oberfläche des Tisches zu liegen, wodurch der auf ihr angebrachte zweite Code für die Kamera sichtbar wird. So kann durch das an sich zur Positionsbestimmung eingesetzte optische Trackingsystem zuverlässig auch den Öffnungs-Zustand der Tokens auf der Oberfläche erkennen.

Konzeptuell sind bei der Beschreibung der Verwendung dieser Tokens verschiedene Ebenen zu unterscheiden, die klar voneinander abgegrenzt werden müssen und in denen deshalb unterschiedliche Bezeichnungen verwendet werden. Diese Ebenen und die Beziehungen zwischen diesen sind in Abbildung 6.11 in Form einer einfachen Taxonomie dargestellt. Als „*Modellelement*“ wird jenes Konzept bezeichnet,



Abbildung 6.10.: Geöffnetes Container Token

das als Teil des eigentlichen Modells die darzustellenden Inhalte repräsentiert und semantisch Bedeutung trägt. Dieses Modellelement wird in Hardware durch das eben beschriebene „*Modellierungs-Token*“ beschrieben, das in Bezugnahme auf seine Eigenschaft, zusätzliche Information beinhalten zu können auch als „*Container-Token*“ bezeichnet werden kann. Das Modellierungs-Token trägt einen eindeutigen „(ReacTIVision-)Marker“, welcher wiederum einen eindeutigen „Code“ repräsentiert. Softwareseitig wird das Modellelement durch ein „*Objekt*“ (im objektorientierten Sinn, also die Instanz einer Klasse) repräsentiert, das ein Attribut enthält, in dem eine „*ID*“ abgelegt wird. Diese ID entspricht dem Code, der hardwareseitig durch den ReacTIVision-Marker repräsentiert wird und ermöglicht so eine eindeutige Zuordnung zwischen Hardware- und Softwarerepräsentation eines Modellelements.

Einbettbare Tokens

Die einbettbaren Token erlauben wie in Kapitel XY beschrieben die Verschachtelung von Modellen und das Hinzufügen von Zusatzinformation. Sie werden in einem definierten Interaktionsvorgang an Information gebunden und dann in einen Container gelegt. Hinsichtlich des Hardware-Designs sind folgende Anforderungen zu beachten:

- Die Token müssen klein genug sein, um auch mehrere Einheiten in einem Container unterzubringen.
- Sie müssen gleichzeitig groß genug sein, um einen ReacTIVision-Code zur eindeutigen Identifikation anzubringen.
- Sie müssen so ausgeführt sein, dass haptisch oder akustisch erkennbar ist, ob in einem geschlossenen Container Tokens eingebettet sind oder nicht (z.B.

6. Eingabe und Interpretation

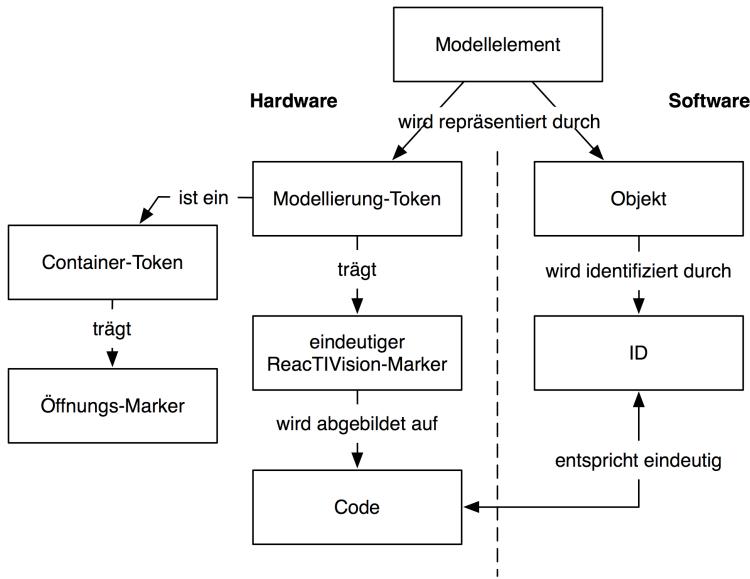


Abbildung 6.11.: Modellelemente – Taxonomie

durch Veränderung des Gewichts oder Geräusche beim Schütteln eines Containers).

Die einbettbaren Tokens wurden aus flexiblen Kunststoffmatten gefertigt und mit einem Metallstück versehen (siehe Abbildung 6.12), das einerseits als Griff dient und andererseits sowohl das Gewicht erhöht und akustisches Feedback beim Schütteln eines Container-Tokens verursacht. Die einbettbaren Tokens wurden exemplarisch in drei Ausführungen gefertigt, je nach Art und Anzahl der einzubettenden Information kann eine Definition weiterer Tokens sinnvoll sein. Folgende Tokens wurden für den aktuellen Entwicklungsstand des Werkzeugs angefertigt:

- Einbettung von Teilmodellen (inhärente Kernfunktionalität des Werkzeugs)
- Einbettung von digitalen Ressourcen bzw. Dateien am lokalen Rechner (Erweiterung zur Verknüpfung des Modells mit den realen digitalen Ressourcen aus dem Arbeitskontext)
- Einbettung von Fotos (Erweiterung zur Verknüpfung des Modells mit dem realen Arbeitskontext, z.B. mittels Fotos von Arbeitsmitteln oder Personen)

Alle Tokens sind auf der Unterseite mit einem ReacTIVision-Code versehen, der sie eindeutig identifiziert (siehe Abbildung 6.12). Aufgrund der beschränkten Größe der Tokens ist dieser Code von der Kamera, die die Tischoberfläche erfasst, nur in der Mitte der Oberfläche erfassbar (da dort die geringsten Linsen-Verzerrungen auftreten und der Code somit noch erkannt werden kann). Um etwaige Bedienungsprobleme zu vermeiden, wurde die deshalb die Interaktion mit einbettbaren Tokens

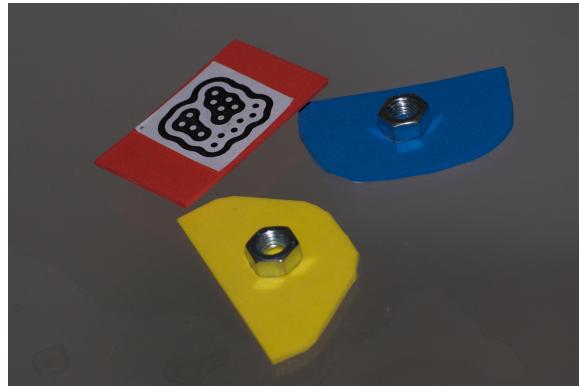


Abbildung 6.12.: Einbettbare Tokens

(Registrierung, Binden von Information) auf die in Abbildung 6.6 dargestellte zweite Kamera (“Registrierungs-Kamera”) verlegt, die durch die physische Nähe zu den Modellierenden die Codes der einbettbaren Tokens problemlos erfassen kann.

Werkzeug-Tokens

Neben den Tokens, die Modellierungsinhalte repräsentieren, wurden auch Tokens angefertigt, die der Interaktion mit dem System ansich und der Steuerung des Modellierungsablaufs dienen. Hier sind zwei Arten zu unterscheiden: einerseits gibt es Werkzeuge, die der Manipulation des Modells dienen (z.B. Herstellung von Verbindungen zwischen Tokens, Benennung von Tokens) und andererseits solche, die der Steuerung von modellierungsunterstützender Zusatzfunktionalität dienen (etwa Tokens, mit denen die Rückverfolgung der Modellierungshistorie gesteuert werden kann). Außerdem sind orthogonal zu dieser Klassifizierung noch Tokens zu unterscheiden, die beim Einsatz ein Ereignis auslösen und solche, die das System solange in einen anderen Zustand versetzen, bis sie wieder entfernt werden (gleich einem Taster). Im Folgenden werden die einzelnen Werkzeuge beschrieben und den eben definierten Kategorien zugeordnet.

Manipulation des Modells Die bei der Verwendung des Systems wichtigsten Werkzeug-Tokens sind jene, die zur Benennung und Verbindung von Modellierung-Tokens verwendet werden. Diese *Markierungs-Tokens* sind als auf die Oberfläche aufsetzbare Stäbe konzipiert, die unten eine Standfläche aufweisen, welche auch den Code aufnimmt. Am oberen Ende ist eine Verbreiterung angebracht, die eine stabile Handhabung gewährleistet (siehe Abbildung 6.13). Bedingt durch die kleine Standfläche kann nur der einfachste ReacTIVision-Code verwendet werden, der lediglich aus einem schwarzen Ring besteht. Das ReacTIVision-System erkennt diesen Ring

6. Eingabe und Interpretation

nicht als regulären Code sondern als „Cursor“, also als Zeiger, der ausschließlich eine Position in der Ebene besitzt, aber keine Rotation aufweist. Für die Verwendung zur Markierung von Modellierungs-Tokens zum Zwecke der Benennung oder Verbindung ist dies auch nicht notwendig.



Abbildung 6.13.: Markierung-Token

Das Modellierungswerkzeug muss zwischen unterschiedlichen Arten von Verbindungen unterscheiden können, da diese bei der vorgestellten Art der Externalisierung zum Einsatz kommen können. Im Wesentlichen ist zwischen ungerichteten und gerichteten Verbünden zu unterscheiden. Im Gegensatz zu ungerichteten Verbünden weisen gerichtete Verbinder Pfeilspitzen an einem Ende oder an beiden Enden auf. Für die Werkzeug-Tokens bedeutet dies, dass neben den bereits beschriebenen, einfachen Markierungs-Tokens auch solche zum Einsatz kommen, die anstatt einer runden Grundfläche eine größere, dreieckige Grundfläche aufweisen, die eine Pfeilspitze symbolisiert (siehe Abbildung 6.13). Technisch werden die beiden Arten von Markierungs-Tokens durch einen zweiten Cursor-Code unterschieden, der an Tokens mit dreieckiger Grundfläche zusätzlich angebracht ist.

Die Markierungs-Tokens sind der Kategorie der ein Ereignis auslösenden Tokens zuzuordnen. Dies bedeutet, dass das System zu jenem Zeitpunkt eine Aktion setzt, an dem das Token auf die Oberfläche aufgesetzt wird. Wird ein Markierungs-Token auf der Oberfläche belassen, verursacht es keine weiteren Aktionen bis es entfernt und erneut aufgesetzt wird.

Dem hingegen ist das nächste hier behandelte Token der Kategorie der den Systemzustand beeinflussenden Tokens zuzuordnen. Es hat also solange Auswirkungen auf den Modellierungsablauf, solange es auf der Oberfläche belassen wird. Die konkrete Funktion dieses Tokens ist der Wechsel in einen Werkzeugmodus, der ein explizites Entfernen bzw. Löschen von Bestandteilen des Modells, insbesondere Verbindungen, erlaubt. Das Token ist als Radiergummi ausgeführt und codiert so die

6.2. Konzeption und Umsetzung der Hardwarekomponenten

Metapher des expliziten Löschens von Information aus dem Modell (siehe Abbildung 6.14). An der Unterseite des Tokens ist eine ReacTIVision-Code angebracht um es gegenüber dem System eindeutig zu identifizieren.



Abbildung 6.14.: Lösch-Token

Der Einsatz der *Löschen-Tokens* verändert im System im Wesentlichen die Interpretation des Markierungs-Tokens, das nun anstatt der Herstellung einer Verbindung das Löschen derselben auslöst. Der zugehörige Interaktionsablauf ist im Detail in Abschnitt 6.3.4 dargelegt.

Ein weiteres Werkzeug im Kontext der Manipulation des Modells ist das *Registrierung-Token*, für die Benennung von Blöcken mittels handgeschriebenen Haftnotizen. Wie in Abschnitt 6.3.2 im Detail ausgeführt, gibt es zwei Möglichkeiten um ein Modellierung-Token zu benennen. Einerseits kann der herkömmlich Eingabeweg über die Tastatur gewählt werden, andererseits ist es möglich, die Modellierung-Tokens mittels aufgeklebten Haftnotizen zu benennen. Um die handgeschriebene Information in die digitale Information übernehmen zu können, wird diese mittels Bildanalyse-Verfahren aus einem Bild extrahiert, das von der oben bereits erwähnten Registrierungskamera (siehe Abbildung 6.6) aufgenommen wird. Die Aufnahme wird durch das erwähnte Registrierungstoken ausgelöst. Dieses fungiert gleichzeitig als Halter für noch unbeschriftete Haftnotizen und weist am linken und rechten Rand jeweils einen ReacTIVision-Code auf (siehe Abbildung 6.15). Diese beiden Codes erlauben es, im Bild die Position der Haftnotiz und damit die zu extrahierende Information zu identifizieren.

Das Registrierungstoken ist in die Kategorie der Ereignis auslösenden Tokens einzzuordnen. Seine Verwendung erwies sich im praktischen Einsatz als wenig intuitiv und technisch instabil bzw. zu langsam, so dass – wie erwähnt – als alternativer Eingabeweg zusätzlich eine Tastatur ins System aufgenommen wurde. Die Interaktionsabläufe sind in beiden Fällen ähnlich und werden im Detail in Abschnitt 6.3.2 dargestellt.

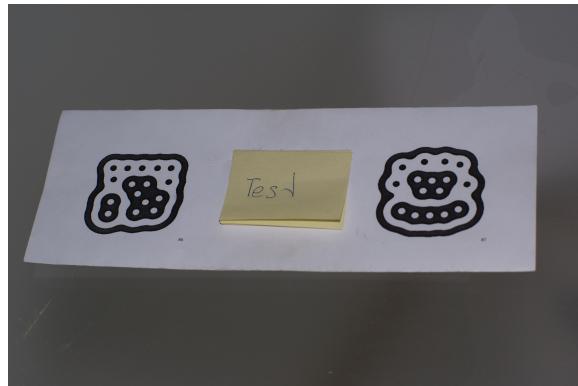


Abbildung 6.15.: Registrierungstoken

Steuerung von Unterstützungsfunktionen Im Werkzeug wurden entsprechend den Anforderungen aus Kapitel XY den Modellierungsablauf unterstützende Funktionen implementiert. Diese werden über dezidierte Werkzeug-Tokens gesteuert. Der Anwendungsbereich dieser Art von Tokens ist die Steuerung aller Funktionen, die im Kontext der Verfolgbarkeit der Modellierungs-Historie umgesetzt wurden.

Das *Snapshot-Token* dient der durch die Benutzer ausgelösten Speicherung des aktuellen Systemzustandes. Wie in Kapitel XY beschrieben, verfolgt das System den Modellierungsablauf und speichert selbsttätig stabile Systemzustände (zur Identifikation dieser Zustände siehe Abschnitt 6.3). Zusätzlich können Benutzer mit eben dem hier beschriebenen Token die explizite Aufnahme des aktuellen Modellzustandes veranlassen, wenn ihnen dieser wesentlich erscheint. Das Token ist als ereignisauslösendes Element konzipiert und wird durch kurzes Aufsetzen des Tokens auf die Modellierungsoberfläche aktiviert. Physisch ist es prototypisch als großes rotes Quadrat ausgeführt, an dessen Oberseite ein Griffstück angebracht ist (siehe Abbildung 6.16). Auf der Unterseite sitzt wiederum ein ReacTIVision-Code zur Identifikation des Tokens.

Neben der Speicherung von Modellzuständen können diese auch wieder abgerufen und in ihrer zeitlichen Sequenz dargestellt werden. Zur Aktivierung des Historien-Modus und zur Navigation in der Zeitlinie wurde ein weiteres Token eingeführt, das als einziges sowohl der zustandsverändernden als auch der ereignisauslösenden Token-Kategorie zuzuordnen ist. Zustandsverändernd wirkt das *History-Token*, insfern, als dass das Aufsetzen auf die Oberfläche das System in den Historienmodus schaltet und ein Entfernen die aktuelle Modell-Ansicht wieder herstellt. Durch Rotation des Token im bzw. gegen den Uhrzeigersinn kann nun durch die in ihrer zeitlichen Abfolge angeordneten gespeicherten Modellzustände navigiert werden. Die Rotation löst dabei jeweils nach einem bestimmten zurückgelegten Drehwinkel ein

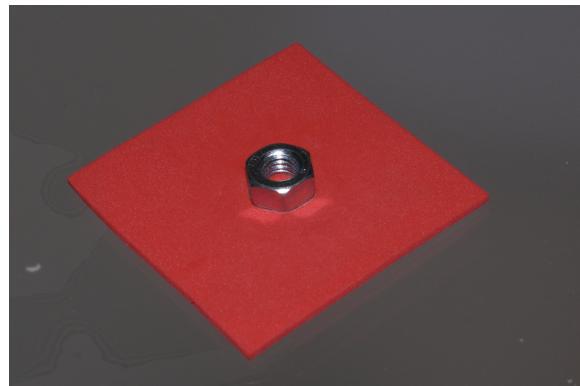


Abbildung 6.16.: Snapshot-Token

Ereignis aus, das den entsprechend vorhergehenden bzw. nachfolgenden Systemzustand abruft. Um die Rotation möglichst natürlich handhabbar zu machen, ist das Token mit runder Grundfläche etwa handflächengroß ausgeführt (siehe Abbildung 6.17). Auf der Unterseite der Grundfläche ist wiederum der ReacTIVision-Code angebracht.

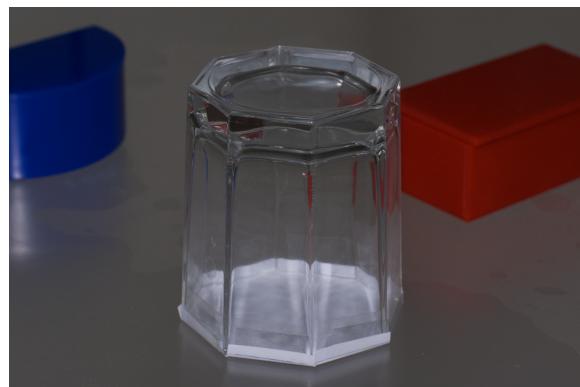


Abbildung 6.17.: History-Token

Das letzte hier behandelte Token zur Steuerung einer Unterstützungsfunction ist jenes, dass die in Kapitel XY als Anforderung definierte und in Kapitel XY im Detail beschriebene *Wiederherstellungsunterstützung* auslöst. Die Verwendung dieses Tokens ist identisch mit jener des oben beschriebenen Snapshot-Tokens. Wie dieses ist es der ereignisauslösenden Kategorie zuzuordnen und aktiviert die Wiederherstellungsunterstützung wenn es bei aktivem Historien-Modus auf der Modellierungsoberfläche aufgesetzt wird. Die eigentliche Wiederherstellungsunterstützung ist dem Output-Kanal zuzuordnen und in ihrem Ablauf deshalb im Detail in Kapitel XY (Abschnitt XY) beschrieben.

6.2.3. Eingabe auf der Tischoberfläche

Nachdem nun die Tokens beschrieben wurden, die den Benutzern unmittelbar zur Interaktion mit dem System dienen, liegt der Fokus dieses Abschnitts nun auf der technischen Umsetzung der Erfassung der aktuell auf der Oberfläche sichtbaren ReacTIVision-Codes und damit der Tokens.

Wie bereits oben beschrieben, wurde bei der Umsetzung des optischen Tracking-Ansatzes eine Identifikation der Codes von der Unterseite gewählt, um etwaigen Erkennungsproblemen mit Verdeckungen vorzubeugen. Dies bedingt, dass die Modellierungsoberfläche für Kameras möglichst transparent sein muss um den Identifikationsvorgang nicht negativ zu beeinflussen. Grundsätzlich bietet sich dazu der Einsatz einer (Acryl-)Glasplatte an, die keinen Einfluss auf die Erfassung der auf ihr platzierten Codes hat. Der Einsatz einer (Acryl-)Glasplatte bringt jedoch zwei Probleme mit sich. Einerseits kann die Verwendung einer transparenten Oberfläche dazu führen, dass Modellierende durch den Einblick auf den inneren Aufbau des Systems von der Aufgabe abgelenkt bzw. irritiert werden. Andererseits verhindert eine transparente Oberfläche den Einsatz derselben als Rückkanal zur Darstellung von zusätzlicher Information durch einen Videoprojektor. Dies ist im konkreten Fall besonders schwerwiegend, da die *a-priori* unspezifischen Modellierungs-Tokens erst durch die Projektion der zusätzlichen Information explizit bedeutungstragend werden. Deswegen wurde eine semi-transparente Acrylglass-Oberfläche eingesetzt, die einerseits für die Erfassung der Codes ausreichend durchlässig ist, andererseits aber auch das projizierte Licht soweit bricht, dass die darzustellende Information auf der Oberfläche sichtbar ist.

Die Kamera – eine AVT Guppy Fo8oB (Allied Vision Technologies GmbH, 2008) – sitzt wie in Abbildung 6.6 schematisch dargestellt in der Mitte der Bodenplatte. Sie ist mit einem Weitwinkel-Objektiv ausgestattet, das es erlaubt, die gesamte Oberfläche zu erfassen, ohne dass der Tisch höher als 100 cm sein muss (Öffnungswinkel etwa 45°). Die Kamera bietet eine Auflösung von 1024 x 768 Bildpunkten und ist über eine Firewire-Schnittstelle (IEEE 1394 (Bloks, 1996)) mit dem Rechner verbunden. Die Kamera kann so bis zu 30 Bilder pro Sekunde an den Rechner senden. Das Weitwinkelobjektiv verursacht Verzerrungen in den Randbereichen, was trotz der Möglichkeit von ReacTIVision, das Bild softwareseitig zu entzerren, dazu führt, dass links und rechts etwa 10 cm der Tischoberfläche nicht zur Identifikation von Codes verwendet werden können. Dieser Problematik kann aufgrund der beschränkten Kameraauflösung nur durch größere Codes begegnet werden, was durch die beschränkte Größe der Tokens nicht möglich ist. Im Prototypen wurden deshalb die Randbereiche des Systems ausgespart und können zur Ablage von aktuell nicht eingesetzten Tokens verwendet werden.

Illumination und Umgebungslichtabhängigkeit

Optische Tracking-Systeme leiden generell unter ihrer Abhängigkeit von den Umgebungslichtbedingungen. Probleme, die in diesem Zusammenhang auftreten, hängen nicht nur mit der absoluten Stärke der Einstrahlung von Umgebungslicht zusammen (die zu schwach, aber auch zu stark sein kann), sondern auch mit der Änderung der Lichtbedingungen über die Zeit. Gegenstand dieses Abschnitts ist die genauere Erörterung dieser Problematik und die Darlegung von Möglichkeiten ihr zu begegnen.

Zu geringe Umgebungshelligkeit führt zu Erkennungsproblemen, da der Kamera nicht ausreichen Licht und damit Kontrast zur Verfügung steht, um ein Bild zu liefern, in dem die ReacTIVision-Codes zuverlässig identifiziert werden können (dies gilt im Übrigen auch für alle anderen optischen Identifikations-Ansätze). Das Problem wird zusätzlich verschärft, da zur Aufnahme des Bildes nicht das Spektrum des sichtbaren Lichts verwendet wird, sondern in den Infrarot-Bereich (“near infrared”, Wellenlänge etwa 550 nm) ausgewichen wird. Dies ist notwendig, da die Kamera andernfalls Reflexionen des auf die Oberfläche projizierten Output-Kanals (siehe Abschnitt XY) erfasst, was eine Identifikation der Codes unmöglich macht. Die meisten Kunstlicht-Quellen strahlen jedoch nicht ausreichend Licht im verwendeten Infrarot-Bereich ab, um eine ausreichende und gleichmäßige Ausleuchtung der Oberfläche zu gewährleisten. Dieser Problematik wurde dadurch begegnet, dass im betreffenden Infrarot-Bereich strahlende Beleuchtungsquellen in den Tisch (konkret in den Ecken der Bodenplatte) eingebaut wurden (siehe Abbildung 6.6). Um eine gleichmäßige, diffuse Ausleuchtung zu erreichen, wurde der Tisch rundum mit weißen Kunststoffplatten mit strukturierter Oberfläche verkleidet, die das vorhandene Licht streuen. Um Reflexionen der Lichtquellen (die nach oben abstrahlen) auf der Oberfläche zu vermeiden, wurden etwa 50 cm über den Lichtquellen Streuscheiben angebracht. So konnte eine diffuse Beleuchtungssituation geschaffen werden, deren Stärke ausreicht, um die Codes auch bei vollkommener Dunkelheit in der Umgebung zuverlässig zu erkennen.

Das zweite Problemfeld betrifft zu starke Lichteinstrahlung aus der Umgebung. Zu hohe Umgebungshelligkeit führt zu einem Überstrahlen des Kamerabildes in manchen Bereichen, womit in Teilen der Oberfläche keine Codes erkannt werden können. Dies liegt in dem von ReacTIVision eingesetzten adaptiven Code-Extraktions-Algorithmus begründet, der bei schwachen oder guten Beleuchtungsverhältnissen leichte Unterschiede in der Ausleuchtung korrigieren kann, bei starken Unterschieden oder großer Helligkeit jedoch nicht mehr zuverlässig funktioniert. Wie bereits erwähnt, arbeitet die Kamera im Infrarot-Bereich. Dies ist dann problematisch, wenn starke Sonneneinstrahlung vorhanden ist, da diese einen hohen Anteil an Lichts der relevanten Wellenlänge enthält. Auch bestimmte Kunstlichtquellen wie Leuchstoffröhren, die unmittelbar über dem Tisch angebracht sind, verursachen Probel-

6. Eingabe und Interpretation

me. Während punktuelle Helligkeitszonen („Spotlights“) nicht durch die Software korrigiert werden können und vermieden werden müssen, kann eine generell zu hohe Umgebungshelligkeit durch die Kalibrierung der ReacTIVision-Software bzw. der ihr zugrunde liegenden Kamera-Treiber korrigiert werden. Im Konkreten ermöglicht ReacTIVision bei Firewire-Kameras eine Veränderung der Kamerablende (größere Blende – weniger Licht am Sensor) sowie der Signalverstärkung (weniger Verstärkung – dunkleres Bild). Durch Veränderung dieser beiden Parameter kann nahezu jede Beleuchtungssituation korrigiert werden, sofern sie über die Zeit stabil bleibt.

Bei Umgebungslicht-Bedingungen, die sich über die Zeit verändern (z.B. bei Tageslicht, das sich durch Wolkenbewegungen oder tageszeitabhängig verändert), kann die manuelle Konfiguration des ReacTIVision-Frameworks nicht sinnvoll durchgeführt werden. Leichte Veränderung der Helligkeit kann das ReacTIVision-Framework durch den adaptiven Erkennungs-Algorithmus noch selbst korrigieren, bei größeren Veränderungen funktioniert jedoch die Erkennung nicht mehr stabil. Dies äußert sich darin, dass Codes für einige Sekundenbruchteile von der Kamera nicht mehr erfasst werden, was wiederum zu unerwünschten von Framework gemeldeten Ereignissen führt (z.B. dass ein Container-Token verschwunden wäre, obwohl es sich physisch noch auf der Oberfläche befindet). Da diese Erkennungsausfälle im Grenzbereich gerade noch möglicher Erkennung zwar regelmäßig auftreten, in ihrer zeitlichen Ausdehnung aber eher kurz sind, wurde im Rahmen der Interpretation der vom ReacTIVision-Framework gemeldeten Ereignisse eine Stabilisierungsebene eingezoomt, die versucht, Ausfälle und Fehlerkennungen zu identifizieren und dementsprechend nicht an die übergeordneten Software-Ebenen weiterzugeben. Die in dieser Ebene umgesetzten Maßnahmen sind in Abschnitt 6.4.2 beschrieben.

Durch die Kombination aller beschriebenen Maßnahmen ist es möglich, eine sehr weitgehende Einsetzbarkeit der Werkzeugs unabhängig von den herrschenden Umgebungslichtbedingungen zu gewährleisten. Lediglich bei extremen Verhältnissen wie direkter Einstrahlung von in der Stärke variierenden Lichtquellen verhindert einen stabilen Einsatz des Systems.

6.3. Benutzerinteraktion mit dem Werkzeug

Bevor nun die Interpretation der Eingabedaten durch die Software beschrieben wird, muss vorab geklärt sein, auf welche Art und Weise Benutzer mit dem System interagieren können. Bisher wurde erklärt, wie die Eingabe technisch funktioniert und welche Elemente den Benutzern zur Interaktion mit dem System zur Verfügung stehen. Im Folgenden werden die einzelnen Aktionen, die Benutzer grundsätzlich

im Kontext des Modellierungsablaufs setzen können, identifiziert und deren konkrete Umsetzung durch das vorgestellte System beschrieben.

Ein grundsätzliches Designziel war es, die Interaktion mit dem System so „natürlich“ wie technisch möglich zu gestalten. Unter „natürlich“ ist hier zu verstehen, dass

- die Eingabe von Information möglichst ausschließlich über direkte Aktionen der Hände der Benutzer (d.h. ohne Übersetzung in die digitale Welt durch Maus oder Tastatur) gestaltet werden kann, und
- dass diese Aktionen bzw. deren Ablauf nur durch inhaltliche Vorgaben der Modellierungsproblematik, nicht aber durch technische Einschränkungen, vorgegeben sind.

Wie bereits im letzten Abschnitt beschrieben, konnten diese Forderungen aufgrund der Eigenschaften des eingesetzten Tracking-Systems nicht vollständig umgesetzt werden. Wo noch Einschränkungen hinsichtlich der oben genannten Forderungen bestehen, wird in den folgenden Abschnitt darauf hingewiesen und ein möglicher Lösungsansatz skizziert.

6.3.1. Hinzufügen und Verändern von Modellelementen

Um Elemente zu einem Modell hinzuzufügen müssen entsprechende Tokens auf der Modellierungsoberfläche platziert werden. Sofern sich diese im Erfassungsbereich der Kamera befinden, werden sie identifiziert und an der entsprechenden Stelle im Modell identifiziert. Beim erstmaligen Hinzufügen wird das Token dem Benutzern gegenüber mittels seiner Nummer (die durch den ReacTIVision-Code vorgegeben ist) dargestellt. Ist das Token bereits benannt und wird erneut hinzugefügt, wird die zuvor vergebene Benennung geladen und angezeigt.

Beim erstmaligen Hinzufügen eines Elementtyps (also eines roten, blauen oder gelben Tokens) fragt das System nach der Bedeutung dieses Elementtyps. Diese kann angegebenen werden, darf aber auch unspezifiziert bleiben und kann zu einem späteren Zeitpunkt festgelegt oder verändert werden. Diese Interaktion zu einem späteren Zeitpunkt muss explizit durch ausgelöst werden, indem ein Token des betreffenden Typs vor die Registrierungskamera gehalten wird.

Um die Position und Rotation eines Modellelements zu verändern, muss es entsprechend an einen anderen Ort verschoben oder gedreht werden. Dabei ist es nicht notwendig, dass das Token ständig von der Kamera erfasst wird. Kurze Aussetzer in der Erfassung, die durch schnelles Verschieben oder Abheben des Tokens und Aufsetzen an einem anderen Ort ausgelöst werden, werden von der Software ignoriert,

das Element wird so behandelt, als wäre es nicht verschwunden gewesen (hinsichtlich der Auswirkungen des Verschwindens eines Tokens siehe Abschnitt 6.3.4).

Eine simultane Erweiterung oder Veränderung des Modells durch mehrere Benutzer ist möglich, die Software schränkt die Anzahl der möglichen zeitgleich ablaufenden Interaktionen nicht ein. Da alle in diesem Abschnitt beschriebenen Interaktionen eindeutig einem Token zugeordnet werden können und dieses zu jedem Zeitpunkt eindeutig identifiziert werden kann, kann das System mit dieser Art von Eingaben umgehen. Die Erfassungsrate liegt in der derzeitigen Implementierung bei etwa 15 Bildern in der Sekunde, woraus sich ein Auswerteeintervall von etwa 67 Millisekunden ergibt. In diesem Abstand wird jeweils die gesamte Oberfläche erfasst und etwaige Änderungen an die Applikationsschicht weitergemeldet.

6.3.2. Benennen von Modellelementen

Die Benennung von Modellelementen ist eine der wichtigsten Interaktionsformen im Modellierungsprozess. Modellelementen wird damit ein Name zugewiesen, der zur Identifikation des Elements gegenüber den Benutzern verwendet wird. Um der Anforderung nach möglichst direkter, unmittelbarer Interaktion mit dem System in der realen Welt (d.h. auf der Modellierungsoberfläche) nachzukommen, wurde ein auf der Verwendung von Haftnotizen basierender Interaktionsmodus eingeführt, der zur Benennung von Modellelementen eingesetzt wird. Alternativ kann die Benennung durch Eingabe der Bezeichnung über eine Tastatur erfolgen.

Im Falle der Benennung mittels Haftnotizen wird ein Werkzeugtoken eingesetzt, auf dem die Haftnotizen angebracht sind. Unmittelbar nach der Platzierung eines Elements wird eine neue Haftnotiz beschriftet und mittels der Registrierungskamera erfasst. Während die Haftnotiz nun auf dem Modellelement angebracht werden kann, wird im System das erfasste Bild ausgewertet, der geschriebene Text wird (als Grafik) extrahiert und dem zuletzt auf der Oberfläche platzierten Token zugeordnet. Soll ein Element nachträglich (um)benannt werden – wenn also zwischenzeitlich ein weiteres Element platziert wurde – muss das zu benennende Element mit dem Markierungstoken ausgewählt werden. Das Markierungstoken wird dazu in der Nähe des Elements auf die Oberfläche gesetzt. Visuelles Feedback auf der Oberfläche signalisiert eine erfolgreiche Auswahl. In der Folge kann die neue Benennung mittels des Registrierungstokens erfasst und am Element angebracht werden. Die Benennung mittels Haftnotizen ist insofern problematisch, als dass die Extraktion der handschriftlichen Benennung nur unzuverlässig funktioniert. Dies liegt vor allem an zu geringem Kontrast zwischen Haftnotiz und Schrift (bei schlechten Lichtverhältnissen) und einer zu geringen Auflösung der Registrierungskamera. Außerdem werden keine Schrifterkennungs-Algorithmen eingesetzt, wodurch der Schriftzug für

das System immer lediglich eine Menge von Bildpunkten ohne weitere Bedeutung bleibt.

Um die Nachteile der Benennung mit Haftnotizen zu vermeiden, wurde ein alternativer Interaktionsmodus zur Benennung von Modellelementen eingeführt. Dieser läuft grundsätzlich identisch ab, setzt jedoch anstatt der Haftnotizen eine Tastatur ein, über die die Benutzer die Benennung dem System bekannt geben. Die eingegebene Information wird dann auf der Oberfläche unmittelbar unterhalb des Elements angezeigt (siehe Kapitel XY). Die Auswahl des zu benennenden Elements erfolgt wiederum durch das Markierungstoken. Die Vorteile dieses Systems liegen in der geringen Fehlerrate und der hohen Geschwindigkeit der Eingabe (etwa Faktor 10 schneller als die Eingabe mittels Haftnotiz). Der Nachteil dieses Ansatzes ist der exklusive Zugang zum Eingabemedium, der Tastatur. Während es beim Einsatz von Haftnotizen grundsätzlich möglich ist, diese simultan zu erstellen (auch wenn die Registrierung sequentiell erfolgen muss), kann bei der Verwendung der Tastatur zu einem Zeitpunkt immer nur ein Element bearbeitet werden.

Grundsätzlich sind die beiden Interaktionsmodi zur Benennung von Elementen als Ergänzung zueinander zu sehen. Sie können grundsätzlich gleichzeitig eingesetzt werden und bieten den Benutzern eine Wahlmöglichkeit zwischen einer langsameren, dafür aber physisch existenten oder einer schnelleren, dafür nur über die Ausgabekanäle des Systems sichtbaren Benennung. Die Erkennungsprobleme der Haftnotizen-Variante (die zur Einführung des zweiten Interaktionsmodus geführt hatten) können durch eine Verbesserung der technischen Rahmenbedingung ausgemerzt werden, wozu aber bei der Erstellung des hier vorgestellten Prototypen keine Ressourcen vorhanden waren. Im Falle einer Verbesserung könnten den Benutzern zwei gleichwertige Eingabekanäle zur Verfügung gestellt werden. Vorerst bietet der Einsatz der Tastatur jedoch nicht vernachlässigbare Vorteile hinsichtlich der Zuverlässigkeit und der Genauigkeit der Erfassung von Benennungen.

6.3.3. Verbinden von Modellelementen

Das Verbinden von Modellelementen ist neben dem Hinzufügen und Benennen der dritte wesentliche Interaktionsmodus, der zum Aufbau eines Modells benötigt wird. Verbindungen werden nicht physisch gelegt sondern mit Werkzeugtokens erstellt und dann über die Outputkanäle visuell dargestellt. Die Entscheidung gegen eine physische Repräsentation der Verbindungen fiel zugunsten einer leichteren Veränderbarkeit des gesamten Modells – das Verschieben eines Elements verursacht so keinen zusätzlichen Aufwand, wohingegen physische Verbinde nachgeführt werden müssten bzw. eine beliebige Neuanordnung verhindern würden.

6. Eingabe und Interpretation

Die Erstellung von Verbindern kann wie die Benennung auf zwei unterschiedliche Arten erfolgen, die sich aber ihrer Ausdrucksmächtigkeit bzw. Flexibilität unterscheiden. Der flexiblere Interaktionsmodus ist jener, der unter Einsatz von zwei Markierungstokens durchgeführt wird. Dabei wird jeweils ein Markierungstoken neben einem der beiden zu verbindenden Modellelemente platziert, wodurch diese ausgewählt und anschließend verbunden werden. Das Verbindungskriterium ist, dass die beiden Modellelemente innerhalb von 3 Sekunden nacheinander ausgewählt werden. Dies erlaubt sowohl eine gleichzeitige Platzierung von zwei Markierungstokens als auch die Arbeit mit nur einem Token, das rasch hintereinander neben die zu verbindenden Elemente gesetzt wird. Die Arbeit mit den herkömmlichen Markierungstokens erzeugt ungerichtete Verbindungen. Werden gerichtete Verbindungen benötigt (deren Richtung mit einer oder im Falle einer bidirektionalen Verbindung mit zwei Pfeilspitzen angezeigt wird), müssen spezielle Markierungstokens verwendet werden, deren Grundfläche die Form einer Pfeilspitze hat und die neben dem entsprechend zu markierenden Modellelemente platziert werden müssen.

Der alternative Interaktionsmodus zur Herstellung einer Verbindung kommt ohne Werkzeugtokens aus und basiert ausschließlich auf der Manipulation der Modellelemente. Werden zwei Modellelement an ihren Längsseiten nahe zusammengeführt, so erstellt das System zwischen diesen beiden Elementen eine Verbindung. Im Interaktionsablauf bedeutet dies, dass ein Benutzer die beiden zu verbindenden Elemente ergreift und diese auf der Oberfläche von ihrer ursprünglichen Position aus kurz zusammenführt um sie danach sofort wieder an ihre Originalposition zurück zu stellen. Die so erstellten Verbindungen sind immer ungerichtet und werden nach der Erstellung gleich behandelt wie mit den Markierungstokens erstellte Verbindungen. Durch die Vermeidung der Verwendung von Werkzeugtokens ist dieser Weg zur Herstellung von Verbindungen schneller auszuführen als der zuvor beschriebene (etwa um den Faktor 5). Problematisch wird diese Art der Interaktion dann, wenn die Modellierungsoberfläche bereits sehr voll ist, so dass nur noch wenig Platz zur Zusammenführung von zwei Elementen vorhanden ist. Auch bei Bedarf nach gerichteten Verbindern muss auf Markierungstokens zurückgegriffen werden.

Markierungen können unabhängig von ihrem Erstellungsweg auch benannt werden. Da sie keine physische Repräsentation besitzen, kann der Benennungsmodus mit Haftnotizen nicht verwendet werden. Die Eingabe von Bezeichnungen erfolgt demnach ausschließlich über die Tastatur. Zur Benennung einer Verbindung muss unmittelbar nach deren Erstellung die Bezeichnung eingegeben werden, diese wird dann der zuletzt erstellten Verbindung zugewiesen. Um eine Bezeichnung zu ändern oder im Nachhinein zuzuweisen, muss der Verbinder mittels dem Markierungstoken ausgewählt werden. Durch nachträgliches Aufsetzen des Pfeilspitzen-Tokens auf eine existierende Verbindung wird am jeweils näheren Verbindungsende eine Pfeilspitze hinzugefügt oder entfernt werden.

6.3.4. Löschen von Elementen und Verbindungen

Beim Löschen muss zwischen dem Entfernen von Elementen und Verbindungen unterschieden werden. Beide Modellaspkte sind nicht nur hinsichtlich der Interaktion mit dem System unterschiedlich zu behandeln, es besteht vor allem eine Existenzbeziehung zwischen Elementen und Verbindungen, d.h. dass Verbindungen nicht ohne ihre als Endpunkte dienenden Elemente existieren können und beim Entfernen eines Elements ebenfalls gelöscht werden müssen.

Das Löschen eines Modellelements aus dem aktuellen Modell erfolgt durch Entfernen der zugehörigen Tokens auf der Oberfläche. Hier muss zwischen Entfernungsvorgängen unterschieden werden, die vom Benutzer intendiert sind und solchen, die durch die Manipulation des Modells unabsichtlich und kurzzeitig auftreten. Nur in ersterem Fall dürfen die Verbindungen, die an einem Element hängen, ebenfalls gelöscht werden. Die Unterscheidung zwischen den beiden Fällen wurde durch die Einführung eines Zeitintervalls von 5 Sekunden gelöst, nachdem die betroffenen Verbinder ebenfalls entfernt werden. Wird also ein Modellelement nur an eine andere Stelle verschoben (und verschwindet so kurzzeitig aus dem Blickfeld der Kamera), bleiben die Verbindungen erhalten. Wird das Element zur Seite gestellt, wird es und seine abhängigen Verbindungen zwar auf den Ausgabekanälen nicht mehr dargestellt, das tatsächliche Löschen erfolgt aber erst nach 5 Sekunden. Wird das Element innerhalb dieser Zeitspanne wieder auf der Oberfläche platziert, wird es und seine abhängigen Verbindungen so behandelt, als ob es nicht verschwunden gewesen wäre.

Das Löschen von Verbbindern ist durch den Einsatz der Lösch-Tokens möglich, dass das System in den Lösch-Modus schaltet sobald es auf der Oberfläche platziert wird (und diesen wieder deaktiviert, wenn es entfernt wird). Im Lösch-Modus wird der Einsatz von Markierungstokens alternativ interpretiert. Anstatt der Herstellung einer Verbindung wird bei Markierung von zwei Modellelementen eine eventuell zwischen diesen vorhandene Verbindung endgültig entfernt. Dabei ist keine Unterscheidung zwischen gerichteten und ungerichteten Verbindungen notwendig, es können durchgängig die einfachen Markierungstokens eingesetzt werden.

6.3.5. Einbettung von Zusatzinformation

Der Interaktionsablauf zur Einbettung von Zusatzinformation setzt sich aus zwei Teilen zusammen, die sequentiell abzuarbeiten sind. Der erste Schritt ist der Vorgang des Bindens von Zusatzinformation an ein einbettbares Token. Im zweiten Schritt muss dieses Token in einem Container-Token eingebettet werden. Ein dritter Interaktionsablauf in diesem Zusammenhang ist das Abrufen bzw. die Manipula-

6. Eingabe und Interpretation

tion der eingebetteten Information, der in der Folge ebenfalls im Detail beschrieben wird.

Das Binden von Information an ein einbettbares Token ist ein Vorgang, der nicht ausschließlich in der physischen Welt durchgeführt werden kann. Die anzubindende Information liegt im Regelfall digital vor und muss damit am Rechner selektiert werden. Der zugehörige Informationsablauf beginnt mit der Auswahl eines einbettbaren Tokens, dessen Form die Art der einzubettenden Information bestimmt. Im konkreten Fall wurden drei Arten von einbettbaren Tokens erstellt, von denen einer der Umsetzung der Abstraktion von Modellen (also der Einbettung von Teilmodellen in ein Containerelement) dient und die anderen beiden exemplarisch die Anbindung von digitalen Ressourcen aus der Arbeitsumgebung demonstrieren. Im Sinne des Interaktionsablaufs sind diese Arten zum Teil unterschiedlich zu behandeln. Bei allen Token startet die Interaktion mit der Registrierung des Tokens mittels der Registrierungskamera. Handelt es sich dabei um ein Token, das der Einbettung von Teilmodellen dient, wird der aktuelle Modellzustand auf der Modellierungsoberfläche erfasst, gespeichert und an das Token gebunden. Der Vorgang der Informationsanbindung ist in diesem Fall abgeschlossen. Daneben existieren Tokens, an die beliebige digitale Ressourcen in der Form von Dateien gebunden werden können. Die Registrierung eines derartigen Tokens öffnet in diesem Fall eine Dialogbox am Rechner, in der die digitale Ressource ausgewählt werden kann. Die gewählte Datei wird dann über eine Referenz an das Token gebunden. Die dritte Art von einbettbaren Tokens demonstriert die Anbindung von spezialisierterer Information, die in einem Anwendungsfall sinnvoll sein können und ggf. auch gesondert behandelt werden müssen. Im konkreten Fall wurde die Anbindung von unmittelbar aufgenommenen Bildern implementiert, die so z.B. eine Abbildung eines Ausschnittes der realen Arbeitsumgebung in das Modell ermöglichen. Im Interaktionsablauf löst die Registrierung eines derartigen Tokens die Speichlung eines Bildes, das aktuell von der Registrierungskamera erfasst wird, aus. Dieses Bild wird an das Token gebunden, der Interaktionsablauf ist damit ebenfalls wieder beendet.

Im zweiten Schritt, der zur Einbettung von Information notwendig ist, muss das einbettbare Token einem Containertoken zugeordnet werden. Dazu muss das betreffende Container-Token geöffnet werden. Das einbettbare Token, an das bereits Information gebunden wurde, wird erneut mit der Registrierungskamera erfasst, was eine Zuordnung zwischen diesem Token und dem offenen Container auslöst. Visuelles Feedback über die Ausgabekanäle zeigt eine erfolgreiche Zuordnung an. Das einbettbare Token kann nun auch physisch in den Container gelegt werden und ist diesem damit auch in der realen Welt zugeordnet. Der Interaktionsvorgang endet mit dem Schließen des Containertokens. Der hier beschriebene Vorgang ist insofern als Hilfskonstrukt zu sehen, als das er nicht notwendig wäre, wenn die Containertoken in der Lage wären, ihren Inhalt selbst zu identifizieren. Wie in Abschnitt 6.1.1

argumentiert, wurde aber auf diese Funktionalität verzichtet. Dies ist für den prototypischen Einsatz akzeptabel, im Realbetrieb jedoch insofern problematisch, als dass durch den indirekten, mehrstufigen Zuordnungsprozess ein inkonsistenter Zustand zwischen physischer und digitaler Modellrepräsentation nicht ausgeschlossen werden kann.

Um eingebettete Information wieder abrufen zu können, wurde ebenfalls ein Interaktionsablauf definiert. Dieser beginnt mit dem Öffnen des entsprechenden Container-Tokens. Durch Herausnehmen des informationstragenden Tokens und eine Erfassung desselben mit der Registrierungskamera wird die angebundene Information abgerufen. Handelt es sich um ein Token mit angebundenen Teilmodellen, so wird dieses über die Output-Kanäle dargestellt. Angebundene digitale Ressourcen werden am Rechner geöffnet und dargestellt. Das eingebettete Token bleibt dem Container-Token zugeordnet, es muss also zur Konsistenzsicherung nach dem Abrufen der Information wieder in den Container gelegt werden. Um ein eingebettetes Token aus einem Container zu entfernen, muss bei der Erfassung dieses Tokens das Radierer-Werkzeug-Token eingesetzt werden und das System so in den Lösch-Modus versetzt werden. Die Information bleibt dabei an das einbettbare Token gebunden. Eine Veränderung der angebundenen Information während eines Modellierungsvorgangs ist nicht vorgesehen, es muss ggf. ein weiteres, noch unbelegtes einbettbares Token verwendet werden.

6.3.6. Kontrolle der Modellierungshistorie

Die Kontrolle der Modellierungshistorie umfasst im Wesentlichen drei Interaktionsblöcke, die im Folgenden beschrieben werden. Der erste Block behandelt die Erfassung von Modellzuständen, im zweiten Block wird die Navigation durch die Modellierungshistorie behandelt und der dritte Block beschäftigt sich mit der Unterstützung der Wiederherstellung von gespeicherten Modellzuständen.

Die Erfassung von Modellzuständen erfolgt im Normalfall implizit und muss von den Benutzern nicht explizit ausgelöst werden. Wenn sich das Modell auf der Modellierungsoberfläche länger als 5 Sekunden in einem stabilen Zustand befinden (wenn also weder die Position, noch die Rotation oder die Benennung der Tokens geändert wurden und auch keine Verbindungen oder Einbettungen erstellt wurden), wird der aktuelle Modellzustand gespeichert. Sollen bestimmte Modellzustände aber aufgrund ihrer Relevanz für den Modellierungsverlauf explizit gespeichert und auch entsprechend gekennzeichnet werden, so kann das Snapshot-Token eingesetzt werden. Dieses löst sobald es auf der Modellierungsoberfläche aufgesetzt wird die Speicherung des aktuellen Modellzustandes aus. Das System quittiert die Speicherung mit visuellem Feedback über die Ausgabekanäle. Explizit erfasste Zustände werden

6. Eingabe und Interpretation

im Gegensatz zu automatisch gespeicherten gekennzeichnet und in den weiterverarbeitenden Modulen (vor allem bei der Persistierung) speziell behandelt.

Die Navigation durch die Modellierungshistorie erfolgt mittels einem runden Werkzeug-Token, dass diese Navigation durch Rotation im oder entgegen dem Uhrzeigersinn ermöglicht. Das System wird durch das Aufsetzen dieses Tokens auf die Modellierungsoberfläche in den Historien-Modus geschaltet, was Auswirkungen auf die Darstellung des Modells in den Outputkanälen (siehe Abschnitt XY) hat. Bei Aktivierung des Historien-Modus wird der aktuellste gespeicherte Modellzustand geladen. Durch Drehen des Tokens gegen den Uhrzeigersinn kann in der Zeit zurück navigiert werden. Das Laden der nächstälteren Modellzustands erfolgt dabei in regelmäßigen Intervallen von jeweils 45 Grad (8 Modellzustände pro voller Umdrehung). Der absolute zeitliche Abstand zwischen den gespeicherten Modellzuständen, der sich aus dem Auftreten stabiler Modellzustände ergibt, wird bei der Navigation nicht berücksichtigt – der zurückzulegende Winkel ist immer gleich groß. Gleichermaßen gilt für das Drehen des Tokens im Uhrzeigersinn – dabei wird in regelmäßigen Abständen der nächstjüngere Modellzustand geladen. Wird das Token entfernt, wird der Historien-Modus deaktiviert und wieder in der aktuelle Systemzustand an die Output-Kanäle übergeben.

Soll ein gespeicherter Modellzustand wiederhergestellt werden, so bietet das System Unterstützung für die dazu notwendige Interaktion. Da weder die Tokens noch der Tisch selbst über durch den Rechner steuerbare bewegliche Komponenten verfügen, die eine automatisierten Wiederherstellung eines Modellzustandes ermöglichen würden, kommt die Aufgabe der eigentlichen Wiederherstellung den Benutzern zu. Das System unterstützt dabei insofern, als das über die Ausgabekanäle Information an die Benutzer weitergegeben wird, die die korrekte Platzierung der Elemente anleitet (Details dazu sind in Abschnitt XY nachzulesen). Gespeicherte Modellzustände können dabei entweder aus dem Historienmodus oder aus an eingebettete Tokens gebundene Teilmodelle bezogen werden. Im ersten Fall wird die Wiederherstellungsunterstützung durch ein spezielles Token aktiviert, das auf die Oberfläche aufgesetzt wird während der alte Systemzustand über die Outputkanäle dargestellt wird. Im zweiten Fall muss das Teilmmodell, das an das eingebettete Token gebunden ist, über die Registrierungskamera abgerufen werden, wodurch es ebenfall über die Outputkanäle dargestellt wird. Während der Darstellung wird die Wiederherstellungsunterstützung wiederum durch Aufsetzen des betreffenden Werkzeug-Tokens auf die Oberfläche aktiviert. In beiden Fällen läuft dann die Wiederherstellungsunterstützung durch das System angeleitet automatisiert ab. Nach Abschluss der Wiederherstellung werden die Benutzer aufgefordert eventuell noch vorhandene Werkzeug-Tokens (wie das Historien-Navigations-Token) zu entfernen, wodurch das System wieder den aktuellen Modellzustand als Grundlage der Darstellung verwendet.

6.4. Erfassung der Benutzerinteraktion durch Software

Das Software-System, das die Benutzereingaben von der Hardwareschnittstelle bis hin zur abstrahierten Modellierungsinformation aufbereitet, ist Gegenstand dieses Abschnitts. Die darüber hinausgehenden Komponenten sind Gegenstand der Kapitel X und Y. Die hier behandelten Software-Komponenten umfassen

- jene Komponente, die die Rohdaten von ReacTIVision empfängt und sie hinsichtlich der Modellierungsinformation auswertet und interpretiert,
- die Komponente, die für die Stabilisierung der Erkennungsleistung sorgt, wenn Fehlerkennung oder Erkennungsausfälle auftreten, sowie
- jene Komponente, die für das Tracking des Modellzustandes und damit für die Nachvollziehbarkeit des Modellierungsablaufs zuständig ist.

Wie in Abschnitt 6.1.3 ausgeführt, soll zur Verknüpfung dieser Komponenten das TUIpist-Framework zum Einsatz kommen. Im vorliegenden Prototypen wurde das System jedoch zwar modular aber ohne Einsatz des TUIpist-Frameworks implementiert. Eine erste Umsetzung der Architektur auf die TUIpist-Strukturen liegt vor, ist jedoch noch nicht im produktiven Einsatz. An dieser Stelle wird deshalb der Aufbau der Komponenten in ihrer derzeitigen Implementierung beschrieben, da diese auch bei der Evaluierung des Werkzeugs eingesetzt wurde. Erste Umsetzungserfahrungen deuten darauf hin, dass der Aufwand zur Einbindung von TUIpist sich wie erwartet in Grenzen hält.

Die Systemarchitektur, wie sie sich in der derzeitigen Implementierung darstellt ist in Abbildung 6.18 dargestellt. Die Komponenten des Interpretations-Moduls leiten sich aus den umzusetzenden Funktionen und zu erkennenden Interaktionsabläufen (siehe Abschnitt 6.3) ab. Die Komponenten sowie die Verknüpfungen untereinander werden in den folgenden Abschnitten im Detail beschrieben, wobei die Struktur der Beschreibung im Gegensatz zum vorangegangenen Abschnitt nicht an der Funktionalität sondern an den involvierten Technologien orientiert ist und damit einen weiteren Detaillierungsschritt in der Beschreibung der Benutzereingabe im hier vorgestellten System darstellt.

6.4.1. Interpretation der Rohdaten

Das ReacTIVision-Framework liefert wie in Abschnitt 6.1.2 beschrieben die aus dem Kamerabild extrahierten Informationen über die ausan. Für jeden erkennbaren Code wird eine separate Nachricht gesandt, sobald sich zumindest ein Parameter (also Position oder Rotation) verändert. Diese Nachrichten treffen über die UDP-

6. Eingabe und Interpretation

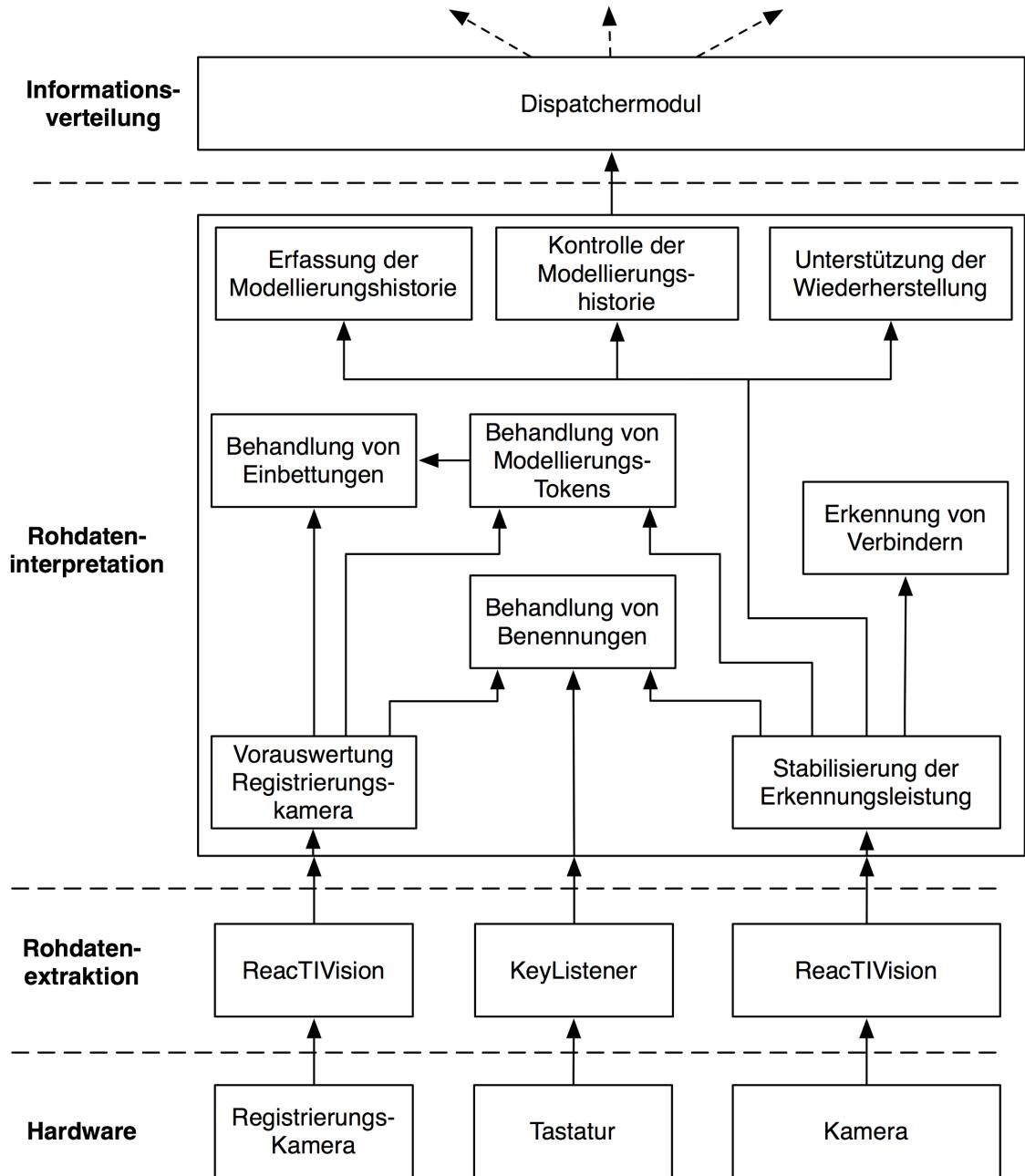


Abbildung 6.18.: Softwarearchitektur zur Erkennung von Benutzerinteraktion

Schnittstelle bei einem von ReacTIVision bereitgestellten Java-Objekt ein, das die Auswertung der Nachricht übernimmt und über einen Callback-Mechanismus⁶ vom Anwendungsentwickler implementierte Methoden aufruft, die auf das jeweilige Ereignis reagieren. Die Ereignisse, die auftreten können, beschränken sich auf das Erscheinen eines Tokens, die Änderung der Parameter eines Tokens und das Verschwinden eines Tokens sowie die gleichen drei Ereignisse für die Behandlung eines Cursors (wie das Markierungs-Werkzeugtoken). Dazu werden jeweils die relevanten Parameter zur Verfügung gestellt (im Wesentlichen die ID des Tokens, seine Position, seine Rotation und einige weitere abgeleitete Parameter, die in der konkreten Anwendung nicht eingesetzt werden).

In den betreffenden Methoden wird im ersten Schritt ausgewertet, ob das identifizierte Token ein Modellierungs-Element oder ein Werkzeug ist. Im Falle eines Modellierungs-Elements wird im nächsten Schritt festgestellt, um welche Art von Token es sich handelt und ob es im aktuellen Modellierungsvorgang bereits im Einsatz war (ob z.B. bereits Information über die Benennung des Tokens oder dessen Inhalt vorhanden ist). Diese Information wird ggf. geladen und den weiterverarbeitenden Komponenten zu Verfügung gestellt. Im Falle eines Werkzeug-Tokens wird dieses identifiziert und an die für die Behandlung zuständige Methode weitergeleitet. Diese Methoden schalten das System je nach Art des eingesetzten Tokens in einen alternativen Zustand oder lösen eine Aktion auf den übergeordneten Ebenen aus.

Ist für die Interpretation einer Benutzerinteraktion mehr als ein Token notwendig (z.B. beim Ziehen von Verbindungen), werden die Kandidaten, die möglicherweise den Beginn einer derartigen Interaktionssequenz darstellen in einen Buffer geschrieben. Beim Auftreten eines Tokens, das möglicherweise den zweiten Teil dieser Interaktion darstellt, wird dieser Buffer geprüft und die Interpretation ggf. auf Basis der gespeicherten und aktuellen Token-Daten durchgeführt.

Manche Benutzereingaben müssen – um eine komfortable Erstellung des Modells zu ermöglichen – ggf. auch nach Entfernen eines Tokens für einen bestimmten Zeitraum gespeichert werden. Beispiele hierfür sind Markierungen, die auch nach Entfernen des Markierungstokens aktiv bleiben um Zeit für die Benennung oder die Herstellung einer Verbindung zu geben oder auch Modellierungstoken, die durch Manipulation temporär aus dem Erkennungsbereich der Kamera geraten und in diesem Fall nicht sofort als „entfernt“ interpretiert werden sollen. Auch in diesem Fall werden Buffer eingesetzt, die die entsprechende Information mit einem Zeitstempel versehen aufnehmen. Zur Sicherstellung der Modellkonsistenz (also der Überein-

⁶entsprechend dem in (Gamma et al., 1995) beschriebenen Beobachter-Muster (Observer), das bei der Entwicklung wiederverwendbarer objektorientierter Software als Entwurfsmuster zum Einsatz kommt

stimmung zwischen physischem und digitalem Modell) prüfen speziell dazu konzipiert Methoden, die von einem Timer in regelmäßigen Abständen aufgerufen werden, ob die vorgegebene Gültigkeitsdauer der Information bereits abgelaufen ist und entfernen diese ggf. endgültig aus dem Buffer und dem Modell (löschen also konkret eine Markierung oder entfernen ein Modellelement und alle abhängigen Verbindungen aus dem Modell). Die Zeitintervalle dieser Prüfungen sind auf den jeweiligen Anwendungsfall abgestimmt und bewegen sich im Bereich zwischen drei und fünf Sekunden.

6.4.2. Stabilisierung der Erkennungsleistung

Da wie bereits beschrieben beim Einsatz von ReacTIVision bei grenzwertigen Beleuchtungsbedingungen zu Erkennungsproblemen oder Fehlerkennungen neigt, wurden im Programmcode des Interpretationsmoduls Mechanismen verankert, die potentielle Fehlerkennungen und Erkennungsprobleme identifizieren und deren Auswirkungen ignorieren sollen. Im Folgenden werden häufige Fehlerquellen benannt und der jeweilige Lösungsansatz beschrieben.

Bildrauschen

Bei wenig vorhandenem Umgebungsumgebungslicht müssen die Kameraparameter wie oben bereits beschrieben soweit nachgeführt werden, bis wieder ausreichender Bildkontrast vorhanden ist. Zur Nachführung eignen sich die Blende der Kamera und die Signalverstärkung. Eine offene Blende erhöht den Lichteinfall auf den Sensor und verbessert damit die Bildhelligkeit und den Kontrast. Je weiter offen die Blende eingestellt ist, desto geringer ist jedoch auch die Tiefenschärfe des Kamerabilds, d.h. dass die Kamera sehr exakt auf die Modellierungsoberfläche fokussiert werden muss. Da der Abstand von der Kamera zu Mitte des Tisches geringer ist als zu den Randbereichen, kann bei offener Blende nie die gesamte Oberfläche scharf eingestellt werden. Da ein unscharfes Bild die Erkennungsleistung massiv mindert, sind der Nachführung des Blenden-Parameters Grenzen gesetzt. Die Signalverstärkung wirkt demgegen nicht direkt auf die Kamera sondern auf das von ihr gelieferte Bild und ist deswegen keine physikalischen Beschränkungen unterworfen. Bei der Erhöhung der Signalverstärkung werden helle Bereiche im Bild heller, dunkle bleiben eher dunkel, der Kontrast wird also erhöht. Problematisch ist in diesem Kontext, dass nicht nur das Nutzsignal (also das eigentliche Bild) verstärkt wird, sondern auch das im Bild enthaltene Rauschen. Das Rauschen ist wiederum ein physikalisches Phänomen der digitalen Bilderfassung und hat seinen Ursprung im bildaufnehmenden Chip. Während das Rauschen bei guten Lichtverhältnissen im Vergleich zum Nutzsignal vernachlässigbar ist, wirkt es bei schlechten Lichtverhäl-

nissen und dementsprechend starker Signalverstärkung störend. Generell ist die Kamera deshalb so zu kalibrieren, dass der Spielraum bei der Blendeneinstellung maximal ausgenutzt wird (so dass die gesamte Oberfläche noch scharf erscheint) und erst danach die zu einer stabilen Erkennung notwendige Signalverstärkung eingestellt wird.

Ist trotz maximaler Blendenöffnung so wenig Licht vorhanden, dass es durch eine hohe notwendige Verstärkung zu Bildrauschen kommt, ergeben sich Probleme bei der Bilderkennung. Diese äußern sich teilweise im temporären Nicht-Erkennen von auf der Oberfläche platzierten Tokens (weil das den Code enthaltende Nutzsignal im Bildrauschen untergeht), vor allem aber im Auftreten von „Cursorflackern“. Mit „Cursorflackern“ wird hier das Phänomen bezeichnet, das der Bilderkennungsalgorithmus vermeintlich auf die Oberfläche aufgesetzte Cursor-Tokens (wie das Markierungstoken) erkennt und an das Interpretationsmodul weiterleitet. Es kann so vorkommen, dass fehlerhafte Markierungen angezeigt werden oder sogar umgewollte Verbindungen entstehen. Diese Cursor haben eine sehr geringe Lebensdauer (im Bereich von unter 500 Millisekunden), da das Rauschen zufallsverteilt und dynamisch ist und eine als Cursor erkannte Konstellation dementsprechend nicht lange erhalten bleibt.

Um das „Cursorflackern“ zu verhindern kann einerseits auf Seiten des ReacTIVision-Frameworks die Cursor-Größe erhöht werden, was die Wahrscheinlichkeit einer Fehlerkennung durch die größere notwenige Fläche, auf der das Rauschen in der richtigen Form auftreten muss, reduziert. Dementsprechend müssen aber auch etwaige Cursor-Tokens in ihrer Größe angepasst werden, was nur in Grenzen möglich ist. Der alternative Ansatz zur Kompensation von „Cursorflackern“ wurde im Interpretationsmodul implementiert und basiert auf der Messung der Lebensdauer des Cursors. Wird von ReacTIVision das Auftreten eines Cursors gemeldet, so wird dieser vom Interpretationsmodul nicht direkt ausgewertet sondern in einen Buffer geschrieben. Nach 600 Millisekunden wird geprüft, ob der Cursor noch vorhanden ist oder ob er bereits wieder entfernt wurde (ob also eine entsprechende Nachricht von ReacTIVision weitergegeben wurde). Im ersten Fall kann von einem bewusst gesetzten Cursor ausgegangen werden und dieser zur Weiterverarbeitung freigegeben werden. Im zweiten Fall handelt es sich mit hoher Wahrscheinlichkeit um „Cursorflackern“, das ignoriert werden kann.

Technisch ist diese Funktion so umgesetzt, dass eine Meldung über das Auftreten eines Cursors ein entsprechendes Objekt mit Zeitstempel in einen Buffer schreibt. Die Meldung über das Verschwinden eines Cursors entfernt dieses Objekt dementsprechend wieder aus dem Buffer. Ein Timer löst alle 300 Millisekunden eine Prüfung des Buffers aus, bei dem alle Cursor, die älter als 600 Millisekunden sind, zur Weiterverarbeitung frei gegeben werden. Alle jüngeren Cursor werden unverändert im Buffer belassen und – sofern dann noch vorhanden – im nächsten Durchlauf

6. Eingabe und Interpretation

erneut geprüft und ggf. frei gegeben. Durch den potentiell simultanen Zugriff auf den Buffer (durch die Prüfungsroutine und den Cursor hinzufügenden oder entfernenden Methoden) müssen hier synchronisierte, dass heißt simultaner Veränderung gegenüber stabile Buffer-Klassen eingesetzt werden.

Spotlights

Bei ungleichmäßiger Umgebungsbeleuchtung treten auf der Modellierungsoberfläche häufig kleinere Bereiche auf, in denen die Helligkeit gegenüber der Umgebung erhöht ist. Derartige Bereiche entstehen unter anderem durch einfallendes Sonnenlicht, dass durch unzureichende Verdunkelungsmaßnahmen nur zum Teil abgeschirmt wird oder auch durch Reflexionen, die durch die Lampe des Videoprojektors verursacht werden. In beiden Fällen führen diese Bereiche, die im Folgenden als „Spotlights“ bezeichnet werden, zur Fehlerkennung von Cursorn auf der Oberfläche. Im Gegensatz zum zuvor beschriebenen „Cursorflattern“ bleiben diese „Spotlights“ jedoch über die Zeit stabil und werden vom ReacTIVision-Framework als ständig vorhandene Cursor identifiziert. Bleibt diese Fehlerkennung unkompenziert, so wird ein Token, das im Bereich des „Spotlights“ aufgesetzt oder dorthin bewegt wird, ohne weitere Benutzerinteraktion markiert (also ob es mit dem Markierungstoken ausgewählt werden würde). Dies führt wie zuvor zu Problemen bei der Benennungen von Tokens bzw. zu unbeabsichtigten Verbindungen.

Um die Fehlerkennung von Cursorn durch Spotlights zu verhindern, kann einerseits wiederum die Möglichkeit der oben bereits beschriebenen Konfiguration der Cursorgröße im ReacTIVision-Framework genutzt werden. Da diese jedoch den ebenfalls erwähnten Einschränkungen unterliegt, ist eine Behandlung im Interpretationsmodul notwendig. Der Lösungsansatz zur Kompensation beruht wiederum auf einer Messung der Lebensdauer des betreffenden Cursors. Im Gegensatz zur Behandlung des „Cursorflackers“, bei dem eine minimale Lebensdauer festgelegt wurde, wird bei der Behandlung von „Spotlights“ eine maximale Lebensdauer definiert, nach deren Überschreitung ein Cursor ignoriert wird. Diese Grenze ist mit fünf Sekunden festgelegt – jeder Cursor, der länger auf der Oberfläche identifiziert wird, wird ignoriert. Spotlights sind somit maximal fünf Sekunden nach ihrem Auftreten wirksam. Wenn sich in diesem Zeitraum kein Token in der Nähe des Spotlights befindet, bemerken die Benutzer die Fehlerkennung nicht. Befindet sich zum Zeitpunkt der Erkennung ein Token in der Nähe, so wird dieses einmalig markiert, wobei diese Markierung durch den in Abschnitt 6.3.3 beschriebenen Mechanismus zur Begrenzung der Lebensdauer von Markierungen nach spätestens drei Sekunden wieder verschwindet.

Technisch wurde diese Kompensationsmaßnahme als Erweiterung des oben beschriebenen Buffer-Ansatzes umgesetzt. Bei der Prüfung der Gültigkeit eines Cursor

im Buffer wird nun neben der minimalen Lebensdauer auch die maximale Lebensdauer miteinbezogen. Ist die maximale Lebensdauer überschritten, so wird der Cursor aus dem Buffer entfernt und damit so behandelt, als ob er von den Oberfläche verschwunden wäre. Diese Maßnahme ist dann wirksam, wenn die „Spotlights“ tatsächlich über die Zeit stabil erkannt werden – setzt die Erkennung eines „Spotlights“ kurzfristig aus, so beginnt mit der Wiedererkennung desselben die oben beschriebene 5-Sekunden-Frist bis zum Ignorieren des erkannten Cursors von neuem. Um diese potentiellen Problematik zu begegnen, werden die Positionen von bereits erkannten Spotlights ebenfalls in einem Buffer abgelegt. Tritt ein neuer Cursor an der Stelle eines bereits bekannten Spotlights auf, wird dieser sofort ignoriert. Dabei kann die Applikation über die Laufzeit theoretisch zu restriktiv bei der Akzeptanz von Cursoren umgehen, vor allem wenn Markierungstokens regelmäßig länger als fünf Sekunden auf der Oberfläche belassen werden und die zugehörigen Cursor dadurch als Spotlights erkannt und deren Positionen gesperrt werden. In der Praxis ist dieses Vorgehen jedoch kein Problem, da manuell platzierte Token kaum oder nur zufällig so positionsstabil auf der Oberfläche platziert werden können, dass exakt die gleiche (und demnach gesperrte) Position getroffen wird. „Spotlights“ sind demhingegen hoch positionsstabil und werden dadurch durch diese Maßnahme erfasst.

Ein weiteres Problem, das durch „Spotlights“ verursacht wird, die von unten auf die Oberfläche geworfen werden (also z.B. durch den Videoprojektor verursacht werden) ist eine potentielle Verdeckung von Teilen eines ReacTIVision-Codes, wodurch das betroffene Token nicht mehr erkannt werden kann. Da die Störung permanent ist, kann diese durch Software nicht kompensiert werden und muss durch physische Maßnahmen kompensiert werden. Im konkreten Fall wurde das Spotlight, dass durch den Videoprojektor verursacht wurde, durch eine punktuelle Abdeckung am Umlenkspiegel (siehe Abbildung 6.6) verhindert. Diese Maßnahme verhindert zwar auch eine Projektion im betroffenen Bereich, diese Beeinträchtigung ist aber insofern akzeptabel, als dass auf der Oberfläche ein Bereich von lediglich 0,5 cm im Durchmesser betroffen ist.

Überstrahlte Bereiche

Überstrahlte Bereiche können auftreten, wenn sich externe Lichtquellen direkt über der Tischoberfläche befinden oder bei internen (im Werkzeug eingebauten) Lichtquellen keine Streuscheiben verwendet werden. Außerdem kann auch durch direkte Sonneneinstrahlung ein Überstrahlen auftreten. In Abgrenzung zu „Spotlights“ sind überstrahlte Bereiche großflächiger (potentiell die gesamte Oberfläche) und werden nicht als Cursor erkannt. Überstrahlte Bereiche verursachen jedoch andere Probleme, die im Folgenden behandelt werden.

6. Eingabe und Interpretation

Das größte Problem bei überstrahlten Bereichen ist die Verhinderung der Erkennung von Codes im betroffenen Teil der Oberfläche. Überstrahlte Bereiche erscheinen am Kamerabild rein weiß, unabhängig von den aktuell aufgesetzten Tokens. Dabei ist es unwesentlich, dass externe Lichtquellen von Tokens teilweise verdeckt werden – die Überstrahlungseffekte beeinflussen den adaptiven Erkennungsalgorithmus negativ und verhindern so trotzdem eine Erkennung. Von Seiten der Interpretationsschicht können keine Gegenmaßnahmen ergriffen werden, da auch über die Zeit hinweg im Normalfall keine Daten vorliegen, aus denen der Systemzustand im Falle von Fehlerkennungen extrapoliert werden könnte. Softwareseitig ist damit nur eine Beeinflussung über die Kalibrierungsparameter des ReacTIVision-Frameworks möglich. Auch hier ist der Spielraum insofern eingeschränkt, als dass eine Veränderung der Signalverstärkung keine Verbesserung bringt, da überstrahlte Bereiche bereits den Bilderfassungsschip der Kamera übersteuern und damit im betroffenen Bereich keine Information vorhanden ist, die durch Reduzierung der Bildverstärkung sichtbar gemacht werden könnte. Der einzige Parameter, dessen Veränderung in diesem Zusammenhang sinnvoll ist, ist die Blendenöffnung der Kamera. Durch Schließen der Blende fällt weniger Licht auf den Sensorchip – das Bild wird dunkler. Jedoch werden auch nicht überstrahlte Bereiche abgedunkelt, was die Erkennung von Codes in diesen Teilen der Oberfläche schwieriger macht. Hier kann wiederum eine Erhöhung der Signalverstärkung Abhilfe schaffen, die aber exakt mit den Grenzwerten der Erkennbarkeit in den ehemals überstrahlten Bereichen abgestimmt werden muss.

Ein zweites Problem, dass bei nicht vollkommen überstrahlten Bereichen auftreten kann, ist das zuvor bereits beschriebene „Cursorflackern“. Dessen Auftreten liegt in der Funktionsweise des adaptiven Erkennungsalgorithmus von ReacTIVision begründet, der in Bereichen mit geringem Kontrast (wie es nicht vollkommen aber beinahe überstrahlte Bereiche sind) auf geringe Strukturunterschiede im Bild relativ stark reagiert. Als Gegenmaßnahmen kommt auf Seite von ReacTIVision die wiederum die Veränderung der Blende oder der Signalverstärkung in Frage, auf Seiten des Interpretationsmoduls werden die Maßnahmen wirksam, die auch bei herkömmlichen „Cursorflackern“ zum Einsatz kommen (siehe oben).

Bildverzerrungen

Bildverzerrungen werden durch das eingesetzte Objektiv der Kamera ausgelöst. Je größer der Öffnungswinkel des Objektivs, desto höher sind bei Standardobjektiven die Verzerrungseffekte im Randbereich des erfassten Gebietes. Im konkreten System kommt ein Objektiv mit starkem Weitwinkel zum Einsatz, das die Erfassung der gesamten Oberfläche aus relativ geringem Abstand (und damit geringerer Tischhöhe) ermöglicht. Verzerrungseffekte führen im Normalfall zu einer Fehlposi-

tionierung von Token im Randbereich, da das Kamerabild im Vergleich zur realen Welt gestaucht erscheint. ReacTIVision bietet dazu die Möglichkeit, mit Hilfe eines Kalibrierungs-Musters eine Entzerrungsmatrix zu erfassen, die zur Kompensation dieses Fehlers verwendet werden kann. Da die Bildverzerrung sich zeitlich nicht verändert, ist keine weitergehende Kompensation dieser Fehlerkennung in höheren Softwareebenen (wie dem Interpretationsmodul) notwendig.

Problematischer wird die Bildverzerrung dann, wenn durch sie in den Randbereich soviel Bildinformation (durch die Stauchung des Bildes) verloren geht, dass eine Erkennung von in diesen Randbereichen platzierten Tokens nicht mehr möglich ist. Dies tritt auf, wenn die Stauchung so stark wird, dass die schwarzen und weißen Bereich des Codes nicht mehr trennscharf voneinander zu unterscheiden sind und ineinander fließen. Je kleiner der Code im Bild dargestellt ist (also je weiter entfernt er ist, je kleiner er physisch ist oder je geringer die Kameraauflösung ist), desto eher tritt dieser Effekt auf. Da im gegebenen System sowohl die Entfernung zwischen Kamera und Oberfläche als auch die Kameraauflösung vorgegeben sind, ist der einzige zu variierende Faktor die physische Größe des Codes. Diese wird jedoch von den Dimensionen der Tokens bestimmt, die im konkreten System im Interesse der Handhabbarkeit der Modellierungselemente eher gering gewählt wurden. Der nicht erkennbare Bereich kann nur durch Einsatz einer höher auflösenden Kamera oder durch größere Tokens erschlossen werden. Im ersten Fall würde der Auswertungsaufwand und damit die notwendige Rechenleistung massiv erhöht, im zweiten Fall wird die Handhabbarkeit negativ beeinflusst und die Anzahl der gleichzeitig einsetzbaren Tokens reduziert. Softwareseitig kann aufgrund der fehlenden Information in den Quelldaten keine Kompensation vorgenommen werden.

6.4.3. Erkennung von Markierungen und Verbindungen

Wie im Abschnitten 6.3.2 und 6.3.3 beschrieben, werden ReacTIVision-Cursor verwendet, um eine oder mehrere Elemente auf der Tischoberfläche im Zuge eines Interaktionsablaufs zu markieren. ReacTIVision-Cursor sind helle, kreisrunde Bildbereiche eines bestimmten Durchmessers. Diese Bildbereiche können durch Marker erzeugt werden, in dem ein leerer weißer Kreis von einem schwarzen Ring umschlossen wird. Alternativ erzeugt auch das Aufsetzen von Fingern derartige Bereiche auf der Oberfläche, wodurch grundsätzlich die Realisierung eines (Multi-)Touch-Displays möglich wäre. Da durch variierende Fingergrößen und verschiedene Arten des Aufsetzens des Fingers (und damit verschiedenen Abdrücken auf der Oberfläche) bei unterschiedlichen Modellierern die Erkennung von Fingern nur teilweise stabil und zuverlässig funktioniert, wurde im konkreten System die Marker-Variante zur Realisierung von Cursors gewählt. Diese ist aufgrund der definierten Größe

6. Eingabe und Interpretation

des hellen Bildbereichs und der damit einhergehenden exakteren Konfigurierbarkeit auch unempfindlicher gegenüber Störungen.

Im Gegensatz zu vollwertigen ReacTIVision-Markern, die ein Token zu jedem Zeitpunkt eindeutig identifizieren, können mehrere gleichzeitig eingesetzte Cursor grundsätzlich nicht unterschieden werden (da sie alle den gleichen Code aufweisen). ReacTIVision löst dieses Problem, indem für Cursor eine nach der Reihenfolge des Auftretens vergebene eindeutige Session-ID zugeordnet wird, der einen Cursor bis zum Zeitpunkt des Verschwindens eindeutig identifiziert (diese Session-ID wird im übrigen auch vollwertigen Markern zugeordnet, hat aber in der hier vorgestellten Anwendung keine Bedeutung). Potentiell problematisch ist die Zuordnung einer Session-ID zu einem Cursor dann, wenn die Beleuchtungsbedingungen schlecht sind, so dass das Cursor-Token nicht durchgängig erkannt wird. In diesem Fall wird bei jeder (Wieder)-Erkennung des betreffenden Tokens eine neue Session-ID zugeordnet, was gleichbedeutend mit dem Auftreten eines neuen Cursors ist. Zur Compensation können die bereits beschriebenen Mechanismen eingesetzt werden, vor allem das Ignorieren von zeitnah nach dem Verschwinden eines Cursors auftretenden neuen Cursorn an exakt der gleichen Position verhindert derartige Fehlerkennungen weitgehend.

Wird ein Cursor-Token auf der Oberfläche aufgesetzt und von ReacTIVision erkannt, werden die in Abschnitt 6.4.2 beschriebenen Prozesse zur Stabilisierung der Erkennungsleistung ausgeführt. Sobald der Cursor entsprechend lange stabil auf der Oberfläche erkannt wird, so dass eine Fehlerkennung weitgehend ausgeschlossen werden kann, wird im Umkreis der Cursorposition (Radius etwa 5 Zentimeter) nach aktuell auf der Oberfläche vorhandenen Modellierungs-Tokens gesucht. Das dem Cursor am nächsten liegende Token wird ausgewählt (insofern es innerhalb der Grenze von 5 Zentimetern liegt). Den Benutzern wird diese Auswahl über die Ausgabekanäle rückgemeldet. Damit ist der Markierungsvorgang grundsätzlich abgeschlossen.

In der Verarbeitung von Cursor sind noch zwei Spezialfälle zu betrachten, die im Kontext der Herstellung von Verbindungen auftreten. Im ersten Fall muss das Auftreten von zwei Cursoren erkannt werden, die die zu verbindenden Modellelemente markieren. Eine Verbindung wird dann hergestellt, wenn die beiden Cursor innerhalb von 3 Sekunden jeweils in der Nähe eines Modellierungs-Tokens erkannt werden. Die grundlegende Erkennung einer Markierung läuft für beide Cursor wie im letzten Absatz beschrieben ab. Zusätzlich wird nach jeder erkannten Markierung geprüft, ob auf der Oberfläche eine weitere aktive Markierung vorhanden ist (also eine, die in den letzten drei Sekunden gesetzt wurde). Ist dies der Fall und wird die andere Markierung nicht bereits für andere Zwecke (z.B. einen gerade laufenden Benennungsvorgang) verwendet, werden beide Markierungen gelöscht und anstelle derer eine Verbindung zwischen den markierten Modellelementen eingefügt. Der

zweite Spezialfall ist die Erkennung von gerichteten Verbindungen, also das Auftreten von Werkzeug-Tokens, die einen Verbindungsendpunkt mit Pfeilspitze kennzeichnen. Wie in Abschnitt 6.2.2 beschrieben, unterscheiden sich diese Tokens von den anderen Markierungstokens durch einen zweiten Cursor-Marker, der unmittelbar neben dem ersten angebracht ist. Tritt nun ein zweiter Cursor auf der Oberfläche auf, der sich nahe eines bereits erkannten Cursors befindet (die Benachrichtigung über Erkennung erfolgt ereignisgesteuert und damit sequentiell, auch wenn beide Cursor-Marker gleichzeitig auf der Oberfläche auftreten), muss die bereits erstellte Markierung für das nächstliegende Modellierungs-Token semantisch umgedeutet werden und von einem Endpunkt einer ungerichteten Verbindung auf den einer gerichteten Verbindung konvertiert werden.

Softwareseitig erfolgen die betreffenden Prüfungen im jenem Algorithmus, der auch die Herstellung von Verbindungen prüft. Wenn also ein Cursor auf der Oberfläche auftaucht und gleichzeitig schon ein weiterer aktiver Cursor vorhanden ist, wird neben den oben beschriebenen Prüfungen zur Verbindungsherstellung auch eine Prüfung auf Nähe zu bereits vorhandenen Cursoren durchgeführt und ggf. die Markierung des betroffenen Modellelements umgewandelt. Diese Veränderung der Markierung wird auch den Benutzern über die Ausgabekanäle visuell kommuniziert.

6.4.4. Erkennung von geöffneten Tokens

In den Abschnitten 6.2.2 und 6.3.5 wurde bereits die Erkennung des Öffnungszustandes eines Modellierungs-Tokens konzeptionell beschrieben. Diese wird durch das Kamerasytem und ReacTIVision festgestellt. Dazu ist auf der beweglichen Rückwand der Modellierungs-Tokens ein ReacTIVision-Code angebracht, der beim Öffnen auf der Modellierungsoberfläche zu liegen kommt und dadurch von der Kamera erfasst wird. Technisch sind dazu eine strukturelle Maßnahme und die Implementierung einer entsprechenden Interpretationsroutine notwendig.

Die strukturelle Maßnahme betrifft die Erkennung des auf der Rückwand angebrachten ReacTIVision-Codes. Aufgrund der geringen Größe der Rückwand (10 cm x 4 cm) kann kein herkömmlicher ReacTIVision-Code in ausreichender Größe angebracht werden. ReacTIVision erlaubt jedoch wie in Abschnitt 6.1.2 beschrieben die Erstellung von eigenen Codes beliebiger Form. Die abgebildete Code-Topologie muss lediglich im Mapping-File vermerkt werden, um eine Abbildung auf einen eindeutigen Identifikationsnummer zu ermöglichen. Wie in Abbildung 6.9 zu erkennen ist, wurde diese Möglichkeit im vorliegenden Fall genutzt. Auf den Rückwänden der Modellierungs-Tokens ist ein einfacher, langer Code angebracht, dessen Identifikationsnummer im Interpretationsmodul die Routinen zur Beeinflussung des Öffnungszustands eines Tokens aufruft.

6. Eingabe und Interpretation

Im Interpretationsmodul wird beim Eingehen der Meldung über das Auftauchen eines entsprechenden Markers dessen Position in Relation mit den Positionen der auf der Modellierungsoberfläche vorhandenen Modellierungs-Tokens gesetzt. Da auf allen Tokens die gleichen Marker zur Identifikation des Öffnungszustands angebracht sind, muss das betroffene Token über die Nähe zum erkannten Marker identifiziert werden. Befindet sich also der Marker eines Modellierungstokens in unmittelbarer Nähe zum aufgetauchten Token (Radius etwa 2,5 cm), so wird dessen Zustand auf „offen“ gesetzt. Befinden sich mehrere Modellierungs-Tokens in diesem Radius, was bei sehr eng gesetzten Modellen möglich ist, so wird das nächstliegende ausgewählt (außer in einigen seltenen Ausnahmekonstellationen ist das jenes Token, zu dem der erkannte Öffnungs-Marker gehört). In der Weiterverarbeitung wird das das betreffende Modellelemente repräsentierende Objekt (im objektorientierten Sinn) über die Änderung seines Öffnungszustandes benachrichtigt. Dieses wechselt seinen Zustand und löst visuelles Feedback über die Ausgabekanäle aus. Außerdem reagiert es nun auf Anfragen bezüglich der Einbettung von Zusatzinformation bzw. deren Abruf.

Sobald das Öffnungs-Token wieder von der Oberfläche verschwindet (das zugehörige Modellierungs-Token also geschlossen wurde), wird im Umkreis des verschwundenen Markers nach nach geöffneten Modellierungs-Tokens gesucht. Das am nächsten liegende Element wird geschlossen, was wiederum durch eine entsprechende Benachrichtigung des das Modellelement repräsentierende Objekt realisiert wird. Grundsätzlich könnte eine Zuordnung zwischen Öffnungs-Marker und Modellierungs-Token auch über die Session-ID des Öffnungs-Markers durchgeführt werden, die im Öffnungsvorgang dem entsprechenden Modellierungs-Token zugeordnet werden könnte. Von dieser Vorgehensweise wurde jedoch Abstand genommen, da im Falle von Fehlerkennungen des Öffnungs-Markers (kurzfristige Ausfälle, die zur Zuweisung einer anderen Session-ID führen) zu Zuordnung nicht mehr korrekt vorgenommen werden kann. Deshalb wurde ein zustandsloser („state-less“) Algorithmus eingesetzt, der durch die erneut durchgeführte Suche nach dem betroffenen Modellierungs-Token nicht von einer früher gespeicherten Session-ID abhängig ist.

Einbetten und Abrufen von zusätzlicher Information

Im Zusammenhang mit der Erkennung von geöffneten Tokens kann auch die softwareseitige Behandlung von eingebetteter Information beschrieben werden. Diese wurde konzeptionell bereits in Abschnitt 6.3.5 beschrieben. Anhand der nun folgenden Beschreibung aus software-technischer Sicht kann nun auch die Einbindung der Registrierungs-Kamera beschrieben werden.

Die Registrierungskamera dient in diesem Kontext der Erkennung von einbettbaren Tokens, wobei bei der Erkennung drei Fälle unterschieden werden müssen (in Klammer jeweils die Bedingungen, die zur Auswahl eines der Fälle führen):

1. Information anbinden (noch keine Information angebunden)
2. Token einbetten (Information angebunden, Token noch nicht eingebettet, Container-Token geöffnet)
3. Information abrufen (Information angebunden, Token noch nicht eingebettet oder Token eingebettet und Container-Token geöffnet)

Technisch können Fall 1 und 2 auch gleichzeitig auftreten, wenn Information an ein Token gebunden wird, während gleichzeitig ein Container-Token geöffnet wird. In diesem Fall erfolgt die Einbettung unmittelbar nach dem Anbinden der Information.

Software-seitig beginnt der Prozess der Behandlung eines einbettbaren Tokens immer gleich mit dem Schritt der Erfassung des einbettbaren Tokens durch die Registrierungskamera. Diese ist konzeptionell als weiterer Eingabekanal zu sehen, die wie die Hauptkamera Daten an das Interpretationsmodul liefert (siehe Abbildung 6.18), die dieses kontextsensitiv (also abhängig vom aktuellen Modellzustand) auswerten muss. Das Signal der Registrierungskamera wird von einer zweiten ReacTIVision-Instanz aufgenommen und ausgewertet. Diese ist mittels eines Brückenobjekts an das Interpretationsmodul angebunden. Dieses Brückenobjekt nimmt eine Vorauswertung statt und liefert bereits höherwertige Information an das Interpretationsmodul. So ist jegliche Benutzerinteraktion, die nicht unmittelbar Information über den aktuellen Modellzustand benötigt, in diesem Objekt gekapselt (z.B. das Anbinden digitaler Information an ein einbettbares Token). Interaktion, die verteilt über die Registrierungskamera und die Tischoberfläche abläuft, wird an definierten Schnittstellen an das die zuständigen Komponenten im Interpretationsmodul übergeben.

Im konkreten Fall bedeutet dies, dass nach der Erfassung des Tokens durch die Registrierungskamera die oben getroffene Fallunterscheidung wirksam wird, die – wie eben angedeutet – je nach Art des einbettbaren Tokens noch zusätzliche Unterscheidungen erfährt. Bei Tokens an die noch keine Information gebunden wurde, wird diese zusätzliche Unterscheidung wirksam. Bei blauen einbettbaren Tokens wird eine Auswahlbox geöffnet, mittels der eine anzubindende digitale Ressource im lokalen Dateisystem ausgewählt werden kann. Ein gelbes Token löst die Aufnahme eines Fotos aus. Dieses wird wiederum im lokalen Dateisystem gespeichert und an das Token gebunden. Diese beiden Arten werden vollständig im Brückenobjekt abgehandelt, den übrigen Komponenten wird lediglich ein Datenobjekt übergeben, das der Weiterverarbeitung (also z.B. der eigentlichen Einbettung) dient. Die Behandlung eines roten Tokens kann nicht im Brückenobjekt verarbeitet werden, da

es der Speicherung von Submodellen dient und dementsprechend auf Information über den aktuellen Modellzustand auf der Modellierungsoberfläche angewiesen ist. Ein rotes Token löst deshalb einen Modellerfassungsvorgang wie in Abschnitt 6.4.7 weiter unten beschrieben aus. Das erfasste Submodell wird danach an das rote Token gebunden.

Ist bereits Information an ein Token gebunden, kann die Erfassung durch die Registierungskamera der Startpunkt für die Fälle 2 oder 3 sein. Fall 2 tritt dann ein, wenn das einbettbare Token noch keinem Container zugeordnet ist und ein Container geöffnet ist. In diesem Fall wird das Objekt, das die einzubettende Information repräsentiert, an das Modell-Objekt des Container-Tokens übergeben und ab diesem Zeitpunkt von diesem verwaltet. Die erfolgreiche Zuordnung wird über die Ausgabekanäle visuell rückgemeldet, das Token kann in der Folge auch physisch eingebettet werden (siehe Abschnitt 6.3.5). Fall 3 tritt ein, wenn entweder kein Container geöffnet ist oder das betreffende Token bereits einem Container zugeordnet ist. Hier tritt wiederum die Unterscheidung nach Art des einbettbaren Tokens in Kraft. Bei roten Token wird der gespeicherte Modellzustand über die Ausgabekanäle dargestellt. Bei gelben oder blauen Tokens wird die angebundene digitale Ressource – wenn möglich – mittels der Standardapplikation des Betriebssystems, die dem betreffenden Dateitypen zugeordnet ist, geöffnet und dargestellt.

6.4.5. Benennung von Modellelementen

Abschnitt 6.3.2 beschreibt die zur Benennung von Modellelementen und Verbindungen notwendige Interaktion. Wie dort angegeben, existieren zwei Eingabemodalitäten für diese Funktionalität, die sich in ihrer technischen Umsetzung stark unterscheiden. Gemeinsam ist beiden, dass sie die angegebene Benennung dem jeweils markierten Element zuweisen. Ist aktuell kein Modellierungs-Token markiert, wird die Benennung dem zuletzt hinzugefügten Token zugeordnet. Die Verwaltung der Benennung obliegt dem Modellobjekt – sie wird jedoch auch im Interpretationsmodul zwischengespeichert, um die letzte Benennung eines zwischenzeitlich von der Modellierungsoberfläche entfernten Objekts (dessen digitale Repräsentation bereits entfernt wurde) wiederherstellen zu können. Dies ist notwendig, um die Wiederherstellung von eingebetteten Submodellen gewährleisten zu können, dessen Modellierungstokens sich während der Modellerstellung auf einer anderen Ebene nicht auf der Oberfläche befinden.

Benennung mittels Tastatur

Die Benennung mittels Tastatur ist die einzige Interaktion mit den Benutzern, die die Verwendung eines herkömmlichen Eingabemediums bedingt. Um die Eingabe

trotz des „Medienbruchs“ möglichst nahtlos zu gestalten, wurde die Möglichkeit geschaffen, nach der Auswahl eines Modellierungs-Tokens ohne weitere Interaktion über die graphische Benutzungsschnittstelle des Systems mit der Eingabe beginnen zu können. Mit dem ersten Tastendruck öffnet sich am Bildschirm eine Eingabemaske, mittels der die Benennung durchgeführt werden kann. Eine Bestätigung mit der Eingabe-Taste schließt die Eingabemaske und weist die Benennung dem ausgewählten Element zu. Beginnt ein Benutzer ohne vorhergehender Auswahl mit der Informationseingabe, so wird diese dem zuletzt hinzugefügten Element (Modellelement oder Verbinder) zugewiesen.

Technisch ist die Eingabe mittels einem KeyListener implementiert, der das Beobachtermuster (Gamma et al., 1995) in Java in Bezug auf Tastatureingaben benachrichtigt. Das Interpretationsmodul wird also über jeden Tastendruck informiert und muss über dessen Auswirkungen entscheiden. Ist der Tastendruck nicht dem Systemkonfigurationsmodus (siehe Abschnitt 7.4.1) zuzuordnen, wird die Eingabemaske aktiviert, die dann alle weiteren Tastatureingaben abfängt, bis sie wieder geschlossen wird. Die Eingabe wird in der Folge an die zuständigen Komponenten bzw. das betroffene Modell-Objekt zur Weiterverarbeitung übergeben.

Benennung mittels Haftnotiz

Die Benennung mittels Haftnotiz ermöglicht die handschriftliche Benennung eines Modellelements oder Verbinders. Technisch wird sie über die Registrierungskamera abgewickelt. Die Haftnotiz muss dazu beschriftet werden und in der Folge mittels dem Registrierungstoken (siehe Abschnitt 6.2.2) der Registrierungskamera zur Verfügung gestellt. Die Erkennung des Registrierungstokens löst die Aufnahme eines Bildes aus, aus dem in weiterer Folge die Beschriftung extrahiert wird. Dazu sind am Registrierungstoken zwei ReacTIVision-Marker angebracht, die an definierten Positionen links und rechts von der Haftnotiz sitzen. Durch diese beiden bekannten Position ist eine Ausrichtung des Bildes möglich, so dass sich die Haftnotiz an einer definierten Position befindet, an der sie schließlich ausgeschnitten wird. Dazu ist es notwendig, das Bild soweit zu rotieren, dass sich die beiden Marker auf einer horizontalen Achse befinden und der physisch linke Marker links im Bild dargestellt wird. Danach wird das Bild soweit skaliert, bis der Abstand der beiden Marker einem vorab definierten Wert entspricht. Damit sind die Ausmaße der Haftnotiz in horizontaler und vertikaler Richtung bekannt. Wird das Bild nun soweit verschoben, dass sich die beiden Marker an vordefinierten Positionen befinden, kann auch die Haftnotiz im Bild lokalisiert werden. Durch einen Ausschneidevorgang wird die Haftnotiz extrahiert. Das Bild wird nun mit einem adaptiven Algorithmus in eine Schwarz-Weiß-Grafik umgewandelt. Der Algorithmus identifiziert dazu die hellsten Bereiche (die Haftnotiz) und die dunkelsten Bereiche (der Schriftzug) im Bild. Die

hellsten Bereiche werden weiß, die dunkelsten Bereiche werden schwarz gesetzt. Die dazwischenliegenden Werte mittels einem Schwellwert der bei $2/3$ der Differenz zwischen hellstem und dunkelstem Wert liegt, auf weiß oder schwarz gesetzt. Das Resultat wird den weiterverarbeitenden Komponenten als Pixelgrafik zur Verfügung gestellt (wie bei der Benennung mittels Tastatur). Durch die Bildverarbeitung ist die Position des Registrierungstokens im Bild der Registrierungskamera irrelevant, da diese weitgehend kompensiert werden können. Durch die adaptive Umrechnung in eine Schwarz-Weiß-Grafik kann diese Art der Benennung auch bei schlechten Beleuchtungsbedingungen eingesetzt werden.

6.4.6. Festlegung der Bedeutung von Modellelementen

Durch die Möglichkeit der Festlegung der Bedeutung eines bestimmten Modelltyps (siehe Abschnitt 6.3.1) wird die Forderung nach semantisch flexibler Modellierung umgesetzt. Die Arten von Modellelementen sind semantisch nicht vorbelegt sondern werden während des Modellierungsvorgangs spezifiziert. Die Frage nach der Angabe des Bedeutungstyps wird ausgelöst, wenn eine Tokenart erstmals auf der Oberfläche erkannt wird. Über eine Eingabemaske und die Tastatur kann dann die Bedeutung textuell eingegeben werden. Wird dieser Vorgang von den Benutzern abgebrochen, besteht die Möglichkeit die aktive Nachfrage bezüglich der Bedeutungszuordnung für den laufenden Modellierungsvorgang auszuschalten, um im Erstellungsprozess nicht unterbrochen zu werden. Die Bedeutungszuordnung kann aber immer ausgelöst werden, wenn ein Token in den Erfassungsbereich der Registrierungskamera gehalten wird. Das System fragt dann nach der Zuordnung der Bedeutung zum jeweiligen Elementtypen.

Auf den Modellierungsvorgang selbst hat die Festlegung der Bedeutung technisch keine Auswirkungen. Für die Persistierung des Modells ist sie jedoch von Bedeutung, da neben dem eigentlichen Modell auch das Metamodell (und damit die Arten von verwendeten Modellelementen und deren Bedeutung) gespeichert werden (siehe Kapitel XY).

6.4.7. Tracking des Modellzustandes

Das Tracking des Modellzustandes läuft von den Benutzern unbemerkt immer im Hintergrund, kann jedoch auch explizit mit dem Snapshot-Token ausgelöst werden. Auch die Speicherung von Submodellen auf einbettbare Token greift auf die gleichen Routinen zurück. Die Speicherung von stabilen Modellzuständen basiert auf der Verfolgung des Zeitpunkts der letzten Änderung im Modell. Dazu wird ein Zeitsstempel mitgeführt, der bei jeder Modelländerung aktualisiert, also auf die aktuelle

Systemzeit gesetzt wird. Änderungen sind das Platzieren, Verschieben oder Entfernen von Modellierungs-Tokens, das Herstellen oder Löschen einer Verbindung, das Benennen eines Modellelements oder einer Verbindung und das Einbetten von Zusatzinformation. In regelmäßigen Abständen wird (durch einen Timer ausgelöst) geprüft, wieviel Zeit seit der letzten Modelländerung vergangen ist. Sind mehr als fünf Sekunden vergangen, wird der aktuelle Systemzustand gespeichert und die Prüfung gestoppt, bis die nächste Änderung auftritt. Damit wird verhindert, dass in von langen Phasen ohne Änderung mehr als eine Aufnahme angefertigt wird.

Bei der Speicherung des Modellzustandes (egal ob automatisiert oder explizit ausgelöst) wird eine Kopie der digitalen Modellrepräsentation angefertigt. Diese beinhaltet im Sinne einer „Deep Copy“ Kopien aller Informationen die im Modell abgelegt sind, so dass keine Referenzen mehr auf das aktuelle Modell zeigen. Damit wird sichergestellt, dass laufende Änderungen am Modell keine Auswirkungen auf den gesicherten Zustand haben. Technisch wurde dies gelöst, indem die „clone“-Methode aller datenträgenden Objekte (Modellobjekte und Verbindungsobjekte) so überladen wurden, dass sämtliche referenzierten Objekte (etwa Benennungen, eingebettete Objekte,...) dupliziert werden und im kopierten Objekt Referenzen auf die erstellten Duplikate eingefügt werden. Dieses Vorgehen ist bei großen Modellen und langer Modellierungszeit durchaus speicherintensiv und bedarf einer ausreichenden Größe des Heaps (der ggf. durch Kommandozeilen-Parameter beim Start des Systems angepasst werden muss).

Navigation in der Modellierungshistorie

Die Navigation in der Modellierungshistorie wird mittels des in den Abschnitten 6.2.2 und 6.3.6 beschriebenen runden Tokens durchgeführt. Die Erkennung der Rotation erfolgt durch den von ReacTIVision übergebenen Rotationswert des Tokens, der in Radian geliefert wird. Das Interpretationsmodul schaltet dementsprechend bei jeweils bei Vielfachen von $\frac{\pi}{4}$ (also alle 45°) einen gespeicherten Schritt weiter oder zurück (je nach Drehrichtung).

Zur Verwaltung der Modellierungshistorie wird ein Objekt benutzt, das alle gespeicherten Zustände in der Reihenfolge ihrer Speicherung referenziert, den aktuell ausgewählten Zustand speichert und den Abruf von sowie die Navigation durch die gespeicherten Zustände ermöglicht. Dieses Objekt wird durch das Interpretationsmodul kontrolliert und gesteuert – für übergeordnete Software-Module ist das Umschalten zwischen dem aktuellen Modellzustand und einem gespeicherten Zustand transparent. Das Format, in dem die gespeicherten Zustände ausgeliefert werden, entspricht jenem in dem auch die aktuellen Modell-Daten verwaltet werden – dadurch wird auch das Umschalten auf einen gespeicherten Zustand im Zuge einer Wiederherstellung einer vergangenen Modellversion erleichtert.

Wiederherstellung eines Modellzustandes

Der Ablauf der Wiederherstellungsunterstützung wird – wie in Abschnitt 6.3.6 bereits erwähnt – erst in Kapitel XY im Detail beschrieben, da er vorrangig die Ausgabekanäle betrifft. Da die Modellierungshistorie konzeptionell jedoch im Interpretationsmodul angesiedelt ist und der Vorgang der Wiederherstellung auch durch dieses ausgelöst wird (nach Erkennung des entsprechenden Tokens), werden die technischen Details hier besprochen.

Im Zuge der Wiederherstellungsunterstützung müssen die Unterschiede zwischen dem aktuellen Modell und wiederherzustellenden gespeicherten Zustand erkannt werden, um die Benutzer schrittweise bei der Wiederherstellung anleiten zu können. Da nur die physischen Bausteine durch den Benutzer manipuliert werden müssen, beschränkt sich die Differenzbildung zwischen den Modellen auf diese. Rein digitale Information – wie die Verbindungen – können am Ende des Wiederherstellungsvorgangs geladen und über die Ausgabekanäle dargestellt werden. Bei den physischen Bausteinen, also im wesentlichen den Modellierungs-Tokens sind im ersten Schritt drei Fälle zu unterscheiden:

- Tokens, die im aktuellen Modell enthalten sind und im wiederherzustellenden Modell nicht vorhanden waren
- Tokens, die sich im aktuellen Modell an einer anderen Position befinden als im wiederherzustellenden Modell
- Tokens, die im aktuellen Modell nicht vorhanden sind, im wiederherzustellenden Modell aber vorhanden waren

Im ersten und dritten Fall reicht für die Erkennung eine Differenzbildung zwischen den beiden Modellobjekt-Mengen aus, wobei im ersten Fall die Menge der wiederherzustellenden Modellobjekte von der aktuellen Menge der Modellobjekt abgezogen werden muss um die zu entfernenden Tokens zu identifizieren, im dritten Fall umgekehrt die Menge der aktuellen Modellobjekte von der Menge der wiederherzustellenden Modellobjekte abgezogen wird um die hinzufügenden Tokens zu identifizieren. Im zweiten Fall muss für jedes Token dessen aktuelle und gespeicherte Position verglichen werden, um die zu verschiebenden Tokens zu identifizieren. Zusätzlich muss noch der Inhalt der Tokens (also deren eingebetteten Tokens) abgeglichen werden, um auch hier Differenzen identifizieren zu können.

Zur Durchführung der Wiederherstellungsunterstützung wird das System in einen speziellen Zustand geschaltet, in dem es aufgabengesteuert arbeitet. Jede vorzunehmende Änderung im Modell wird auf eine Aufgabe abgebildet. Eine Aufgabe ist ein spezielles Objekt, das neben der vorzunehmenden Änderung auch eine Methode enthält, die prüft, ob die Aufgabe erfolgreich abgeschlossen wurde (wobei die Bedingungen von der konkreten Aufgabe abhängen). Befindet sich das System im Wieder-

herstellungsmodus, wird bei jeder von ReacTIVision gemeldeten Änderung auf der Modellierungsoberfläche geprüft, ob mit dieser Änderung die Aufgabe erfüllt wurde. Ist dies nicht der Fall, bleibt die Aufgabe aktiv. Wurde die Aufgabe erfüllt, wird durch einen erneuten Vergleich zwischen Ist- und Soll-Zustand die nächste Aufgabe identifiziert und aktiv gesetzt. Dazu wird entsprechendes Feedback auf den Ausgabekanälen ausgegeben. Sobald Ist- und Soll-Zustand gleich sind, werden die rein digitalen Teile des Modells geladen, womit das aktuelle Modell dem gespeicherten Zustand vollständig entspricht.

6.4.8. Verteilung des Modellzustandes

Die Verteilung der darzustellenden Modellinformation an die übergeordneten Softwaremodule ist Gegenstand des letzten Abschnitts in diesem Kapitel. Die interpretierten und aggregierten Informationen werden jenen Softwaremodulen zur Verfügung gestellt, die bisher in ihrer Gesamtheit als „Ausgabekanäle“ bezeichnet wurden. Diese Ausgabekanäle sind in der vorgestellten Anwendung mehrere grafische Oberflächen, die den Systemzustand visuell wiedergeben (siehe dazu Details in Kapitel 7).

Wichtig im Kontext dieses Kapitels ist dabei, dass die Ausgabekanäle frei definierbar und erweiterbar sein sollen. Um dieser Anforderung auch vor dem Einsatz eines Frameworks wie TUIpist gerecht zu werden, wurde eine Verteilungskomponente implementiert, die auf dem bereits oben mehrmals erwähnten Besuchermuster von Gamma et al. (1995) basiert. Dabei registrieren sich Ausgabekanäle beim Interpretationsmodul und werden danach über eine definierte Schnittstelle mit der entsprechenden Modellinformation versorgt. Die Schnittstelle ist Teil des Interpretationsmoduls und muss von jedem Ausgabekanal vollständig implementiert werden, auch wenn dieser die betreffende Information nicht benötigt (in diesem Fall bleibt die ausgabeseitig aufgerufene Methode leer).

6.5. Zusammenfassung

In diesem Kapitel wurde jener Teil des Werkzeugs im Detail beschrieben, der sich mit der Erfassung und der Interpretation der Benutzereingaben beschäftigt. Im ersten Teil wurden in Frage kommende technologische Ansätze zur Erfassung der Benutzeraktivität betrachtet und einander gegenübergestellt. Nach der Entscheidung für den Einsatz eines optischen Ansatzes wurden für diesen Zweck geeignete Frameworks betrachtet. Dabei wurden Framework zur optischen Identifikation von physischen Tokens sowie generische Frameworks, die als Middleware für Tabletop Interfaces geeignet sind, beschrieben und verglichen. Die Entscheidung fiel hier zugun-

6. Eingabe und Interpretation

ten ReacTIVision (Kaltenbrunner und Bencina, 2007) als optisches Erkennungs-framework und TUIpist (Furtmüller und Oppl, 2007) als Middleware.

Im zweiten Teil wurde die Umsetzung des Hardwarekomponenten beschrieben, die maßgeblich von den zuvor getroffenen Technologie-Entscheidungen abhängig ist. Dabei wurde die Infrastruktur des Werkzeugs beschrieben (der „Tisch“), wobei in diesem Kapitel der Fokus auf jenen Komponenten lag, die die Erfassung von Benutzerinteraktion dienen. Zusätzlich wurden die eigentlichen Tokens, mit denen die Benutzer mit dem System interagieren, beschrieben. Hier ist zwischen Modellierungs-Tokens und Werkzeug-Tokens zu unterscheiden, wobei erstere der eigentlichen Modellbildung dienen und zweitere das Systemverhalten kontrollieren.

Die eigentliche Interaktion der Benutzer mit dem System ist Gegenstand des dritten Teils. Hier wurde die Verwendung der Tokens bei der Modellbildung und zur Steuerung des Systems skizziert und damit jene Interaktionsabläufe definiert, die vom System erkannt und korrekt interpretiert werden müssen. Die Unterabschnitte dieses Teils entsprechen im Wesentlichen den Anforderungen, die die das Werkzeug erfüllen muss, und beschreiben deren Umsetzung.

Nach der Definition der Interaktionsabläufe konnte im vierten Teil die software-technische Behandlung der durch das ReacTIVision-Framework erkannten Eingabedaten beschrieben werden. Die ersten beiden Abschnitt behandeln allgemein die Systemarchitektur und der Kompensation eventueller Erkennungsprobleme der Tokens auf der Modellierungsoberfläche. Zweiteres ist notwendig, weil optische Frameworks generell sensibel auf Änderung der Beleuchtungsbedingungen reagieren und es dementsprechend zu (temporären) Fehlfunktionen bei der Erkennung kommen kann. Dazu wurden die möglichen Fehlerquellen aufgelistet und die umgesetzten sowie zusätzlich mögliche Kompensationsansätze beschrieben. In den übrigen Abschnitten wird die software-technische Erkennung und Behandlung der im vorhergehenden Teil definierten Interaktionsabläufe beschrieben. Der Fokus lag hier auf den konzeptionellen Implementierungsdetails und der Beschreibung der Lösung spezifischer Herausforderungen, die durch die gewählte Technologiekonstellation auftreten. Im letzten Abschnitt wurde als Brücke zu den weiteren Kapiteln jene Komponente beschrieben, die für die Verteilung der Information an die weiterverarbeitenden Software-Module zuständig ist.

7. Ausgabe

In diesem Kapitel wird die konzeptionelle Ausrichtung und technische Umsetzung jenes Teils des Werkzeugs behandelt, der sich mit der Ausgabe von Information an die Benutzer beschäftigt. Bei Tangible Interfaces erfolgt die Ausgabe von Information zumeist kohärent mit dem Eingabemedium, eine physische Trennung zwischen Eingabe- und Ausgabekanälen wie in der herkömmlichen Mensch-Maschine-Interaktion liegt nicht vor (Ullmer und Ishii, 2000). Fishkin (2004) relativiert die strikte Forderung nach Kohärenz in seiner Taxonomie für Tangible Interfaces (wie in Abschnitt 5.2.11 beschrieben und klassifiziert Benutzungsschnittstellen unter anderem nach dem Grad der Kohärenz von Ein- und Ausgabe. Dementsprechend sind nicht nur jene Ausgabekanäle Gegenstand dieses Kapitels, die Information in direkter Verbindung mit den Eingabemedien zurückspiegeln, sondern auch jene, die Information auf anderen, nicht-kohärenten Wegen ausgeben.

Im ersten Abschnitt dieses Kapitels werden auf Basis der in Kapitel XY genannten Anforderung an das Werkzeug die den Benutzern mitzuteilenden Informationen identifiziert, noch ohne konkret auf die technologische Realisierung der Ausgabekanäle einzugehen. Im darauf folgenden Abschnitt werden die technischen Möglichkeiten zur Ausgabe von Information betrachtet. Im Anschluss werden diese Möglichkeiten hinsichtlich ihrer Eignung für die im konkreten Anwendungsfall auszugebende Information bewertet und entsprechend zugeordnet.

Auf Basis der grundsätzlichen Technologieentscheidung werden in der Folge Software-Frameworks beschrieben, die die Realisierung der gewählten Ausgabekanäle ermöglichen. Die Entscheidung für ein konkretes Framework wird auf Basis der funktionalen und nicht-funktionalen Anforderungen an die Ausgabe und deren Umsetzung getroffen. Der letzte Abschnitt beschreibt die eigentliche Umsetzung der Ausgabekanäle mittels der gewählten Technologie und geht die spezifischen Eigenschaften und Implementierungsentscheidungen der vorgestellten Lösung ein.

7.1. Auszugebende Information

Den Benutzern des Systems müssen während der Modellierung unterschiedliche Information zur Verfügung gestellt werden. Einerseits ist dies Information, die das

Modell selbst betrifft, andererseits muss auch Information ausgegeben werden, die Aspekte des Modellierungsablaufs beschreibt oder unterstützt.

Die das Modell betreffende Information umfasst folgende Aspekte:

- Die Modellelemente betreffende Information (Art, Position, Benennung)
- Die Verbinder betreffende Information (Art, Endpunkte, Benennung)
- Eingebettete Elemente betreffende Information (Art, Inhalt, Container)

Zur Unterstützung des Modellierungsablaufs müssen folgende Aspekte zur Verfügung gestellt werden:

- Information über vergangene Modellzustände
- Information zur Wiederherstellung von Modellzuständen

Hier wird bewusst noch nicht auf die technische Umsetzung dieser Ausgabe eingegangen. In den folgenden Abschnitten wird erörtert, welche grundlegenden Technologien in Frage kommen, bevor auf Basis deren Eignung und den Vorgaben aus den Technologieentscheidungen zur Informationseingabe eine konkrete Lösung ausgewählt wird.

7.2. Technologische Grundlage der Ausgabe

Bei der Ausgabe von Information muss im Falle von Tangible Interfaces zwischen Ansätzen mit unterschiedlich stark ausgeprägter Kohärenz mit den Eingabekanälen unterschieden werden. Unter Kohärenz ist hier zu verstehen, das jene Artefakte, die zur Eingabe verwendet werden gleichzeitig auch die Reaktion des Systems – also die Ausgabe – wiederspiegeln. In der von Fishkin (2004) vorgeschlagenen Taxonomie (siehe Abschnitt 5.2.11) werden in der Dimension „Embodiment“ auch Werkzeuge als Tangible Interfaces klassifiziert, bei denen die Ausgabe vollständig von der Eingabe entkoppelt ist. Diesem Verständnis folgt auch diese Arbeit.

Bei Tabletop Interfaces bietet sich die Tischoberfläche als Ausgabemedium an, um kohärente Informationsausgabe zu gewährleisten. Die Tischoberfläche dient hier wie in Kapitel 6 beschrieben der Eingabe und kann durch unterschiedliche technologische Maßnahme auch zur Ausgabe genutzt werden. Eine weitere Möglichkeit, die Ausgabekohärenz bei Tabletop Interfaces sicherzustellen bzw. zu steigern, ist die Verwendung der zur Interaktion mit dem System verwendeten Tokens als Ausgabemedium. Je nach verfolgtem Ansatz (bzw. einer Kombination) sind unterschiedliche technische Maßnahmen zu setzen. In den folgenden Abschnitten werden die erwähnten grundsätzlich in Frage kommenden Ansätze betrachtet und im Anschluss hinsichtlich ihrer Eignung für das hier entwickelte System beurteilt. Basierend auf

der grundsätzlichen Technologieentscheidung werden im Anschluss unterschiedliche technische Lösungen zur Erfüllung der Anforderungen beschrieben.

7.2.1. Ansätze zur kohärenten Ausgabe

In diesem Abschnitt werden Ausgabeansätze behandelt, die nach (Fishkin, 2004) in der Embodiment-Dimension den Ausprägungen „full“ oder „nearby“ zuzuordnen sind. Die Ausgabe erfolgt bei den hier vorgestellten Ansätzen also direkt über die Eingabetokens („full“) oder ist räumlich unmittelbar in der Umgebung der Tokens angesiedelt.

Darstellung auf der Tischoberfläche

Bei Tabletop-Interfaces ist die Nutzung der Tischoberfläche ein naheliegender und gängiger Ansatz zur Realisierung der Ausgabekanäle. Die Ausgabe erfolgt hierbei vorrangig visuell, also durch die Darstellung der auszugebenden Information. Ein derart ausgestaltetes Interface ist hinsichtlich seiner Ausprägung in der Embodiment-Dimension als „nearby“ zu klassifizieren. Technologisch kommen zur Darstellung horizontal eingesetzte Bildschirme oder Oberflächen, auf die projiziert werden kann, in Frage.

Bei der Verwendung von Bildschirmen sind die Größe der zur Anzeige verwendbaren Oberfläche sowie die zur Anzeige verfügbare Auflösung (also indirekt die Größe eines Bildpunktes) wesentliche Kriterien. Bei heute verfügbaren LCD¹-Modulen mit Größen bis zu 132 cm in der Diagonale und Auflösungen von 1920 x 1080 Bildpunkten ist die Technologie soweit ausgereift und verfügbar, das dieser Ansatz grundsätzlich für den Einsatz in Tabletop Interfaces in Frage kommt. Vorteile sind die geringe Bauhöhe der Ausgabeeinheit (im Vergleich zu den im Folgenden vorgestellten Projektions-Lösungen). Nachteile sind die relative geringe Leuchtstärke, die einen Einsatz bei Tageslichtbedingungen schwierig machen sowie die Blickwinkelabhängigkeit, die bei horizontalem Einbau der Anzeigeeinheit stärker zum Tragen kommt als bei herkömmlicher vertikaler Verwendung.

Als Alternative zur Verwendung von aktiven Anzeigeeinheiten können Projektoren verwendet werden, die die darzustellende Information auf die Tischoberfläche projizieren. Hier ist zwischen Lösungen zu unterscheiden, bei denen die Projektion von oben erfolgt und jenen, bei denen von unten auf eine durchscheinende Tischoberfläche projiziert wird. Bei ersteren muss der Projektor in ausreichender Höhe über der Tischoberfläche angebracht werden, damit das projizierte Bild die notwendige Größe erreicht. Bei beschränkter Höhe nach oben kann ein Projektor mit

¹Liquid Crystal Display (Flüssigkristallanzeige)

Weitwinkelobjektiv oder ein Umlenkspiegel benutzt werden, der durch eine Vergrößerung des Abstands zwischen Projektor und Oberfläche auf einer nicht vertikalen Achse die notwendige Bauhöhe reduziert.

Der größte Nachteil einer Projektion von oben ist die Abschattung der projizierten Information bei Manipulationen der Tokens auf der Oberfläche. Dieses Problem tritt nicht auf, wenn von unten auf eine durchscheinende Oberfläche projiziert wird. Bei diesem Lösungsansatz kommt es zu keinerlei Abschattungen, die Information wird wie bei Einsatz eines Bildschirms ständig angezeigt. Nachteilig wirkt sich hier der durch den Einsatz einer durchscheinenden Oberfläche verursachte Leuchtkraftverlust aus. Bei dieser Form der Projektion wird immer ein Teil des durch den Projektor ausgestrahlten Lichts von der Oberfläche nach unten zurück reflektiert. Im Gegensatz zur Projektion von oben ist dadurch der Kontrast der Darstellung wesentlich geringer. Für den Einsatz unter Tageslichtbedingungen erscheint also der erstgenannte Ansatz generell besser geeignet. Ein kritischer Faktor ist bei der Projektion von unten der notwendige Abstand zwischen Projektor und Tisch, da sich dieser direkt auf die Höhe des Tisches auswirkt. Um eine akzeptable Bauhöhe zu erreichen – also den Tisch durch durchschnittliche große Personen bedienbar zu halten (Höhe nicht mehr als etwa 100 cm) – ist hier der Einsatz eines Umlenkspiegels nahezu unabdingbar. Beiden Projektions-Ansätzen gleich ist, dass die abzudeckende Oberfläche variabel durch den Abstand des Projektors gewählt werden kann. Grenzen sind hier nach unten durch die Fokussierbarkeit des Projektors bei kleinen Abständen und nach oben durch die Abnahme der Projektionshelligkeit bei großen Abständen gegeben. Bei großen Oberflächen ist zudem auf die verfügbare Auflösung des Projektors zu achten, da die Größe eines Bildpunktes mit zunehmendem Abstand so ansteigt, dass eine feinauflösende Projektion der Information auf der Oberfläche nicht mehr möglich ist.

Aktive Anzeige auf Tokens

Alternativ zur Darstellung auf der Tischoberfläche können bei Tabletop-Interfaces auch die Tokens selbst als Ausgabekanal dienen. Da hier das Eingabemedium gleich dem Ausgabemedium ist, ist diese Form der Ausgabe hinsichtlich der Embodiment-Dimension in die Ausprägung „full“ einzuordnen. Technologisch können je nach Art der darzustellenden Information Tokens mit Displays ausgestattet werden oder lediglich visuelle Statusanzeigen beinhalten, die Feedback über den aktuellen Zustand des Tokens geben. In beiden Fällen müssen die Tokens generell mit Elektronik und Energieversorgung ausgestattet sein und die Möglichkeit haben, selbst oder über eine Verbindung mit der Infrastruktur ihren Zustand festzustellen.

Um textuelle oder grafische Information auf Tokens darzustellen ist die Verwendung von Displays notwendig. Bei der Verwendung von herkömmlichen LCD-Displays

ist durch die benötigte Hintergrundbeleuchtung sowie der zur Aufrechterhaltung der Anzeige notwendigen Stromversorgung der Energieverbrauch verhältnismäßig hoch. Alternativ können neuere Technologien wie OLED²- (Shinar, 2004) oder eInk-Displays (Comiskey et al., 1998) verwendet werden. OLEDs bestehen aus organischen Materialien und benötigen keine Hintergrundbeleuchtung, das das Material selbst Licht emittiert. eInk verwendet eine papierartige Oberfläche zur Anzeige, strahlt selbst kein Licht aus und ist deshalb auf Umgebungshelligkeit zur Verwendung angewiesen. eInk bietet in hellen Umgebungen die besten Kontrastverhältnisse (vergleichbar mit bedrucktem Papier) kann allerdings beim heutigen Stand der Entwicklung keine Farben darstellen. Der größte Vorteil von eInk liegt in der Eigenschaft, dass die Anzeige auch ohne Energieversorgung aufrecht bleibt – Energie ist lediglich zur Änderung des Display-Inhalts notwendig.

Durch das Wegfallen der Hintergrundbeleuchtung sind OLED- und eInk-Displays wesentlich dünner als LCD-Module und können auch auf nicht ebenen Oberflächen angebracht werden. Allen drei Ansätzen gleich ist, dass zur Ansteuerung des Anzeigemoduls Elektronik notwendig ist, die die darzustellende Information auf die zur Verfügung stehenden Bildpunkte abbildet und das Display entsprechend ansteuert.

Neben der Verwendung eines Displays kann der Status eines Tokens auch mit Leuchtanzeigen in der Form von LEDs³ visualisiert werden. Der Nachteil dieses Ansatzes ist die schwierige Realisierbarkeit von komplexen Statusanzeigen – durch die auf zwei Zustände (ein/aus) beschränkte Aussagekraft einer LED sind andere als bipolare Visualisierungen schwer zu realisieren. Möglich ist die Verwendung von mehreren LEDs. Sollen dadurch mehrere voneinander unabhängige Aussagen visualisiert werden, führt diese Variante jedoch rasch zu schwer erfassbaren und kaum unterscheidbaren Visualisierungen. Lediglich die Kopplung mehrerer LEDs zur aussagekräftigen Visualisierung von dynamischen Zuständen hat sich als intuitiv erfassbar und verständlich erwiesen (Zuckerman et al., 2005). So können gekoppelte LEDs z.B. dazu verwendet werden, Flussrichtungen von Ressourcenströmen anzuzeigen, indem eine LED-Reihe entweder von links nach rechts oder von rechts nach links angesteuert wird. Auch beim Einsatz von LEDs ist eine ständige Energieversorgung notwendig. Zur Reduktion des Energieverbrauchs können wiederum die oben genannten Alternativ-Technologien OLED und eInk zum Einsatz kommen, wobei diese aktuell nicht in den Bauformen herkömmlicher LEDs angeboten werden. eInk-Anzeigen eignen sich aufgrund ihrer langsamen Schaltdauer außerdem nicht für die Realisierung dynamischer Anzeigen.

Tokens mit Aktuatoren

²Organic Light Emitting Diode

³Light Emitting Diode (Leuchtdiode)s

Neben der Verwendung von Displays zur Realisierung eines „full embodied“ Ausgabekanals können Tokens auch mit Aktuatoren ausgestattet werden, die es erlauben, das Token bzw. dessen Verhalten selbst ohne direkte Benutzerinteraktion zu beeinflussen. Beispiele für Aktuatoren sind unter anderem Vibrationsmodule oder mechanische Einheiten zur Veränderung der äußeren Form des Tokens oder dessen Position (auf der Tischoberfläche). Der Einsatz von Aktuatoren ist ob der technologischen und anwendungsspezifischen Vielfalt nicht generisch beschreibbar wie das bei reinen optischen Anzeigeeinheiten der Fall war. Aktuatoren müssen auf den jeweiligen Anwendungsfall abgestimmt sein. Ihr Einsatz ist im Allgemeinen eher disruptiv, unterbricht durch die vom Benutzer nicht selbst ausgelöste Interaktion dessen aktuelle Aktivität und zieht die Aufmerksamkeit auf sich. Dies kann im einzelnen Anwendungsfall erwünscht sein, kann aber zu unerwünschten Effekten bei der Verwendbarkeit des Systems führen.

Wie bei Anzeigemodulen müssen auch hier die Tokens mit Energie versorgt werden und mit Elektronik ausgestattet sein, die für die Ansteuerung der Aktuatoren sorgt.

Im Bereich der Tabletop Interfaces ist die Verwendung von Aktuatoren eher selten anzutreffen. Im Bereich der Ambient Interfaces kann der Einsatz von Aktuatoren aber sinnvoll sein, wenn sich das Interface so in die Umgebung seines Einsatzbereichs integrieren kann (Gross, 2003).

7.2.2. Ansätze zur entkoppelten Ausgabe

Ansätze zur entkoppelten Ausgabe werden von Fishkin (2004) in seiner Taxonomie in der Embodiment-Dimension unter den Ausprägungen „environment“ oder „distant“ eingeordnet.

„Environmental Embodiment“ ist dann gegeben, wenn Eingabe- und Ausgabekanäle räumlich nicht kohärent sind, aber Eingaben trotzdem offensichtlich Reaktionen in der unmittelbaren Umgebung auslösen. Klassische von Fishkin genannte Ansätze sind hier Audiokanäle aber auch die Veränderung von Umgebungslicht oder Temperatur. Auch olfaktorische Interfaces wären in diese Kategorie einzuordnen. In den folgenden Abschnitten werden jedoch ausschließlich audio-basierte Kanäle beschrieben, da diese im Bereich der Tabletop Interfaces Relevanz besitzen (etwa in (Kaltenbrunner et al., 2006) und (Pedersen und Hornb, 2009))

Die Ausprägung „distant“ kennzeichnet Ansätze, in denen die Eingabekanäle räumlich von den Ausgabekanälen vollkommen entkoppelt sind bzw. entkoppelt werden können. Ein Kriterium zur Einordnung eines Ansatzes unter „distant“ ist, dass Benutzer zur Beobachtung der Ausgabe nicht mehr die Eingabe im Blickfeld haben können (was bei allen anderen Ausprägungen möglich ist). Klassische Vertreter die-

ses Ansatzes sind alle Ansätze, die auf der Darstellung von Information auf herkömmlichen Bildschirmen oder Projektionsflächen basieren.

Darstellung auf Monitoren

Bei der Darstellung von Information auf Bildschirmen kommen die auch in der herkömmlichen Desktop-basierten Mensch-Maschine-Interaktion gängigen Anzeigetechnologien zur Anwendung. Im Kontext von Tabletop Interfaces ist bei Monitoren auf die Sichtbarkeit der Information für alle an der Interaktion beteiligten Personen zu achten – diese ist nicht nur von der Entfernung zum Monitor abhängig sondern bei den heute gängigen LCD-Displays auch vom Blickwinkel. Gegebenenfalls müssen mehrere Monitore verwendet werden, auf denen entweder simultan die gleiche Information dargestellt wird oder – abhängig von der Anwendung – lediglich für die jeweils eingenommene Perspektive relevante Information angezeigt wird. Mit Monitoren kann auch eine vollständig räumlich entkoppelte Darstellung realisiert werden, indem die darzustellende Information auf entfernte Displays übertragen wird. So kann zum Beispiel die Interaktion auf einem Tabletop Interface bzw. deren Auswirkungen auch räumlich entfernt verfolgt werden.

Generell muss bei entkoppelten Ausgabekanälen darauf geachtet werden, dass bei Interaktionen mit den tangiblen Eingabekanälen entsprechend eindeutig zuzuordnendes Feedback über die Ausgabekanäle rückgespiegelt wird. Bei Tabletop Interfaces ist hierbei (wiederum abhängig von der Anwendung) eine schematische Darstellung der Tischoberfläche mit einer Kennzeichnung des Bereichs, in dem eine Interaktion erkannt wurde, sinnvoll.

Projektion auf entfernte Oberflächen

Bei Ausgabe mittels Projektion auf entfernte Oberfläche (im Sinne von Oberflächen, die nicht dem eigentlichen Interface zuzuordnen sind) sind Aspekte wie die Größe des Bildes oder die Blickwinkelabhängigkeit der Darstellung meist keine Herausforderung. Mit einem Projektor können zumeist alle Benutzer ausreichend mit Information bedient werden. Ansonsten gelten die obigen Ausführungen hinsichtlich eindeutig zuzuordnendem Feedbacks analog.

Bei individueller Nutzung eines Tangible Interfaces ist in der Verwendung von Bildschirm oder Projektor noch ein unterschiedlich hoher Grad an Privatheit der Ausgabekanäle festzustellen. Während Bildschirme eher dem mit dem System interagierenden Individuum als Ausgabekanal vorbehalten bleiben, sind projizierte Informationen quasi öffentlich verfügbar. Abhängig vom Anwendungsszenario des Tangible Interfaces kann dies erwünscht sein oder nicht.

Audio-basierte Ausgabekanäle

Audio-basierte Ansätze werden hier als Vertreter von Ausgabekanälen vorgestellt, die in die Kategorie „environmental Embodiment“ eingeordnet werden. Audio-Interfaces arbeiten mit akustischer Ausgabe von Information, die im Allgemeinen in Zusammenhang mit den Eingaben am Tangible Interface stehen.

Die ausgegebene Information kann abstrakt, codiert oder natürlichsprachlich sein. Unter „abstrakter“ Information werden hier akustische Ausgaben verstanden, die nicht unmittelbar Bedeutung im sprachlichen Sinn tragen sondern auf Meldodien oder anderen Klangmustern basieren. Derartige Formen der Ausgabe kommen vor allem im künstlerischen Bereich zu Einsatz (z.B. (Kaltenbrunner et al., 2006), (Pedersen und Hornb, 2009)). „Codierte“ akustische Ausgabe deckt alle Bereiche ab, in denen Klänge verwendet werden, um Information zu transportieren, ohne das diese selbst in diesen Klängen abgebildet ist. Beispiele hierfür sind Signaltöne, die einen Fehler melden oder anderweitig die Aufmerksamkeit der Benutzer auf sich ziehen sollen. Unter „natürlichsprachlicher“ akustischer Ausgabe werden hier schließlich alle Ansätze eingeordnet, die die Ausgabe der Information direkt in Sprache umgesetzt vornehmen. Dies kann mittels gespeicherten (Teil-)Ansagen oder mittels Sprachsynthese realisiert werden. Im Gegensatz zu den anderen beiden Formen der akustischen Ausgabe ist die natürlichsprachliche Ausgabe abhängig vom Sprachverständnis der Benutzer und deshalb in einer bestimmten Konfiguration nur für einen beschränkten Benutzerkreis nutzbar.

Akustische Ausgabe ist allen ihren Ausprägungen der Embodiment-Dimension „environment“ zuzuordnen. Sie kann im Allgemeinen (ohne den Einsatz von Ohrhörern) von allen sich im Umfeld des Interfaces befindlichen Personen wahrgenommen werden. Bei codierter Ausgabe muss darauf geachtet werden, dass die Interpretation der Signale bekannt ist bzw. durch unterstützende (visuelle oder taktile) Maßnahmen erschließbar wird.

7.2.3. Technologie-Entscheidung

Auf Basis der in Abschnitt 7.1 festgelegten auszugebenden Information und den im letzten Abschnitt beschriebenen technologischen Alternativen zur Realisierung unterschiedlich ausgeprägter Ausgabekanäle können nun grundsätzliche Technologieentscheidungen getroffen werden.

Für die im Kontext der eigentlichen Modellierung auszugebenden Information (Benennungen, ...) erscheint ein kohärenter Ansatz am geeignetsten, da die Information inhärenter Bestandteil des Modells ist und deshalb räumlich mit den physischen Elementteilen (den Tokens) in eine einheitliche Modellsicht integriert werden

sollte. Mit der bereits in Kapitel 6 argumentierten Entscheidung für passive Tokens ist die Projektion der Information auf die Tischoberfläche der einzige naheliegende Ansatz. Dabei wird im Sinne der kompakteren Bauweise und der nicht vorhandene Einschränkung bezüglich Verdeckungen der Projektion durch modellierende Personen ein Lösungsansatz mit Projektion auf die Unterseite bevorzugt. Die kompaktere Bauweise ergibt sich aus der Tatsache, dass der Innenraum des Tisches ohnehin bereits für die Erfassung der Tokens durch die Kamera genutzt wird und dass bei gleichzeitiger Nutzung für die Projektion ein Auf- bzw. Überbau des Tisches zur Aufnahme des Projektors entfällt.

Die Ausgabe vergangener Modellzustände erscheint mit den gegebenen technologischen Rahmenbedingungen hingegen nicht für kohärente Ausgabe geeignet. Passive Tokens können ihre Position nicht selbstständig verändern, weshalb die auf der Tischoberfläche vorhandenen Modellierungstokens immer an den Positionen liegen, an denen sie sich im aktuelleñ Modellzustand befinden. Eine kohärente Ausgabe von vergangenen Modellzuständen auf der gleichen Oberfläche ist deswegen nicht möglich – es käme zu Überlagerungen der physischen und projizierten Modelldarstellungen. Aus diesem Grund wird ein zweiter Ausgabekanal verwendet, der die Modellierungshistorie nicht kohärent darstellt. Akustische Ansätze erscheinen ob der Komplexität der Modelle und deren inhärent visuell-diagrammatischen Darstellung im Vergleich zu visuellen Ansätzen als ungeeignet. Zu diesem Zweck wird also ein Bildschirm- oder Projektions-basierter Ausgabekanal implementiert. Dies hat auch den Vorteil, dass ins Modell zu integrierende digitale Ressourcen in ihrer „natürlichen“ Umgebung, dem Bildschirm eines Rechners manipuliert und ausgewählt werden können. Letztendlich lässt sich ein derartiger Ausgabekanal beliebig replizieren und auch für entfernte Ausgabe des Modellzustandes nutzen.

Konkret werden für das vorgestellte System also zwei Ausgabekanäle genutzt, die beide auf der visuellen Darstellung von Information basieren und sich in ihrer Ausprägung der „Embodiment“-Dimension nach Fishkin (2004) unterschieden. Der hauptsächliche Ausgabekanal ist die Tischoberfläche, auf ihr wird Information „nearby“ ausgegeben. Für Funktionen, für die diese kohärente Ausgabeform nicht adäquat einsetzbar ist, existiert ein zweiter, „distant“ Ausgabekanal in Form eines Bildschirms oder eines externen Projektors. Dieser Kanal kann in zwei Modi betrieben werden. Wird er nicht benötigt, kann er entweder deaktiviert werden (und so den Fokus auf den primären Ausgabekanal, die Tischoberfläche, lenken) oder synchron mit der Tischoberfläche den aktuellen Modellzustand anzeigen. Weiteres ist sinnvoll, wenn der Modellierungsprozess auch von Personen verfolgt werden können soll, die sich nicht unmittelbar bei der Modellierungsoberfläche befinden.

Hardwareseitig ergibt sich aus dieser Entscheidung der in Abbildung 7.1 dargestellte Aufbau. Dabei kann der in der Standardkonfiguration vorhandene Bildschirm als

7. Ausgabe

Ausgabekanal je nach den räumlichen Gegebenheiten und dem konkreten Anwendungsszenario durch einen Projektor ersetzt werden.

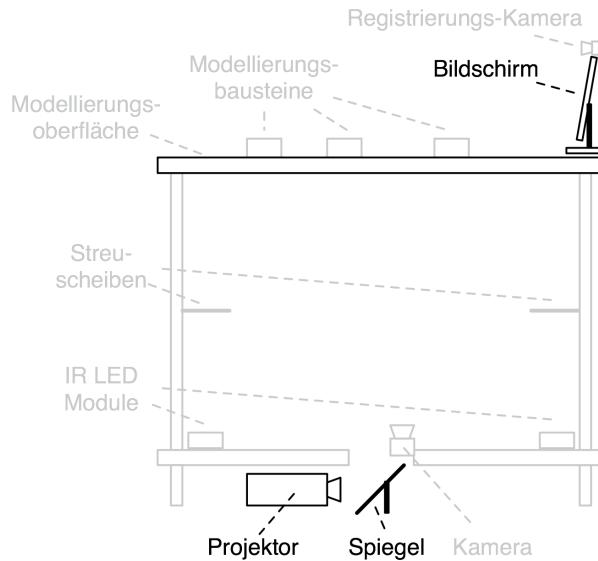


Abbildung 7.1.: Überblick über den Aufbau des Werkzeugs – Ausgabekomponenten

7.2.4. Frameworks zur Ausgabe

Im Sinne der eben getroffenen Technologieentscheidung wird nun eine Softwarekomponente benötigt, die die Ansteuerung der Ausgabekanäle erlaubt. Diese Softwarekomponente muss im Wesentlichen die Modellinformation, die über die Eingabekanäle erhoben wurde, visuell ausgeben. Zusätzlich muss sie den Funktionen zur Modellierungsunterstützung einen Kanal zur Kommunikation mit den Benutzern bieten.

Für die graphische Darstellung diagrammatischer Modelle (also vernetzter Strukturen) existieren verschiedene Frameworks die – basierend auf unterschiedlichen technologischen Ansätzen – die Darstellung von Modellelementen, der Verbindung und der Manipulation des Modells durch bereits implementierte Funktionalität ermöglichen und unterstützen.

In Frage kommende Frameworks

Die Anzahl der Möglichkeiten zur Darstellung graphischer Modelle durch Java ist groß. Die unterschiedlichen Ansätze unterscheiden sich vor allem in der angebote-

nen Abstraktionsstufe, wobei die Pole durch eine direkte Manipulation der Pixelgrafik in der das Modell dargestellt werden soll einerseits und eine rein logische, von der eigentlichen Darstellung vollständig entkoppelte Repräsentation des Modells andererseits gebildet werden. Hier wurden nur Frameworks betrachtet, die eine Verwaltung des darzustellenden Modells auf logischer Ebene ermöglichen und die eigentliche Ausgabe auf den Bildschirm übernehmen und kapseln. Kandidaten für Frameworks, die diese Forderung erfüllen⁴, waren zum Zeitpunkt der Technologie-Entscheidung:

- Eclipse GEF (Moore et al., 2004)
- JHotDraw (Gamma und Eggenschwiler, 1996)
- Piccolo (Bederson et al., 2004)

Piccolo stellte sich in der ersten Betrachtung als Framework zur Spezifikation von Benutzungsschnittstellen heraus, das die Erstellung graphischer Modelleditionen zwar erlaubt, in seinem Umfang aber weit über die benötigte Funktionalität und Flexibilität hinausgeht. Dementsprechend umfangreich und komplex ist auch der Anpassungsprozess des Frameworks an eine gegebene Anwendung. Piccolo wurde in der Betrachtung deshalb im ersten Schritt zurückgestellt und wurde später, nachdem fokussiertere Frameworks identifiziert werden konnten, aus der engeren Auswahl entfernt und nicht mehr näher betrachtet. In der engeren Auswahl standen also noch Eclipse GEF und JHotDraw. Diese beiden Frameworks werden in der Folge detaillierter betrachtet und hinsichtlich ihrer Eignung für die zu erstellende Anwendung beurteilt.

Eclipse GEF Das Graphical Editing Framework (Moore et al., 2004) stellt als Teil der Eclipse Plattform (Holzner, 2004) eine Möglichkeit dar, graphische Editoren zu erstellen und diese in einer Eclipse Applikation eingebettet auszuführen. Das GEF⁵ bildet zusammen mit EMF⁶ (Budinsky et al., 2003) auch die Grundlage für das GMF⁷ (Eclipse Foundation, 2009), dass die graphische Erstellung von domänen-spezifischen Sprachen mit nachfolgender Generierung von entsprechenden Klassen ermöglicht. Diese Klassen binden sich nahtlos in die Eclipse Infrastruktur ein und werden als Teil eines modell-getriebenen Softwareentwicklungszyklus zur Erstellung domänen-spezifischen Applikationen verwendet.

GEF behandelt lediglich die graphische Ausgabe und erlaubt die Darstellung verzweigter Strukturen, die flexibel ausgestaltet werden können. GEF baut dabei konsequent auf dem MVC-Konzept (Krasner und Pope, 1988) auf und trennt strikt

⁴identifiziert im Rahmen von (Feiner, 2008) und (Seiringer, 2008)

⁵Graphical Editing Framework (Eclipse-Komponente)

⁶Eclipse Modeling Framework

⁷Graphical Modeling Framework (Eclipse-Komponente)

zwischen Repräsentation der Daten und deren Darstellung. In dieser Eigenschaft liegt auch der größte Vorteil dieses Ansatzes. Durch die Entkopplung der Visualisierung ist eine flexible Behandlung der Ausgabe möglich. Diese kann entsprechend der Notwendigkeiten des jeweiligen Ausgabekanals angepasst werden und erlaubt auch eine dynamische Veränderung bzw. Erweiterung der Darstellung zur Laufzeit.

Der größte Nachteil des GEF-Ansatzes liegt für die hier vorgestellte Anwendung in der engen Verwobenheit mit der Eclipse-Plattform bzw. den starken Abhängigkeiten zu anderen Komponenten wie dem EMF. Durch diese Abhängigkeiten ist zum ersten der konzeptionelle und technische Einarbeitungsaufwand höher als in funktional fokussierteren Frameworks. Zum zweiten bringen GEF-basierte Applikationen eine Vielzahl von Features mit sich, die im konkreten Anwendungsfall nicht benötigt werden. Auf Seiten der Benutzungsschnittstelle kann die damit einhergehende größere Komplexität zwar vermieden werden, indem die überflüssigen Elemente ausgeblendet werden, implementierungsseitig bleibt die Komplexität und die damit einhergehende größere Anzahl von potentiellen Fehlerquellen aber bestehen.

JHotDraw Das JHotDraw-Framework (Gamma und Eggenschwiler, 1996) ist eine Java-Implementierung des Smalltalk-Frameworks HotDraw (Brant, 1995). Dieses wurde entwickelt, um einerseits eine einfache Möglichkeit zur Erstellung graphischer Editoren beliebiger Natur zu bieten und andererseits die Stärken und Möglichkeiten objektorientierter Softwareentwicklung unter Einsatz der von (REF Entwurfsmuster) vorgeschlagenen Entwurfsmuster zu zeigen.

JHotDraw nutzt intensiv objektorientierte Konzepte, vor allem die dynamische Typisierung von Objekten und implementiert einen Großteil seiner Features als Instanzen der eben erwähnten Entwurfsmuster. Dies erleichtert die Adaptierung und Erweiterung des Systems für eigenen Anwendungsfälle und ermöglicht eine rasche, unaufwändige Implementierung anwendungsspezifischer graphischer Editoren.

Ein Vorteil von JHotDraw liegt in der überschaubaren Struktur und im insgesamt für graphische Editoren-Frameworks eher kompakten Aufbau. Dies ist sowohl auf die resultierenden Applikationen bezogen, die wenig Ressourcen benötigen und ohne merkliche Verzögerung starten, als auch auf Framework-Klassen-Struktur zur Erstellung dieser Applikationen, die eine rasche Entwicklung ohne hohen Einarbeitungsaufwand oder weitgehende Parametrisierungen ermöglicht.

Ein weiterer Vorteil von JHotDraw ist die Eigenständigkeit seiner Klassenbibliothek, die von keinen anderen Bibliotheken abhängt. Die einzige zu erfüllende Abhängigkeit besteht zum Fenster-Management-System Java SWING, das ohnehin Teil des JRE⁸ ist. JHotDraw-basierte Applikationen können so ohne weiteren Installations-

⁸Java Runtime Environment

Aufwand auf allen Rechnern ausgeführt werden, auf denen ein JRE vorhanden ist. Zusätzlich ist SWING auch Teil jener Bibliotheken, die von Java Applets zur Ausführung in Browsern verwendet werden können. Jede JHotDraw-Applikation kann damit ohne Modifikation lediglich durch Austausch der Basisklasse über eine HTTP⁹-Verbindung geladen und im Browser ausgeführt werden.

Framework-Entscheidung

Für das konkret zu implementierende System erscheinen grundsätzlich beide Frameworks als geeignet. JHotDraw weist in der Standarddistribution einen geringeren, dafür aber auf die eigentliche Visualisierungsaufgabe fokussierteren Funktionsumfang auf, was sich positiv auf den für die Verwendung notwendigen Lernaufwand auswirkt. Für die Entwicklung des prototypischen Systems wurde neben diesem Grund aufgrund der leichtgewichtigeren Bibliothek und geringeren Abhängigkeiten von externen Bibliotheken sowie des größeren Umfangs von in der Standarddistribution mitgelieferten benötigten Funktionalitäten JHotDraw ausgewählt.

In einer im Kontext der hier vorgestellten Arbeit angefertigten Diplomarbeit (Feiner, 2008) wurde ein auf GEF basierendes System entwickelt, das an die existierenden Eingabekanäle angebunden werden kann und prototypisch zeigt, wie GEF für die Darstellung flexibler Modelle (im Sinne von dynamisch zugewiesener Semantik und Visualisierung) eingesetzt werden kann. Ein zusätzlicher Aspekt dieser Arbeit ist die Möglichkeit zum synchronen verteilten Betrieb mehrerer Instanzen der Applikation, die simultan am gleichen Modell arbeiten. Die Arbeit schafft damit die Voraussetzungen, synchron an mehreren (physischen oder graphischen) Schnittstellen ein Modell kooperativ zu manipulieren. Sie ist außerdem aufgrund der gemeinsamen Basis-Infrastruktur in Instanzen der Eclipse Rich Client Platform (McAffer und Lemieux, 2005) integrierbar und kann so nahtlos in komplexere, auf dieser Plattform basierende Systeme integrierbar.

7.3. Ausgabe von Information

In diesem Abschnitt wird nun auf Basis der eben getroffenen Framework-Entscheidung und unter Berücksichtigung der grundsätzlichen Entscheidung für zwei Ausgabekanäle, die in Abschnitt 7.2 getroffen wurde, die konkrete Informationsausgabe für das hier vorgestellte Werkzeug beschrieben.

Wie bereits in Kapitel 6 auf abstrakter Ebene beginnend, wird zuerst das Konzept beschrieben, das bei der Umsetzung der Ausgabe verfolgt wird. Aufbauend darauf

⁹Hypertext Transfer Protocol

wird die Architektur der Software beschrieben, die auf den in Kapitel 6 beschriebenen Komponenten aufsetzt und für die Ansteuerung der Ausgabekanäle sorgt. Nach einer detaillierten Beschreibung der konkreten Ausgabeform für alle ausgebenden Informationstypen wird schließlich die Implementierung des hier verfolgten Ansatzes in Software vorgestellt.

7.3.1. Konzept

Wie bereits oben beschrieben wird die auszugebende Information im konkreten Werkzeug auf zwei Ausgabekanäle verteilt. Grundsätzlich wird Information auf einem mit den Eingabekanälen räumlich kohärenten Kanal (der Tischoberfläche) ausgegeben. Ist dies aufgrund der Art der auszugebenden Information nicht adäquat möglich, wird der zweite Ausgabekanal als primäres Kommunikationsmedium verwendet. In der übrigen Zeit spielt der zweite Ausgabekanal eine untergeordnete Rolle und kann zur unterstützenden simultanen Visualisierung der am primären Kanal ohnehin ausgegebenen Information dienen.

Während eines Modellierungsvorgangs bleibt der Fokus der Ausgabe grundsätzlich am primären Ausgabekanal, der Tischoberfläche. Auf dieser wird Information kohärent mit den vorhandenen Tokens ausgegeben wird. In lediglich zwei Fällen erhält der zweite Ausgabekanal den Fokus:

- Wenn Information ausgegeben werden muss, die nicht kohärent mit den auf der Oberfläche vorhandenen physischen Tokens dargestellt werden kann (z.B. bei der Ausgabe von gespeicherten Modellzuständen).
- Wenn Interaktion mit den Benutzern einen Fokus auf die digitale Welt, also auf über den Rechner zugreifbare Ressourcen hat (wie bei der Auswahl von Dateien zur Einbettung in Container-Tokens).

In beiden Fällen wird das Modell auf der Oberfläche nicht simultan verändert, den Benutzern ist es möglich, sich auf die Interaktion mit dem sekundären Ausgabekanal zu konzentrieren.

Während der Arbeit mit dem Werkzeug, bei der der zweite Ausgabekanal grundsätzlich nicht benötigt wird, kann mit diesen auf unterschiedliche Arten umgegangen werden. Zum einen kann der zweite Ausgabekanal abgeschaltet werden, also keine Ausgabe über den Bildschirm oder externen Projektor mehr erfolgen. Vorteil dieser Lösung ist, dass ein Fokuswechsel auf den sekundären Ausgabekanal eindeutig erkennbar ist, da dazu die Darstellung explizit aktiviert wird. Während der eigentlichen Modellierung werden Benutzer nicht abgelenkt und können auf die Interaktion auf der Tischoberfläche fokussieren.

Zum anderen ist es möglich, am sekundären Ausgabekanal die Ausgabe des primären Kanals zu spiegeln, auf diesem also synchron die gleiche Information auszugeben wie auf der Tischoberfläche. Ein Vorteil dieser Lösung ist, dass in dieser Darstellung zusätzlich Information eingeblendet werden kann, die auf der Tischoberfläche in der Überdeckung zwischen physischen Elementen und projizierter Information nicht dargestellt werden kann. So ist unter anderem denkbar, die Abstaktionsstufe, auf der aktuell modelliert wird, am sekundären Ausgabekanal zu visualisieren. Diese Information ist aus dem Modell auf der Tischoberfläche nicht erkennbar (weder eingebettete Teilmodelle noch übergeordnete Modelle sind auf im Modell ohne weitere Interaktion sichtbar). Für den Modellierungsvorgang ist diese Information aber auch zumeist nicht unmittelbar relevant. Ist sie nun auf der sekundären Oberfläche permanent sichtbar, besteht für Benutzer die Möglichkeit, rasch und ohne zusätzlichen Interaktionsaufwand einen Überblick über den Kontext des aktuell bearbeiteten (Teil-)Modells zu erhalten.

Ein zweiter Aspekt der Spiegelung der Ausgabe auf dem sekundären Ausgabekanal ist das unmittelbare Feedback, das Benutzern über die Tätigkeit des Systems zur Verfügung gestellt wird. Wenn ein Modellierungstoken bewegt oder geöffnet wird, erfolgt eine unmittelbare Reaktion in der Modellvisualisierung auf dem sekundären Ausgabekanal. Auch eventuelle Fehlerkennungen werden offensichtlich, wenn z.B. ein Element von der sekundären Visualisierung verschwindet, obwohl es auf der Oberfläche noch vorhanden ist. Für Benutzer ist damit das Systemverhalten klarer erkennbar, die Bildung eines mentalen Modells zur Erklärung der Funktionsweise des Werkzeugs ist einfacher und lenkt nach der Einarbeitungsphase nicht mehr von der eigentlichen Tätigkeit ab (Norman, 1983a). Letztendlich besteht bei der Einbindung mehrerer Benutzer die Möglichkeit, allen Beteiligten einen Überblick über den aktuellen Modellzustand zur Verfügung zu stellen, auch wenn diese im Moment nicht aktiv am Werkzeug arbeiten. Nachteile dieses Ansatzes liegen einerseits in der möglichen Ablenkung der Benutzer von der eigentlichen Modellierungsoberfläche (was insofern weitgehend unkritisch ist, als dass die gesamte Information über den sekundären Informationskanal zur Verfügung gestellt wird), vor allem aber darin, dass der Wechsel des Interaktionsfokus von einem Ausgabekanal zum anderen nicht mehr so explizit sichtbar wird wie im ersten Ansatz. Benutzer sind so unter Umständen desorientiert und können dem Systemverhalten nicht folgen.

Aus den vorangegangenen Ausführungen ist ersichtlich, dass beide Ansätze Vor- und Nachteile bieten, die nicht eindeutig für eine der beiden Lösungen sprechen. Vielmehr muss im einzelnen Anwendungsfall abgewogen werden, welche Lösung eher geeignet ist. Einflussfaktoren sind dabei die Anzahl der an der Modellierung teilnehmenden Benutzer (bessere Verfolgbarkeit des Modellierungsvorgangs bei synchroner Anzeige auf beiden Kanälen), die Modellierungsaufgabe an sich (generelle Notwendigkeit der Nutzung des sekundären Ausgabekanals) sowie das persönliche

Modellierungsverhalten der Beteiligten (eher am Bildschirm oder eher an der Tischoberfläche orientiert). Aus diesen Gründen werden beide Formen der Verwendung des sekundären Ausgabekanals unterstützt, wobei die Benutzer durch eine Tastenkombination zwischen den beiden Modi umschalten können – also ggf. auch in unterschiedlichen Phasen eines Modellierungsvorgangs eine unterschiedliche Konfiguration der Ausgabekanäle nutzen können.

7.3.2. Architektur

Wie im letzten Abschnitt beschrieben, unterscheidet sich die Ausgabe von Information auf den beiden Ausgabekanälen nur in einigen Fällen, im synchronen Ausgabemodus wird weitgehend die gleiche Information zum Teil – den unterschiedlichen Ausgabemedien geschuldet – in unterschiedlichen Ausgabeformen dargestellt.

Grundsätzlich wird die Architektur der Ausgabekanal-Verwaltung erweiterbar angelegt, um veränderte Anforderungen an der Werkzeug oder erweiterte Funktionalität umsetzen zu können. Basis dieser erweiterbaren Architektur ist das in Abschnitt 6.4.8 bereits beschriebene Modul zur Verteilung des aktuellen Modellzustandes an die verwendeten Ausgabekanäle. Jeder Ausgabekanal, der über Änderungen am Modell an den Eingabekanälen benachrichtigt werden soll, muss ein definiertes Interface implementieren, das jene Methoden definiert, die zur Datenübermittlung zwischen dem Interpretationsmodul und den Ausgabekanälen notwendig sind. In der konkreten Implementierung sind zur Zeit zwei auf JHotDraw basierende Ausgabekanäle umgesetzt, zusätzlich existiert ein weiteres Modul, das gegenüber dem System als Ausgabekanal agiert, die übermittelte Information selbst jedoch wieder an beliebige via TCP^{io}/IPⁱⁱ angebundene Clients verteilt (siehe Abbildung 7.2). Auf diesem Weg können entfernte Ausgabekanäle realisiert werden, die es erlauben, den Modellierungsprozess auch ohne räumliche Anwesenheit zur verfolgen. Mittels der gleichen Schnittstelle können auch weitere, nicht auf visuelle Ausgabe beschränkte Kanäle wie z.B. akustische Benachrichtigungen mit Information versorgt werden.

Durch die flexible Architektur von JHotDraw ist es möglich, unterschiedliche Ausgabekanäle mit der gleichen Codebasis durch den Einsatz verschiedener Basisklassen als vollständig eigenständige Applikation auszuführen oder als Applet in Webbrowsern zu starten (siehe Abbildung 7.10). Auf diese Weise ist es möglich, die einmal implementierte Visualisierung des Modellzustandes für die lokalen Ausgabekanäle sowie für entfernte Betrachter zu verwenden, die alle über die gleiche Schnittstelle synchron mit Information versorgt werden.

^{io}Transport Control Protocol

ⁱⁱInternet Protocol

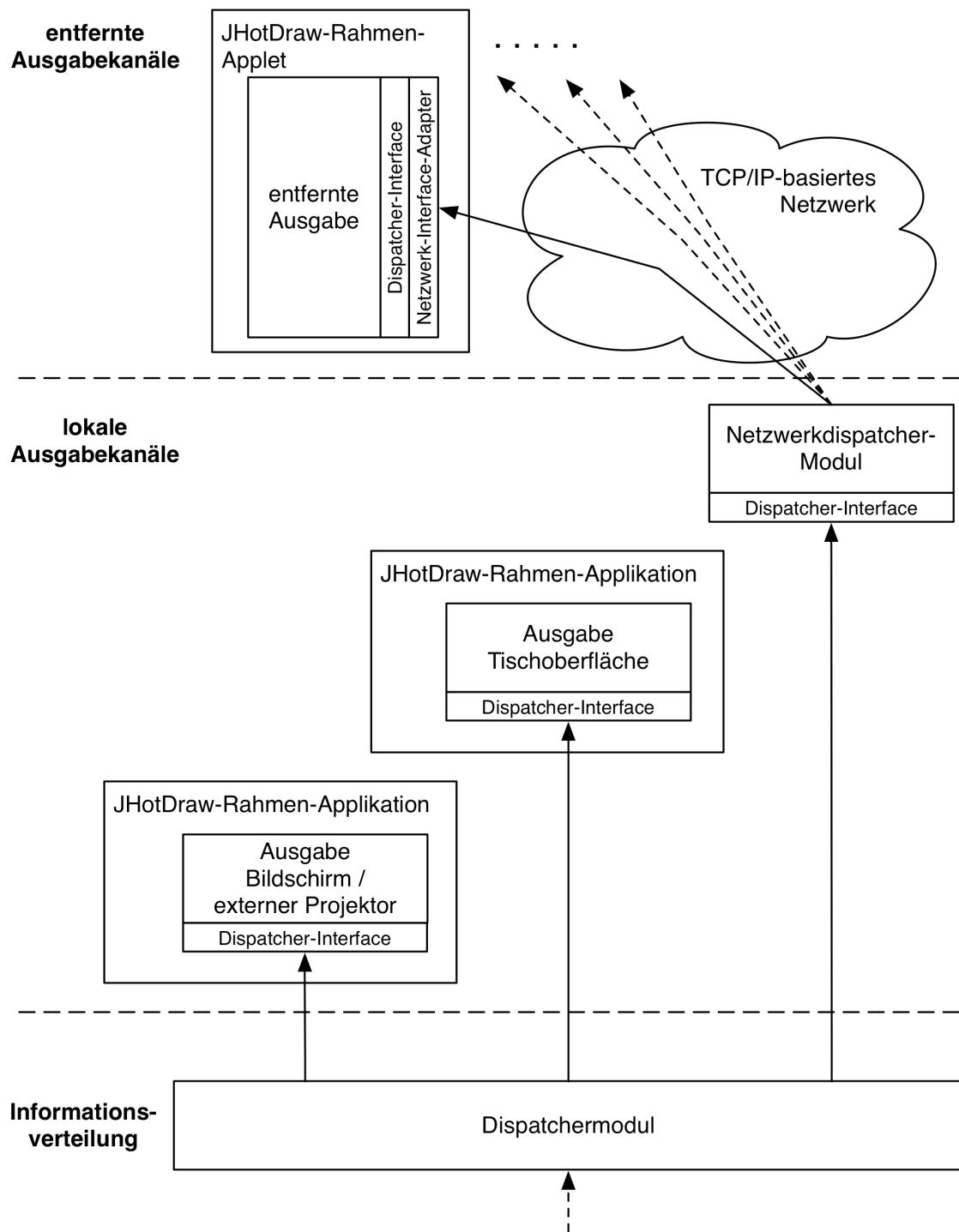


Abbildung 7.2.: Softwarearchitektur zur Verwaltung der Ausgabekanäle

7.3.3. Ausgabe von Information zum Modell

Um im Folgenden auf die Details der Implementierung eingehen zu können, muss an dieser Stelle noch ausgeführt werden, welche Arten von Information in welcher Form über welchen Kanal ausgegeben werden. Grundsätzlich sind konzeptionell zwei Arten von Information zu unterscheiden. Dieser Abschnitt behandelt die Ausgabe von Information zum Modell selbst, im folgenden Abschnitt werden jene Informationen Ausgabekanälen zugeordnet, die der Kontrolle des Systems und der Unterstützung des Modellierungsvorgangs dienen.

Information zu Modellelementen

Information, die im Zusammenhang mit einzelnen Modellelementen ausgegeben wird, wird unmittelbar auf der Modellierungsfläche, also am primären Ausgabekanal, dargestellt. Wenn der sekundäre Ausgabekanal für synchrone Modelldarstellung konfiguriert ist oder entfernte Modellbetrachter eingesetzt werden, so wird diese Information auch dort dargestellt. Die Ausgabe auf entfernten Ausgabekanälen ist dabei immer identisch mit jener am sekundären Ausgabekanal. Dabei wird das Modellelement selbst graphisch repräsentiert, um eine Zuordnung der Information zu ermöglichen. Auf der Tischoberfläche ist eine separate Darstellung des Modellelements nicht notwendig, da dieses durch die kohärente Eingabe ohnedies durch das entsprechende Modellierungstoken dargestellt wird (siehe Abbildung 7.3)

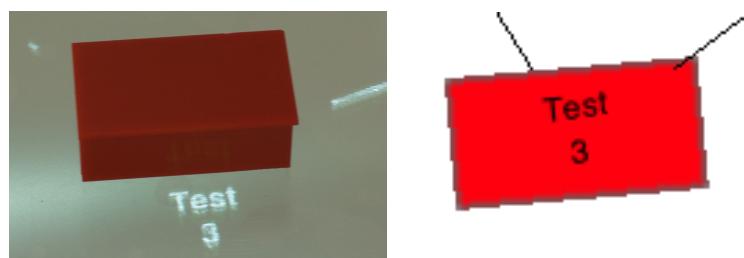


Abbildung 7.3.: Darstellung von Modellelementen
(links: Tischoberfläche, rechts: Bildschirm)

In der aktuellen Implementierung wird lediglich die Bezeichnung des Tokens auf beiden Ausgabekanälen explizit ausgegeben. Der Typ des Tokens ist in der Farb- und Formgebung codiert. Auf dem sekundären Ausgabekanal sind die Elemente in der graphischen Darstellung den physischen Elementen nachempfunden und bilden deren Farbe und Grundriss ab. Zusätzlich wird eine Element, dass als Container fungiert, am sekundären Ausgabekanal durch eine Markierung gekennzeichnet.

Ebenfalls nur auf der sekundären Oberfläche explizit ausgegeben wird die Information hinsichtlich der aktuellen Rotation eines Modellierungstokens. Auf der Oberfläche ist diese am physischen Token erkennbar bzw. wird durch diese festgelegt. In der graphischen Repräsentation am sekundären Ausgabekanal muss diese Rotation ebenfalls abgebildet und nachgeführt werden.

Information zu Verbindern

Da Verbinder nicht physisch existieren sondern immer über die Ausgabekanäle dargestellt werden müssen, gelten die nun folgenden Ausführungen für beide Ausgabekanäle. Wird ein Verbinder hergestellt, so kann neben den beiden Endpunkten (Modellelementen) auch noch die Richtung des Verbinders sowie eine Bezeichnung festgelegt werden.

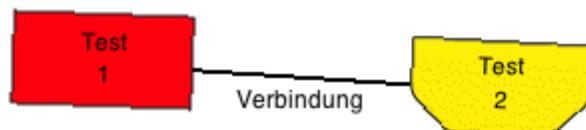
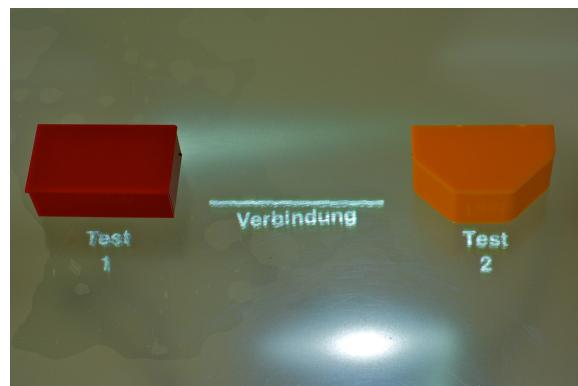


Abbildung 7.4.: Darstellung von Verbindern
(oben Tischoberfläche, unten Bildschirm)

Ein Verbinder wird auf den Ausgabekanälen als Linie dargestellt die zwei Modellierungstokens bzw. deren graphische Repräsentation verbindet (siehe Abbildung 7.4). Die Richtung des Verbinders wird über Pfeilspitzen angegeben, die an den Enden

der Linien angebracht werden (siehe Abbildung 7.5). Dabei können diese Pfeilspitzen fehlen (ungerichteter Verbinder), an einem Ende angebracht werden (gerichteter Verbinder) oder an beiden Enden dargestellt werden (bidirektonaler Verbinder). Die Ausgabe der Benennung eines Verbinders erfolgt – sofern vorhanden – zentriert auf halbem Weg zwischen den verbundenen Modellelementen unterhalb der Linie.

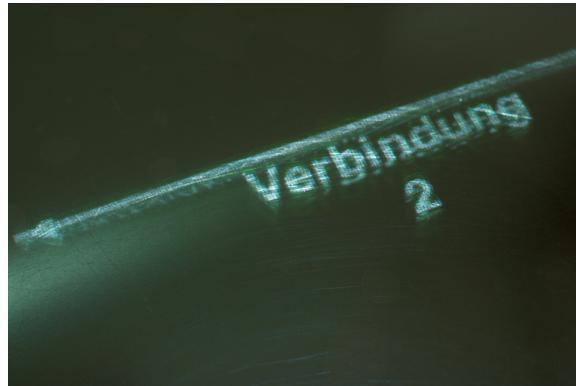


Abbildung 7.5.: Darstellung gerichteter Verbindner

Information zu eingebetteten Elementen

Im Kontext der Ausgabe von Information zu eingebetteten Elementen müssen unterschiedliche Modellaspekte berücksichtigt werden. Im einzelnen sind dies:

- Öffnungszustand von Modellelementen
- Anzahl und Art der eingebetteten Elemente
- Information, welche an eingebettete Elemente gebunden ist

Der Öffnungszustand von Modellelementen ist auf der Tischoberfläche durch den Zustand des jeweiligen Containertokens ersichtlich. Auch die Art und Anzahl der eingebetteten Elemente ist sichtbar, wenn ein Containertoken geöffnet ist. Am sekundären Ausgabekanal, also auf dem Bildschirm wird ein geöffnetes Modellelement vergrößert dargestellt, um innerhalb der Fläche des Elements visuelle Repräsentationen der eingebetteten Elemente einblenden zu können. Somit ist auch am sekundären Ausgabekanal erkennbar, wann bzw. dass ein Element geöffnet ist (siehe Abbildung 7.6).

Jene Information, die an die eingebetteten Elemente gebunden ist, wird auf beiden Kanälen nicht ohne explizite Benutzerinteraktion dargestellt. Fordern die Benutzer auf einem Eingabekanal die Darstellung der eingebetteten Information an, so wird diese auf dem sekundären, nicht kohärenten Ausgabekanal dargestellt. Eine Darstellung der angebundenen Information auf der Tischoberfläche würde durch

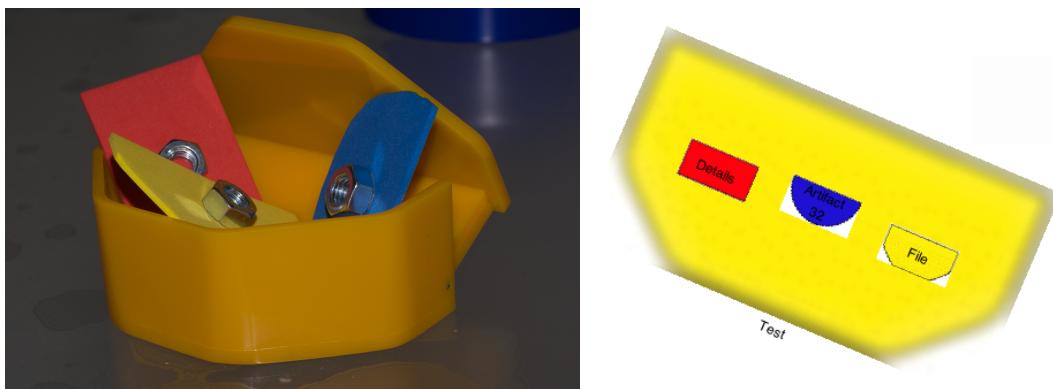


Abbildung 7.6.: Darstellung von Containern und eingebetteten Elementen
(links Tischoberfläche, rechts Bildschirm)

die physisch vorhanden Modellelemente gestört. Die eingebetteten Submodelle bzw. die sonstigen digitalen Ressourcen werden also auf dem Bildschirm dargestellt. Die Submodelle ersetzen dabei den aktuellen Modellzustand temporär, andere Ressourcen werden mittels der im Betriebssystem definierten Standardapplikation geöffnet.

7.3.4. Ausgabe zur Kontrolle des Systems

Neben der eigentlichen Modellinformation wird auch Information ausgegeben, die zum Feedback über den Systemzustand oder der Kontrolle desselben notwendig ist. Wo möglich wird auch in diesem Fall die Tischoberfläche als primärer Ausgabekanal genutzt.

Zustands- und Ereignismeldungen

Bei der Verwendung der die Modellierung unterstützenden Werkzeuge muss den Benutzern Feedback über deren Erkennung bzw. den durch diese ausgelösten Änderung des Systemzustandes gegeben werden. Dabei ist zwischen jenen Interaktionen zu unterscheiden, die ein Ereignis auslösen und jenen, die den Systemzustand permanent ändern. Dieser Unterschied muss auch in der Visualisierung den Benutzern kommuniziert werden. Dabei kann es durch die vom System nicht beeinflussbaren physischen Werkzeugtokens zu potentiellen Mißverständnissen kommen. Wird ein Werkzeugtoken, das lediglich ein Ereignis auslöst (wie zum Beispiel das Markierungstoken), von den Modellierenden auf der Oberfläche belassen, kann dieses als unter Umständen als zustandsverändernd und nicht als ereignisauslösend wahrgenommen werden.

Elementauswahl Wird mittels dem Markierungstoken ein Element ausgewählt, so müssen die Benutzer über die Erkennung der Markierung benachrichtigt werden. Dabei muss erkenntlich sein, welches Element ausgewählt wurde und wie lange die Auswahl gültig ist. Die Auswahl eines Elements ist grundsätzlich ein Ereignis, ist jedoch immer Teil einer umfassenderen Interaktion, in der die Benutzer das ausgewählte Element verwenden (z.B. dieses benennen).

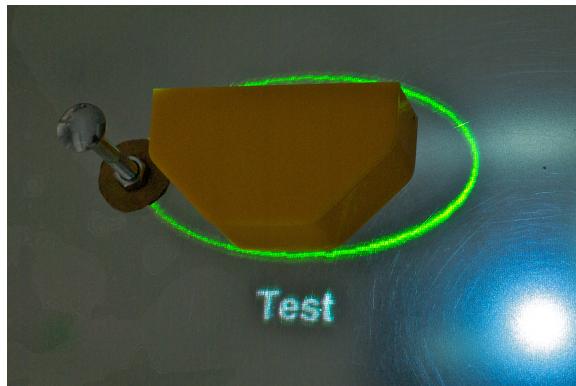


Abbildung 7.7.: Markierung von Modellelementen

Die Anzeige der Auswahl erfolgt auf beiden Ausgabekanälen mittels einer ovalen Markierung, die das ausgewählte Element umschließt (siehe Abbildung 7.7). Diese wird solange angezeigt, bis die Interaktion, der sie zuzuordnen ist, abgeschlossen ist oder ein anderes Element ausgewählt wird. Außerdem ist es möglich, eine Markierung durch explizite Interaktion (erneute Auswahl mit dem Markierungstoken) wieder zu entfernen.

Verbindungsherstellung Die Herstellung von Verbindungen kann auf zwei unterschiedliche Arten durch die Benutzer ausgelöst werden. Wird eine Verbindung durch die Auswahl zweier Modellelemente hergestellt, werden die eben beschriebenen ovalen Markierungen zur Visualisierung der Auswahl verwendet. Dabei kann es zur Auswahl eines Elements als Endpunkt einer gerichteten Verbindung kommen. Die ovale Markierung wechselt in diesem Fall die Farbe, um die erfolgreiche Erkennung einer Pfeilspitze rückzumelden. Nachdem beide Endpunkte erkannt und markiert wurden, wird unmittelbar die Verbindung hergestellt. Diese wird initial als eine Linie dargestellt, die in Farbe und Strichstärke hervorsticht, und in der Folge in einer Animation durch eine herkömmliche Verbindung ersetzt. Diese Animation dient dazu, das Ereignis der Verbindungsherstellung klar zu visualisieren und die Aufmerksamkeit der Benutzer auf dieses Ereignis zu lenken.

Werden zwei Elemente durch ihre räumliche Nähe zueinander in Beziehung gesetzt und eine Verbindung hergestellt, entfallen die ovalen Markierungen (da keine explizite Auswahl der Elemente erfolgt). Es wird lediglich die Animation zur Verbindungsherstellung dargestellt. Auf diesem Wege ist keine direkte Herstellung von gerichteten Verbindern möglich.

explizite Snapshots Im Zusammenhang mit der Speicherung der Modellierungshistorie (also der über die Zeit veränderten Modellzustände) ist es möglich, neben der automatischen Speicherung auch explizit der Aufnahme eines Snapshots auszulösen. Während bei der automatischen Speicherung kein Feedback an die Benutzer gegeben wird (da auch keine Interaktion zum Auslösen derselben stattfindet), wird im Falle einer expliziten Speicherung den Benutzern die erfolgreiche Durchführung der Aktion rückgemeldet. Die Speicherung des Snapshots ist ein Ereignis ohne zeitliche Ausdehnung und wird dementsprechend visualisiert. Dies erfolgt in Form eines kurzen Aufblitzens sowohl der Tischoberfläche als auch des Bildschirms. Die Verwendung der gespeicherten Modellzustände und deren Visualisierung wird in Abschnitt 7.3.4 im Detail behandelt.

Löschmodus Im Gegensatz den in den vorangegangenen Abschnitten behandelten Werkzeugen löst der Einsatz des Lösch-Tokens kein Ereignis aus sondern schaltet das System in einen anderen Zustand, solange es auf der Oberfläche vorhanden ist. Dies spiegelt sich auch in der Visualisierung wieder. Solange das Lösch-Token von der Kamera erfasst wird, wird auf beiden Ausgabekanälen der Hintergrund des Modells (also die gesamte Tischoberfläche bzw. der gesamte Bildschirm) in roter Farbe dargestellt, während im Vordergrund nach wie vor der aktuelle Modellzustand visualisiert wird. Dies entspricht den Interaktionsmöglichkeiten der Benutzer – das Modell kann nach wie vor manipuliert werden, jene Interaktionen, die aber im Normalfall eine Verbindung herstellen, löschen diese in diesem Modus wieder. Wird das Lösch-Token entfernt, wechselt der Hintergrund wieder auf die Standard-Farbe, das System reagiert auf die betroffenen Interaktionen wieder mit der Herstellung eines Verbinders.

Abruf der Modellierungshistorie

Der Abruf der Modellierungshistorie wird mit dem runden Kontroll-Token ausgelöst und kontrolliert. Hinsichtlich der Darstellung ist die Modellierungshistorie die erste der hier beschriebenen Zustands- und Ereignismeldungen, bei der keine kohärente Darstellung möglich ist. Auf der Tischoberfläche befinden sich zum Zeitpunkt der Aktivierung des Historienmodus alle Tokens, die im Elemente im aktuellen Modellzustand repräsentieren. Da der Historienmodus als Referenz dient, in der

7. Ausgabe

die Entstehung des Models rekapituliert werden kann, die gespeicherten Modellzustände aber im Allgemeinen nicht den aktuellen Zustand ersetzen sollen, verbleiben diese Tokens auch auf der Oberfläche. Damit kann nach Deaktivierung des Historienmodus der Modellierungsvorgang unmittelbar fortgesetzt werden. Die auf der Oberfläche befindlichen Tokens stören aber die Darstellung gespeicherter Modellzustände, da sie Teile verdecken und keinen Bezug zum projizierten historischen Modell haben.

Die Information des Historienmodus wird deshalb ausschließlich auf dem sekundären Ausgabekanal – dem Bildschirm – dargestellt (die Kontrolle verbleibt mit dem runden Kontroll-Token auf der Tischoberfläche). Die gespeicherten Modellzustände ersetzen dabei den aktuellen Modellzustand, der im Normalbetrieb synchron zur Tischoberfläche dargestellt wird. Ist das System so konfiguriert, dass der sekundäre Ausgabekanal nicht synchron betrieben wird, so wird er bei Erkennung des Historienmodus aktiviert.

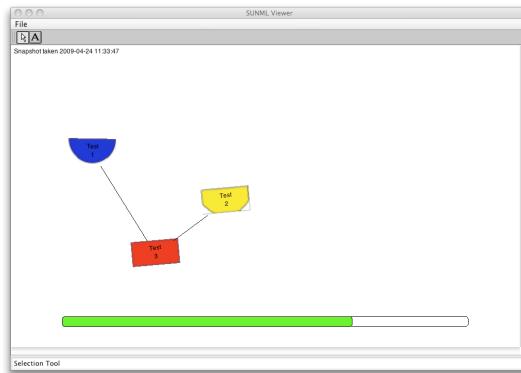


Abbildung 7.8.: Darstellung der Modellierungshistorie

Zusätzlich zur graphischen Darstellung des gespeicherten Modellzustandes wird ein Zeitstempel in der linken oberen Ecke eingeblendet, der den Zeitpunkt der Speicherung angibt. Am unteren Rand des Bildes wird ein Balken eingeblendet, der auf den relativen Zeitpunkt der Aufnahme (bezogen auf die Gesamtzahl der gespeicherten Zustände) angibt (siehe Abbildung 7.8). Die Schrittweite (also die Verlängerung des Balken, die durch das Umschalten auf einen unmittelbar folgenden Zustand verursacht wird) wird dazu dynamisch an die Anzahl der gespeicherten Modellzustände angepasst.

Ist der erste bzw. letzte Modellzustand erreicht, werden weitere entsprechende Kommandos durch Drehen des Kontroll-Tokens ignoriert. Die Benutzer werden durch einen vollständig leeren bzw. ausgefüllten Balken auf das Erreichen des jeweiligen Endpunktes hingewiesen. Wird das Kontroll-Token von der Tischoberfläche

entfernt, so wird der Historienmodus deaktiviert. Die Darstellung auf dem sekundären Ausgabekanal wird wieder auf die synchrone Darstellung des aktuellen Modellzustandes geschalten bzw. wird diese deaktiviert, falls das System dementsprechend konfiguriert ist.

Wiederherstellungsunterstützung

Im Rahmen des Abrufs der Modellierungshistorie kann von den Benutzern die Wiederherstellungsunterstützung aktiviert werden. Im Rahmen der Wiederherstellungsunterstützung leitet das System die Benutzer bei der Rekonstruktion eines gespeicherten Modellzustandes an. Dazu werden auf der Tischoberfläche und am Bildschirm (sofern aktiviert) – also auf beiden Ausgabekanälen – Hinweise angezeigt, wie der aktuelle Modellzustand verändert werden muss, um den gespeicherten Zustand wieder herzustellen. Da der aktuelle Modellzustand als Ausgangspunkt dient, kann eine kohärente Visualisierung zum Einsatz kommen.

Bei der Aktivierung der Wiederherstellungsunterstützung werden die im Modell befindlichen Verbindungen ausgeblendet, da diese nicht manuell verändert werden müssen. Die notwendige manuelle Intervention der Benutzer zur Wiederherstellung beschränkt sich auf die Platzierung der Modellierungstokens an die dem gespeicherten Modellzustand entsprechenden Orte. In der ersten Phase werden vom System schrittweise alle Tokens markiert, die im aktuellen Modell enthalten sind und im gespeicherten Modellzustand nicht vorhanden sind, also entfernt werden müssen. Schrittweise heißt in diesem Zusammenhang, dass immer nur ein Token markiert wird, sobald dieses entfernt wurde, wird das nächste Token zur Entfernung markiert. Die Markierung ist als rote, ovale Hinterlegung des Tokens umgesetzt.

In der zweiten Phase werden jene Tokens behandelt, die sowohl im aktuellen als auch im gespeicherten Modellzustand enthalten sind, sich aber jeweils an unterschiedlichen Positionen befinden, also verschoben werden müssen. Dazu wird wiederum schrittweise eine Markierung an der Zielposition des Tokens eingeblendet und diese mit einem Pfeil mit der aktuellen Position des betroffenen Tokens verbunden (siehe Abbildung 7.9).

In der letzten Phase werden alle Tokens hinzugefügt, die im aktuellen Modell nicht vorhanden sind, es im gespeicherten Modellzustand aber waren. Diese werden schrittweise durch grüne, ovale Markierungen an den jeweiligen Positionen angezeigt, wobei die Benennung des Tokens in die Markierung eingeblendet wird, um das jeweilige zu platzierende Element identifizieren zu können.

Die zur Rekonstruktion des Modellzustandes notwendigen Schritte sind nicht vorberechnet sondern werden vielmehr Schritt für Schritt aus der aktuell auf der Tischoberfläche befindlichen Token-Konstellation extrahiert. Dadurch ist es mög-

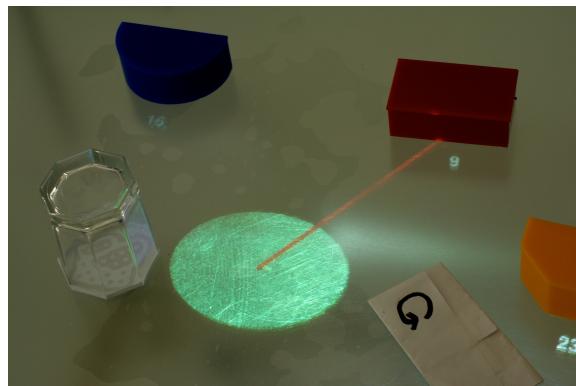


Abbildung 7.9.: Unterstützung der Wiederherstellung von Modellzuständen

lich, auch irrtümlicherweise während der Wiederherstellung entfernte, hinzugefügte oder vor allem verschobene Tokens zu korrigieren, auch wenn die Rekonstruktion grundsätzlich schon in eine spätere Phase getreten wäre. So würde das System beispielsweise nach Abschluss eines Schrittes in der dritten Phase (Elemente hinzufügen) auf eine Anweisung in der zweiten Phase (Verschieben) zurückgreifen, wenn durch das Hinzufügen ein anderes, bereits an der korrekten Position befindliches Element verschoben worden wäre.

Sobald sich die Positionen der Tokens mit jenen des gespeicherten Modellzustandes decken, ist die notwendige Benutzerintervention zur Wiederherstellung abgeschlossen. Die Benutzer werden nun aufgefordert, das Historien-Kontroll-Token und das Wiederherstellungs-Token zu entfernen (sofern sich diese noch auf der Oberfläche befinden). Dieser Schritt schließt die Wiederherstellungsunterstützung ab. Das System blendet die Verbinder (nun aus dem gespeicherten Modellzustand) wieder ein und zeigt damit den (neuen) aktuellen Modellzustand an.

7.4. Umsetzung der Ausgabe mit Software

Wie in Abschnitt 7.3.1 bereits beschrieben, basiert die Implementierung des Ausgabekanäle auf dem JHotDraw-Framework. Die Implementierung erfolgt dabei in Modulen, die die Art und Anzahl der Ausgabekanäle flexibel erweiterbar macht. Die Anforderungen an die Ausgabekanäle unterscheiden sich zwar durchaus im Detail, im Allgemeinen ähneln sie sich aber stark. Dies legt die Verwendung einer gemeinsamen Basisklasse nahe, die alle verallgemeinerbaren Aufgaben der Ausgabekanäle umfasst. Von ihr werden die konkreten Implementierungen der Ausgabekanäle mit ihren spezifischen Funktionen abgeleitet werden. So müssen auf dem bildschirm-basierten Ausgabekanal unter anderem die Modellelemente graphisch dargestellt

werden, wohingegen diese auf der Tischoberfläche in Form der physischen Modellierungstokens bereits vorhanden sind. Beiden Kanälen gemein ist hingegen zum Beispiel die Behandlung der Verbindungen, die hier wie dort dargestellt bzw. projiziert werden müssen. Durch das nur partiell unterschiedliche Verhalten der beiden Ausgabekanäle ist es möglich, diese von einer gemeinsamen Basisklasse abzuleiten und die Spezifika in Subklassen zu implementieren. Dies reduziert auch den Wartungsaufwand der Codebasis und vermindert die Gefahr von Inkonsistenzen.

Konkret wurde die Architektur wie in 7.10 dargestellt auf Klassen abgebildet. Zentrale Schnittstelle zu den informationsliefernden Modulen ist das Interface `DispatcherListener`, dass eine Reihe von Methoden definiert, die Ausgabekanäle implementieren müssen, falls sie an die Informationseingabe- und -interpretations-Infrastruktur angebunden werden sollen. Das Interface ist allgemein einsetzbar, kann also auch durch Ausgabekanäle anderer technologischer Ansätze implementiert werden. Als zentrales Element implementiert die Klasse zur Ausgabe von Information auf dem sekundären Ausgabekanal (dem Bildschirm) dieses Interface. Diese Klasse wurde deswegen als zentraler Knotenpunkt gewählt, weil sie im Vergleich mit den anderen Ausgabekanälen den größten Anteil an auszugegebender Information besitzt. Die Klasse zur Ausgabe von Information auf der Tischoberfläche implementiert lediglich ein Subset an Visualisierungs-Methoden (so müssen z.B. die Elemente selbst nicht ausgegeben werden) und wird deshalb von jener zur Behandlung des sekundären Ausgabekanals abgeleitet. Die nicht benötigten Methoden werden mit leeren Methoden überschrieben. Die zur Ausgabe benötigten Methoden werden unverändert übernommen, unterschiedliche Darstellungsformen (etwa die Stichstärke bei Verbindungen) werden durch Parametrisierung der Datenklassen realisiert, die die darzustellende Information kapseln.

Beide lokale Ausgabekanäle betten die eigentliche Visualisierungs-Klasse in einen JHotDraw-Rahmen ein, der die Applikationsinfrastruktur zur Verfügung stellt. In diesem Fall ist dies die Klasse `DrawApplication`, die eine direkt ausführbare Java-Applikation erzeugt. Die Einbindung der eigentlichen Ausgabekanäle erfolgt über die Klasse `DrawingView`, die als Basisklasse für die die Ausgabekanäle bedienenden Klassen fungiert. Zur Anzeige auf der Tischoberfläche wird die JHotDraw-Applikation in den Vollbild-Modus geschalten und mit schwarzem Hintergrund versehen. Auf der Tischoberfläche ist dadurch keine Projektion zu erkennen, wenn sich keine Modellelemente auf ihr befinden. Die Anzeige am sekundären Ausgabekanal erfolgt in einem regulären Fenster.

Die Ausgabe auf entfernten (visuellen) Kanälen basiert ebenfalls auf der Klasse, die den sekundären Ausgabekanal bedient. Tatsächlich ist die Form der Visualisierung identisch mit einem im Modus synchroner Darstellung betriebenen sekundären Ausgabekanal. Die Implementierung von entfernten Kanälen unterscheidet sich

7. Ausgabe

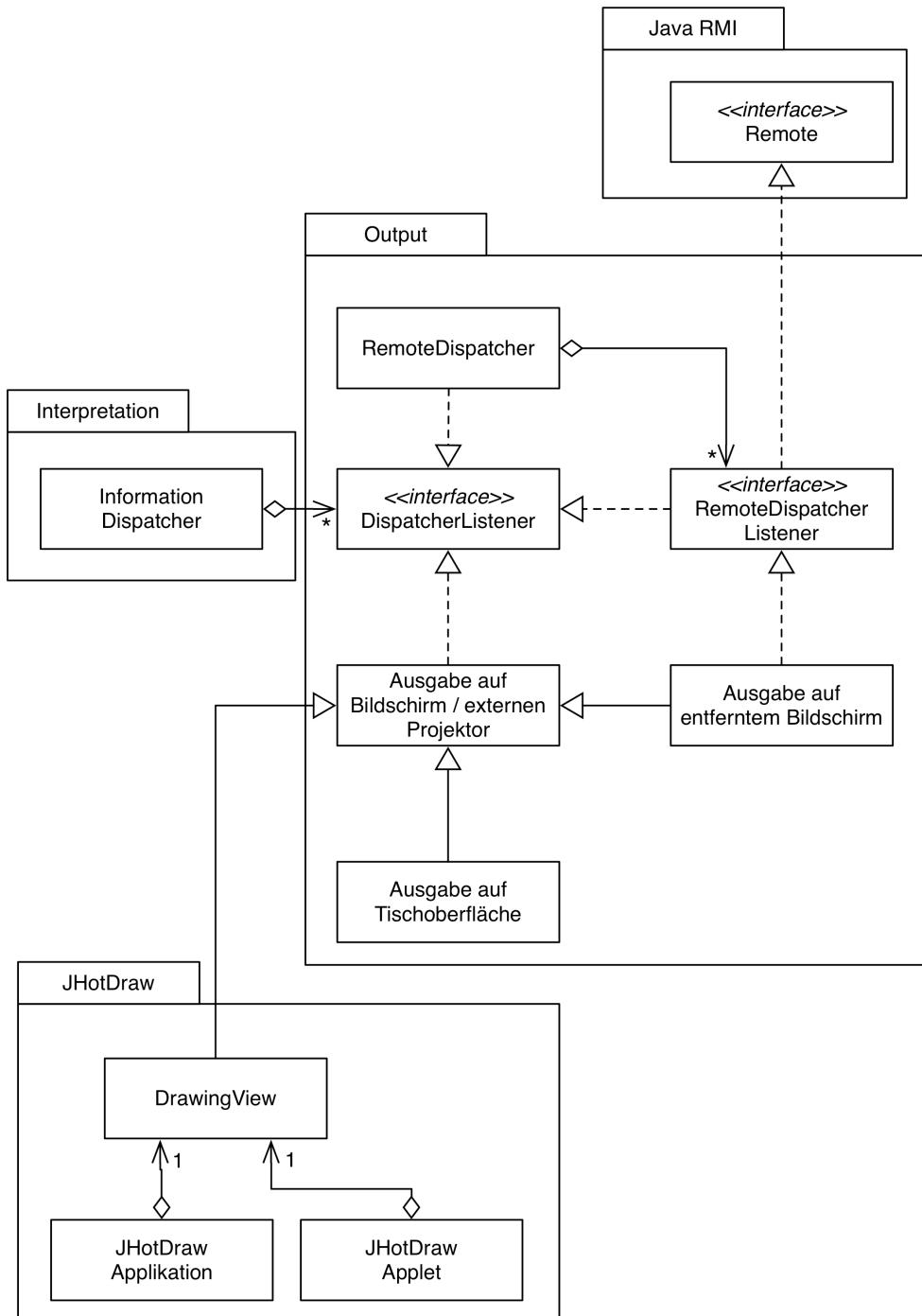


Abbildung 7.10.: Zusammenhänge der Klassen zur Ausgabebehandlung

ausschließlich in der Übermittlung der darzustellenden Information über eine Netzwerkverbindung.

7.4.1. Ausgabe des Modellzustands

Die Zustand des Modells, also Art und Parameter der auf der Oberfläche befindlichen Elemente und Verbindungen sind in Objekten der Klassen `BasicBuildingBlock` bzw. `Connector` abgebildet. Diese Klassen kapseln sämtliche Information, die zur Beschreibung des Elementzustandes notwendig ist. Diese Objekte sind außerdem in der Lage, sich durch ihre `clone`-Methode selbst vollständig zu reproduzieren. "Vollständig" bedeutet in diesem Zusammenhang, dass im Duplikat keine Referenzen mehr auf die Attribute des Originalobjekts vorhanden sind – das Objekt wird im Sinne einer "Deep-Copy" vollständig reproduziert. Dies ist notwendig, um gespeicherte Modellzustände vollständig vom aktuellen Modellzustand zu entkoppeln und persistent abzulegen. Die Ausgabe erfolgt mittels einer `CompositeFigure` (eine Komponente des JHotDraw-Frameworks), in die die graphische Repräsentation des Elements, dessen Benennung sowie ggf. die eingebetteten Elemente geschrieben werden. Eventuell vorhandene Markierungen (wie jene, die durch das Auswahltoken verursacht werden oder die im Rahmen der Wiederherstellungsunterstützung als Hinweis zum Entfernen eines Elements hinzugefügt werden) werden ebenfalls direkt in die `CompositeFigure` aufgenommen. Die hat den Vorteil, dass Markierungen an das Element gebunden bleiben und mit diesem bewegt werden, wenn das betreffende Token verschoben wird.

Eingebettete Elemente enthalten neben ihrer graphischen Repräsentation und ihrer Benennung auch eine Referenz auf die Ressource, die durch sie repräsentiert wird. Ist diese Ressource ein Teilmodell, so wird dieses als gespeicherter Modellzustand referenziert und beim Abruf entsprechend am sekundären Ausgabekanal dargestellt. Sind externe Ressourcen (im Sinne von Dateien im Filesystem) angebunden, so werden diese beim Abruf mittels der durch die Java-Plattform zur Verfügung gestellten Methode `Desktop.open` geöffnet. Diese Methode greift dazu auf die Plattform-spezifischen Dateitypen-Bindungen zu und öffnet die jeweilige Datei mit der zugeordneten Standard-Applikation.

Assoziationen sind von der durch JHotDraw zur Verfügung gestellten Klasse `LineConnection` abgeleitet und werden durch deren Eigenschaften bei der Erstellung explizit an den Endpunkten an zwei `Figures` (in diesem Fall an `BasicBuildingBlocks`) gebunden. Der Verinder wird dadurch mit jeder Bewegung eines der Elemente mitgeführt und muss nicht manuell an den neuen Endpunkt verlegt werden. Methoden zur Anzeige Pfeilspitzen und Benennung können ebenfalls von der Klasse `LineConnection` übernommen werden.

Kalibrierung der Ausgabe

Der Projektor, der die Information auf die Tischoberfläche projiziert, ist aus Gründen der Transportierbarkeit nicht fix im Boden des Tisches eingebaut. Deswegen ist nach dem Aufbau der Systems im Allgemeinen die projizierte Version des Modells nicht deckungsgleich mit der Position der physischen Tokens. Es kann eine Verschiebung entlang beider Achsen auftreten, die alle zu den physischen Tokens ausgegebene Information von diesen absetzt. Dadurch kann die Information unter Umständen nicht mehr eindeutig zugeordnet werden.

Aus diesem Grund ist die Position der projizierten Modellversion einstellbar, die Kalibrierung muss einmalig nach dem Aufbau des Werkzeugs vorgenommen werden. Aufgrund der Verwendung des Umlenkspiegels kann es nicht nur zu einer Verschiebung der Information auf der X- oder Y-Achse kommen, es können auch Spreizungen oder Stauchungen auftreten, welche den Projektions-Fehler von links nach rechts bzw. von oben nach unten zunehmen lassen. Auch diese Spreizungen oder Stauchungen sind softwareseitig im Rahmen der Kalibrierung korrigierbar.

7.4.2. Ausgabe der Modellierungshistorie

Wird die Modellierungshistorie abgerufen, so wird am sekundären Ausgabekanal der zuletzt gespeicherte Modellzustand angezeigt. Am primären Ausgabekanal kommt weiterhin der aktuelle Modellzustand zur Anzeige. Die Modellierungshistorie wird in Form von **Snapshot**-Objekten in einem **History**-Objekt gespeichert. Das Objekt der Klasse **History** dient dabei der Verwaltung der gesamten Historie (also aller Snapshots) und ermöglicht die Ablage neuer Snapshots und Navigation durch diese sowie den Abruf der bereits gespeicherten Modellzustände. In den **Snapshot**-Objekten selbst ist neben dem eigentlichen Modellzustand ein Zeitstempel abgelegt, der den Zeitpunkt der Speicherung kennzeichnet. Außerdem gibt ein Flag an, ob die Speicherung automatisch erfolgte oder explizit ausgelöst wurde. Der eigentliche Modellzustand wird als Collection von **Figures** gespeichert. Da sowohl **BasicBuildingBlocks** als auch **Connections** von der Klasse **Figure** abgeleitet sind, können diese hier einheitlich verwaltet werden. Auch bei der Darstellung muss nicht zwischen den beiden Klassen unterschieden werden, da dem **DrawingView** (also der Zeichenoberfläche) zur Darstellung auch lediglich **Figures** übergeben werden müssen.

Bei der Navigation durch die Modellierungshistorie wird entsprechend der Drehbewegung des Kontroll-Tokens aus dem **History**-Objekt der vorhergehende bzw. nachfolgende **Snapshot** abgerufen. Der aktuell auf dem sekundären Ausgabekanal dargestellte Modellzustand wird gelöscht und durch die im **Snapshot** gespeicherten Modellelemente und Verbinder ersetzt. Zusätzlich wird der Zeitstempel als Text

dargestellt und am unteren Bildschirmrand der Fortschrittsbalken ausgegeben. Dieser zeigt an, wo in der gesamten Historie der aktuell angezeigte Modellzustand zu verorten ist und gibt Feedback darüber, ob eine weitere Navigation nach vor oder zurück möglich ist.

7.4.3. Umsetzung der Wiederherstellungsunterstützung

Wenn die Wiederherstellungsunterstützung ausgelöst wird, werden vor dem Start der Ausgabe der Wiederherstellungsanweisungen alle Verbinder ausgeblendet. Da die Benutzer lediglich die Element (bzw. die diese repräsentierenden Tokens) manipulieren muss, ist die Anzeige der Verbinder nicht notwendig bzw. sogar störend, da sich diese mit den auf der Oberfläche angezeigten Anweisungen zur Wiederherstellung überlappen würden.

Die eigentliche Wiederherstellungsunterstützung beginnt mit der Berechnung des ersten Schrittes, der von den Benutzern zu setzen ist. Dabei wird geprüft, ob im aktuell auf der Oberfläche befindlichen Modell ein Element enthalten ist, das im wiederherstellenden Zustand fehlt. Ist dies der Fall, wird dieses Element markiert und die Benutzer damit aufgefordert, es zu entfernen. Die so gestellte Aufgabe wird in einem Objekt der Klasse Task gespeichert. Bei jeder eintreffenden Meldung über Änderungen auf der Modellierungsoberfläche wird nun mit Hilfe der completed-Methode der Task-Objekts überprüft, ob der erste Schritt damit ausgeführt wurde (ob im konkreten Fall also das Element entfernt wurde). Ist dies der Fall, so wird die Methode zur Berechnung des nächsten Schrittes erneut aufgerufen und ein neues Task-Objekt erzeugt, das bei Änderung auf der Oberfläche wiederum zur Prüfung der Durchführung des Schrittes herangezogen wird. Sind noch Elemente vorhanden, die im Zielzustand nicht enthalten sind, ist der nächste Schritt wiederum eine „Entfernen“-Aufgabe.

Befinden sich keine überflüssigen Elemente mehr auf der Oberfläche, so wird die nächste Aufgaben-Kategorie – „Bewegen“ – geprüft. Hier werden all jene Elemente behandelt, die sowohl im Ausgangs- als auch im Zielzustand enthalten sind, sich jedoch auf unterschiedlichen Positionen befinden. Zur Anzeige dieser Art von Aufgaben wird ein Vektor berechnet, der auf den Ausgabekanälen angezeigt wird und von der aktuellen Position des zu bewegenden Elements auf die Zielposition weist, die zusätzlich mit einer Markierung gekennzeichnet wird. Auch in dieser Kategorie wird bei jeder Änderung auf der Modellierungsoberfläche geprüft, ob die Aufgabe bereits erfüllt ist.

Wurden auch alle verschobenen Elemente an die korrekte Position bewegt, wird die Prüfung der letzten Kategorie von Aufgaben – „Hinzufügen“ – durchgeführt. Hier werden Aufgaben für all jene Elemente definiert, die im Zielzustand enthalten

sind, im Ausgangszustand aber fehlen. Die Zielposition des hinzuzufügenden Tokens wird auf den Ausgabekanälen mittels einer Markierung und der Benennung des Elements angezeigt. Sind auch die Aufgaben dieser Kategorie abgeschlossen, werden die Verbinder des Zielzustandes eingeblendet (indem die gespeicherten Connector-Objekte der Zeichenoberfläche einfach hinzugefügt werden) und das System wieder in den Modellierungs-Modus versetzt.

Implementierungstechnisch ist hier noch auf die Entkopplung zwischen Identifikation des nächsten Schrittes und der Prüfung dessen Durchführung mittels Objekten der Klasse Task hinzuweisen. Dies ist notwendig, weil Wiederherstellungsaufgaben über mehrere Ereignisse auf der Modellierungsoberfläche (also das Verschieben, Hinzufügen oder Wegnehmen von Elementen) aktuell bleiben können und so der die Erfüllung der Aufgabe unabhängig von deren Erstellung geprüft werden muss. Dies hat mehrere Implikationen für die Implementierung.

Zum einen bringt die Klasse Task eine eigene Methode zur Prüfung der Erfüllung der in ihr gekapselten Aufgabe mit – Objekte von Task kennen also die Kriterien ihrer Erfüllung und können diese auch überprüfen. Zum anderen ist es nicht sinnvoll bzw. in machen Fällen nicht möglich, die durchzuführenden Aufgaben bei der Aktivierung des Wiederherstellungsmodus vorzuberechnen. Durch den Ablauf der Wiederherstellung kann der Ausgangszustand auch außerhalb der durch die Aufgaben vorgegebenen Modifikation verändert werden. Dadurch können vorberechnete Aufgaben obsolet bzw. falsch sein und können deshalb nicht sinnvoll zur Rekonstruktion eingesetzt werden. Tatsächlich wird nach jeder abgeschlossenen Aufgabe der nächste Schritt so bestimmt, als wäre der Wiederherstellungsmodus eben aktiviert worden. Der aktuelle Zustand auf der Modellierungsoberfläche wird als Ausgangspunkt herangezogen und wie oben beschrieben mit dem Zielzustand verglichen. So ist es auch möglich, dass auf eine Aufgabe der zweiten Kategorie ("Verschieben") wieder ein Aufgabe der ersten Kategorie ("Entfernen") folgt, falls das Modell im vorhergehenden Schritt entsprechend verändert wurde (also etwa ein nicht benötigtes Element hinzugefügt wurde). Erst wenn in allen drei Kategorien keine Differenzen zwischen aktuellem Zustand und Zielzustand mehr festgestellt werden, ist die Wiederherstellungsunterstützung abgeschlossen.

7.5. Zusammenfassung

In diesem Kapitel wurden alle Aspekte dargestellt, die in Zusammenhang mit der Ausgabe von Information durch das hier entwickelte Werkzeug stehen. Die Struktur des Kapitels wurde dabei analog zu jener des vorhergehenden Kapitels erstellt um eine konsistente Darstellung der technischen Aspekte des Systems zu gewährleisten. Im ersten Teil wurde auf Basis der in Kapitel XY definierten Anforderungen

abgeleitet, welche Arten von Information den Benutzern zur Unterstützung der Modellierung zur Verfügung gestellt werden müssen. Die dort identifizierten Aspekte bilden die Grundlage der Technologieentscheidung und der konkreten Umsetzung des Systems.

Im zweiten Abschnitt wurden die technologischen Grundlagen der Informationsausgabe behandelt. Nach der Identifikation der Kohärenz von Ein- und Ausgabekanälen als wesentliches Designkriterium bei Tangible User Interfaces wurden unterschiedliche technische Lösungen für kohärente und nicht kohärente Informationsausgabe beschrieben. Die Klassifikation der Ansätze erfolgt dabei auf Basis der von (Fishkin, 2004) vorgeschlagenen Taxonomie. Auf Basis der bereits vorhandenen technologischen Einschränkungen durch die Technologie-Entscheidung in Kapitel 6 wurde in der Folge für jede auszugebende Informationsart ein Zuordnung zu kohärenten oder nicht-kohärenten Ausgabeformen getroffen. Daraus resultiert die Entscheidung, mehr als einen Ausgabekanal zu verwenden, da eine durchgängig kohärente Ausgabe nicht möglich ist. Für die Umsetzung der Ausgabekanäle, die auf visuellem Weg realisiert werden, wurden geeignete Software-Frameworks beschrieben und einander gegenübergestellt. Auf Basis der festgelegten funktionalen und nicht-funktionalen Anforderungen kommen die Frameworks JHotDraw (Gamma und Eggenschwiler, 1996) und GEF (Moore et al., 2004) in Frage, von denen für die Erstellung des Prototypen JHotDraw als das System mit der fokussierteren Funktionalität und geringerem Einarbeitungsaufwand ausgewählt wurde. Im Rahmen einer Diplomarbeit (Feiner, 2008) wurde jedoch daneben ein auf GEF basierender Ausgabekanal implementiert.

Der dritte Abschnitt widmet sich auf Basis der auszugebenden Information und der zuvor getroffenen Technologieentscheidung der konzeptionellen Architektur der Ausgabekanäle. Hier wurde festgelegt, welche Information in welcher Form auf welchen Ausgabekanal dargestellt wird. Dabei wurde zur besseren Strukturierung in Ausgaben unterschieden, die dem Modell selbst zuzuordnen sind und in solche, die der Unterstützung der Modellbildung dienen.

Die Beschreibung der konkreten Umsetzung der Ausgabekanäle ist Gegenstand des vierten Abschnitts. Anhand eines Klassendiagramms der konkreten Umsetzung wurden die Zusammenhänge zwischen den Implementierungen der Ausgabekanäle gezeigt, die so ausgelegt sind, dass Änderungen mit minimalem Aufwand an nur einer Stelle für beide Ausgabekanäle durchgeführt werden können. Außerdem wurde beschrieben, wie die Anbindung entfernter Ausgabekanäle realisiert wurde. Für jede Art von auszugebender Information (Modellzustand, Modellierungshistorie und Wiederherstellungsunterstützung) ist außerdem angeführt, welche Abläufe innerhalb des Systems letztendlich zur Ausgabe der korrekten Visualisierung führt.

7. Ausgabe

Mit der Beschreibung der Softwarekomponenten, die für die Ausgabe der Information sorgen, ist die eigentliche Benutzungsschnittstelle vollständig definiert. Auf Basis dieses und des vorhergehenden Kapitels wird in Kapitel 9 das System in die in Kapitel 5 beschriebenen Erklärungs- und Strukturierungsansätze für Tangible Interfaces eingeordnet. Ein weiterer Aspekt der Implementierung, der in dieser Arbeit zu berücksichtigen ist, ist die Persistierung der Modell-Information in einer Form, die eine vollständige Abbildung der Semantik der Modelle und eine Weiterverarbeitung durch externe Systeme erlaubt. Dieser Aspekt ist Gegenstand des folgenden Kapitels.

8. Persistierung

In den vorangegangen drei Kapiteln wurde die Umsetzung des eigentlichen Werkzeugs beschrieben. Neben der Unterstützung des Modellierungsvorgangs ist aber auch die persistente Speicherung der erstellten Modelle zum Zwecke der Weiterverarbeitung ein hier zu beleuchtender Aspekt. Auf die Persistierung wirken vor allem zwei der in Kapitel XY identifizierten Anforderungen ein. Zum ersten ist die Nachvollziehbarkeit des Modellierungsvorganges sicherzustellen – dies gilt nicht nur während des Vorgangs selbst, sondern auch danach. Dementsprechend ist sämtliche Information zu persistieren, die zur Wiederherstellung nicht nur des Modells selbst sondern auch der gesamten Modellierungshistorie notwendig ist. Zum zweiten hat die Forderung nach semantischer Offenheit bei der Modellierung auch unmittelbare Auswirkungen auf die Persistierung. Neben dem Modell selbst muss aufgrund dieser Anforderung auch die Bedeutung der verwendeten Modellierungselemente miterfasst undpersistiert werden, so dass diese bei der Weiterverarbeitung der Modelle verwendet werden kann.

In diesem Kapitel werden nun aufgrund der eben genannten Forderungen technologische Ansätze identifiziert, beschrieben und schließlich hinsichtlich ihrer Eignung für den konkreten Einsatz beurteilt. Der ausgewählte Ansatz wird im darauf folgenden Abschnitt konzeptionell beschrieben. Die Abbildung der Modelle und der ebenfalls zu persistierenden zusätzlichen Information in ein geeignetes Datenmodell ist Gegenstand des darauf folgenden Abschnitts. Schließlich wird die konkrete technische Umsetzung der Persistierung dargelegt und die dazu notwendigen Software-Module im Detail beschrieben.

8.1. Möglichkeiten der Persistenzsicherung

- Serialisierung von Java-Objekten
- Relationale Datenbanken
- XML¹ Topic Maps

¹Extensible Markup Language

8.2. Topic Maps

Topic Maps (ISO JTC1/SC34/WG3, 2008) sind wie bereits in Abschnitt XY beschrieben ein Mittel zur Abbildung von semantischen Netzen. In Topic Maps können beliebige Daten strukturiert aufbereitet und zueinander in Beziehung gesetzt werden. Die Art der zu repräsentierenden Daten ist dabei irrelevant, eine Topic Map trifft keine Aussage über ein den repräsentierten Daten zugrundeliegendes Begriffssystem (sie ist „ontology-agnostic“ (Vatant, 2004)).

Historisch stammen Topic Maps aus dem Bereich der technischen Repräsentation von Thesauri und Indizes (Pepper, 2000) (Rath, 2003). Aus diesen Bereichen motivieren sich auch die Bausteine einer Topic Map, wenngleich der Verwendung durch diesen Ursprung nicht eingeschränkt wird. Die grundlegenden Elemente einer Topic Map sind „Topics“, „Associations“ und „Occurrences“ (siehe Abbildung 8.1).

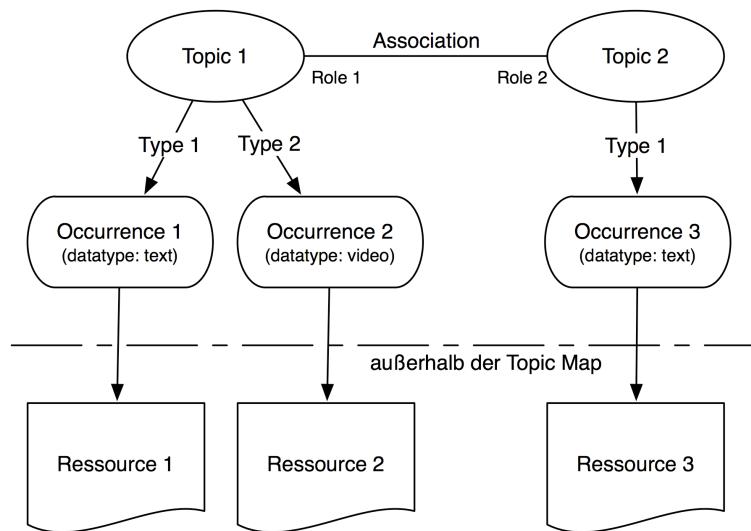


Abbildung 8.1.: Grundlegende Elemente einer Topic Map

„Topics“ sind Begriffe dar und bilden die Knoten des semantischen Netzes. Ein Topic kann beliebige Information darstellen, repräsentiert aber immer genau ein Phänomen der realen Welt (d.h. zu einem Topic muss es eine Entsprechung außerhalb der Topic Maps geben, die beobachtbar oder beschreibbar ist und auf die die modellierende Person Bezug nehmen will²). Eine Topic Map ist damit im Sinne

² „A subject can be anything whatsoever, regardless of whether it exists or has any other specific characteristics, about which anything whatsoever may be asserted by any means whatsoever. In particular, it is anything about which the creator of a topic map chooses to discourse.“ (ISO JTC1/SC34/WG3, 2008, S.8)

von Stachowiak (1973) ein diagrammatisches Modell, das einen bestimmten, für den Modellersteller relevanten Ausschnitt der Realität abbildet.

“Associations“ bilden die Beziehungen zwischen Topics ab und stellen damit die Kanten des semantischen Netzes dar. Eine Association verknüpft Topics semantisch miteinander und kann frei mit Bedeutung belegt werden. Die Art der Beziehungen ist also nicht festgelegt und wird wie die Bedeutung der Topics frei gewählt werden. Topics und Associations decken historisch den Bereich der Darstellung von Thesauri ab, in denen Begriffe definiert und zueinander in Beziehung gesetzt werden.

Der zweite historische Ursprung von Topic Maps, die Indizes, werden durch das Konstrukt der „Occurrences“ abgedeckt. Occurrences (“Auftreten“) sind Referenzen aus der Topic Map in die reale Welt. Sie setzen die Topics einer Topic Map in Bezug zu beliebiger referenzierbarer Information (z.B. Dokumente). Im Kontext der eben genannten Indizes, kann eine Topic Map als der mit Querverweisen versehene Index eines Buches verstanden werden, in dem durch die Angabe von Seitenzahlen auf den Text des Buches verwiesen wird. Diese Verweise durch Angabe der Seitenzahlen sind in diesem Zusammenhang die Occurrences.

Die Ansammlung von durch Associations verknüpften und mit Occurrences versehenen Topics bilden eine Topic Map. Darüber hinaus kann in Topic Maps jedoch noch weiterführende Information repräsentiert werden (siehe Abbildung 8.2), die Gegenstand der folgenden Abschnitte sein werden.

8.2.1. Topics, Subjects, Topic Names und Variants

Wie oben bereits beschrieben, repräsentiert ein Topic ein Phänomen der realen Welt in einer Topic Map. Dieses Phänomen der realen Welt, das durch das Topic repräsentiert wird, wird als „Subject“ bezeichnet. In einer Topic Map darf es zu einem Subject nur exakt ein Topic geben, umgekehrt kann ein Topic auch nicht mehrere Subjects repräsentieren, die Zuordnung zwischen Subject und Topic ist also eindeutig (bijektiv). Im Topic wird dazu exakt ein „Subject Identifier“ registriert, der auf eine Informationsressource verweist, die das Subject für Menschen eindeutig identifizierbar macht (diese Ressource wird als „Subject Indicator“ bezeichnet). Zusätzlich kann ein „Subject Locator“ angegeben werden, der auf das tatsächlich in der realen Welt vorhandene Subject verweist. In Abgrenzung dazu kann es bei der anderen Brücke zwischen realer Welt und Topic Map, den Occurrences, für jeder Topic beliebig viele Zuordnungen geben. Eine Occurrence referenziert auch auf die reale Welt, zeigt aber dort nicht auf das Subject selbst, sondern auf ein dieses Subject beschreibendes Objekt in der realen Welt. Beispielhaft ist dazu in Abbildung 8.3 dieser Zusammenhang anhand des Topics „Tasse“ dargestellt. Ein anderes Beispiel ist ein Topic „London“, das als Subject die reale Stadt London repräsentiert und dem eine

8. Persistierung

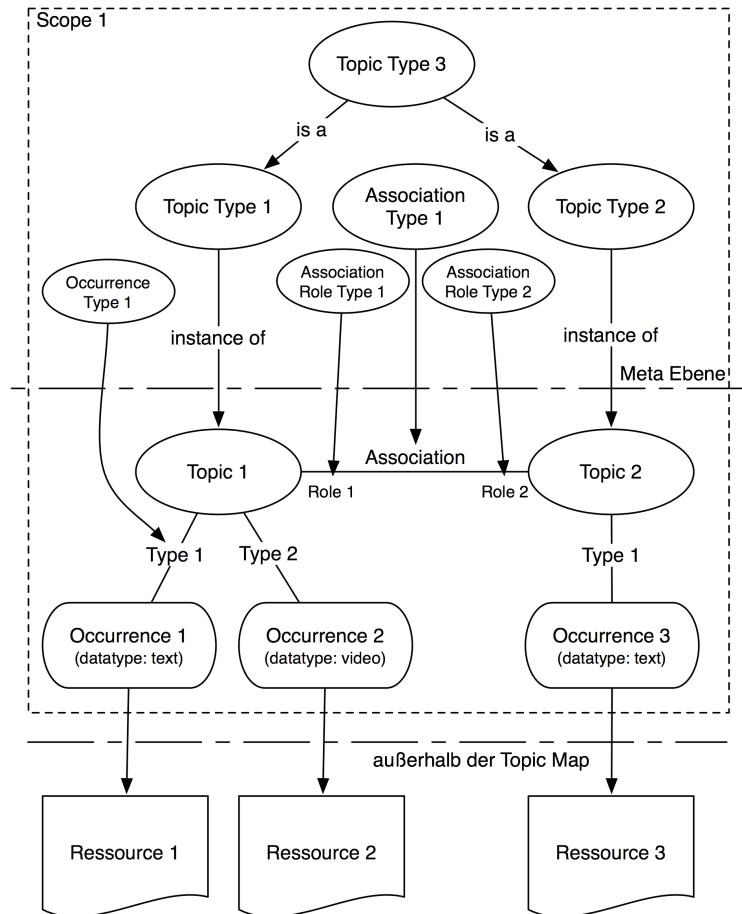


Abbildung 8.2.: Umfassende Darstellung der Elemente einer Topic Map

Occurrence zugeordnet werden könnte, die auf eine Landkarte (als in der Realität vorhandene Beschreibung der realen Stadt London) referenziert.

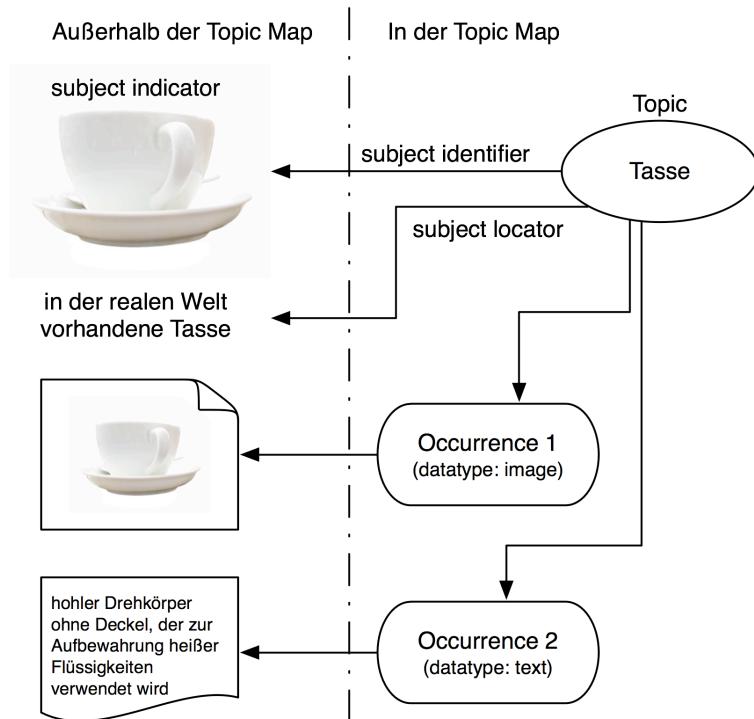


Abbildung 8.3.: Abgrenzung zwischen Subject und Occurrence in Topic Maps

Bislang wurde vereinfacht ein Topic immer mit einem direkt zugeordneten Namen dargestellt. In einer Topic Map besitzt ein Topic jedoch keinen eindeutigen Namen. Es wird vielmehr durch seinen Subject Identifier eindeutig gekennzeichnet. Dieser ist jedoch nicht unbedingt für Menschen les- und/oder interpretierbar – der Subject Identifier hat das Ziel, ein Subject für die Verarbeitung durch Software eindeutig zuordnenbar zu machen. Für die Bezeichnung eines Topics in einer für Menschen interpretierbaren Form ist die Verwendung von „Topic Names“ vorgesehen (siehe Abbildung 8.4). Topic Names werden immer textuell angegeben und beschreiben das Subject, das durch das betreffende Topic referenziert wird. Durch einen Topic Name soll das Subject für Menschen erkennbar sein, wobei die Zuordnung nicht notwendigerweise eindeutig sein muss (Beispiel: der Topic Name „Jaguar“ kann ein Fahrzeug oder eine Großkatze bezeichnen und ist dementsprechend ein zulässiger Name für zwei unterschiedliche Topics). Einem Topic können beliebig viele Topic Names zugewiesen werden – es ist so zum Beispiel möglich, eine mehrsprachige Topic Map zu realisieren, in der zu jedem Topic Topic Names in unterschiedlichen Sprachen angegeben werden.

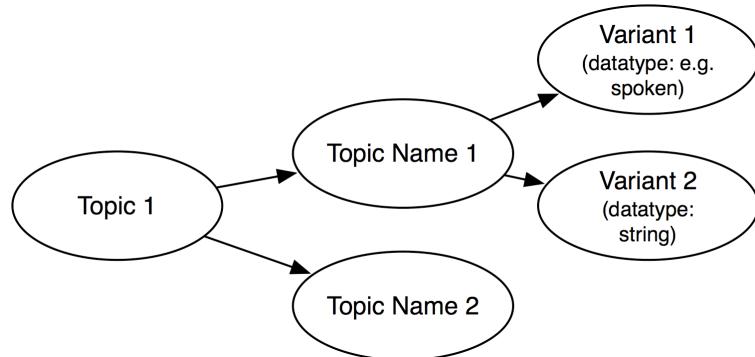


Abbildung 8.4.: Benennung von Topics

Als weitere Detaillierungsstufe können zu jedem Topic Name „Variants“ angegeben werden. Wie durch den Namen angedeutet, handelt es sich dabei um Varianten eines Topic Names, die in bestimmten Zusammenhängen oder für gewisse Anwendungszwecke besser geeignet sein können als der eigentlich Topic Name. Ein Beispiel für eine mögliche Variante ist die Angabe einer gesprochenen Version des Topic Names. Ein weiteres Anwendungsgebiet für Varianten ist die im Standard explizit vorgesehene Angabe eines „Sort Name“ (ISO JTC1/SC34/WG3, 2008, S. 18), der es erlauben soll, Topic Maps in eine durch diesen Namen vorgegebene Ordnung bringen zu können. Varianten werden durch die Angabe eines dezidiert dafür gewidmeten Topics in deren Scope (siehe Abschnitt 8.2.5) als Sort Names gekennzeichnet.

8.2.2. Associations und Roles

„Associations“ stellen Verbindungen zwischen den einzelnen Topics einer Topic Map her. Associations haben beliebig viele Endpunkte, mindestens jedoch einen (sind also nicht von vorneherein immer binär sondern können auch unär sein oder mehr als zwei Topics verknüpfen). Eine Associations enthält wie ein Topic nicht unmittelbar einen für Menschen lesbaren Namen. Diese wird durch ein Topic festgelegt, das die Kategorie festlegt, der die Association zuzuordnen sind (siehe Abschnitt 8.2.4). Diesem Topic kann wiederum mindestens ein Topic Name zugeordnet werden, welcher letztendlich die Benennung der Association festlegt.

Associations werden jedoch nicht direkt mit Topics verknüpft. Um ausdrucksstärkere Verknüpfungen realisieren zu können, agieren „Roles“ als Verknüpfung zwischen Association und den betreffenden Topics. Roles legen die „Rolle“ – also die Bedeutung – eines Topics in exakt der betrachteten Association fest. Diese Bedeutung kann generisch sein und zum Beispiel dazu verwendet werden, die per se ungerichteten Associations unabhängig von ihrer konkreten Bedeutung mit einer Richtung

zu verstehen (zum Beispiel durch die Zuordnung von Roles „Anfang“ und „Ende“) aber auch um die Beziehung semantisch anzureichern (zum Beispiel durch die Zuordnung von Roles „Verantwortlicher“, „Ausführender“ und „Prozessschritt“ in einer Association „durchzuführen“). Die Anzahl der in einer Association referenzierten Roles gibt damit auch die Kardinalität der Association (also die Anzahl ihrer Endpunkte) an. Aus den konkreten Roles wird dann auf die Topics verwiesen, die diese Roles einnehmen bzw. „spielen“ (tatsächlich heißt die betreffende Eigenschaft einer Role „player“). Genau wie Associations werden Roles nicht direkt benannt sondern über ein Topic, das ihre Kategorie bestimmt, mit einer Benennung versehen (Detail wiederum in Abschnitt 8.2.4).

8.2.3. Occurrences und Datatypes

Wie zu Beginn dieses Abschnitts bereits beschrieben und in den Erläuterungen zur Thematik der Subjects (siehe Abschnitt 8.2.1) angedeutet, bilden „Occurrences“ die Brücke aus der Topic Map in die reale Welt, indem sie auf Ressourcen referenzieren, die in einem beliebigen Zusammenhang mit den jeweiligen Topic stehen. Ein Topic kann beliebig viele Occurrences haben. Anders als bei Associations existieren für Occurrences keine Roles (was auch nur bedingt sinnvoll wäre, da jede Occurrence nur zu exakt einem Topic gehören kann). Die Bedeutung der Occurrence für das Topic kann wie bei Associations über die Kategorie der Occurrence festgelegt werden, die wiederum durch ein separates Topic repräsentiert wird (siehe dazu auch Abschnitt 8.2.4). Beispielsweise kann eine Occurrence zur Kategorie „Karte“ gehören und so angeben, dass die so klassifizierte Occurrence zum Topic „London“ auf eine Karte des Stadtgebiets verweist.

Zusätzlich zu der Kategorie wird in einer Occurrence auch der Datentyp der Information angegeben, in dem die referenzierte Information vorliegt. Dabei können beliebige URI (Uniform Resource Identifiers³) verwendet werden. Da URIs beliebigen Inhalt haben können, wäre es in obigen Beispiel möglich durch den Datentyp einer Occurrence festzulegen, ob es sich bei der Karte um eine Rastergrafik oder eine Vektorgrafik handelt und so Information über deren möglich Einsatzgebiete zu einzubetten.

8.2.4. Metamodellierung in Topic Maps

Wie oben bereits mehrmals angedeutet kann in einer Topic Map neben den eigentlichen zu repräsentierenden Informationen (Topics, Associations, Roles und Occurrences) auch Information über die Topic Map selbst eingebettet werden (neben

³wie in RFC 3986 definiert und unter <http://www.ietf.org/rfc/rfc3986.txt> abzurufen

dem Model also auch das Meta-Modell abgebildet werden kann). Die Information umfasst Angaben über die in der jeweiligen Topic Map existierenden Kategorien von Topics, Topic Names, Associations, Roles und Occurrences. Hinsichtlich der Repräsentation dieser Information sind zwei Ansätze zu unterscheiden, von denen der erste bei Kategorieangaben von Topics zum Einsatz kommt, der andere bei Kategorieangaben jeder anderen Art von Information. Allen Kategorien ist gemein, dass sie selbst wiederum als Topics repräsentiert werden und auch als solche verwendet werden können. Es ist also möglich, zu einem Topic, das als Kategorie verwendet wird, selbst wiederum eine Kategorie anzugeben, wodurch die Einführung beliebig vieler Meta-Ebenen möglich ist. Außerdem können als Kategorien verwendete Topics ebenfalls wieder mit Associations verknüpft und mit Occurrences versehen werden. Hinsichtlich der Nomenklatur ist noch darauf hinzuweisen, dass Kategorien im Allgemeinen als „Types“ bezeichnet werden, man also von „Topic Types“, „Topic Name Types“, „Association Types“, „Role Types“ und „Occurrence Types“ spricht.

Topic Types

Topic Types werden in einer Topic Map durch ein spezielle, im Standard festgelegte Association definiert. Soll einem Topic ein Type zugewiesen werden, muss eine Association der Kategorie „type-instance“ eingefügt werden, bei der das Topic selbst die Role „instance“ einnimmt und dem Topic, das die Kategorie repräsentiert, die Role „type“ zugewiesen wird. Diese Beziehung entspricht einer Konkretisierung einer (abstrakten) Kategorie oder Klasse von Topics auf eine bestimmte Instanz, die die Merkmale dieser Klasse trägt. In Abbildung 8.5 besteht eine type-instance-Beziehung (dort als „instance-of“ bezeichnet) zwischen der Kategorie „VW Golf“ und der konkreten Instanz „SR-174 AU“ (also einem Topic, bei dem das amtliche Kennzeichen als Topic Name verwendet wurde). „VW Golf“ fungiert hier also als Topic Type, wobei es selbst ein Topic ist, das sich durch nichts als die eingenommene „type“-Rolle von einem anderen Topic unterscheidet und dementsprechend behandelt werden kann.

Einem Topic können beliebig viele Topic Types zugewiesen werden, indem es in mehr als einer Association die Role „instance“ einnimmt. Es wird so als mehreren Kategorien zugehörig gekennzeichnet. Umgekehrt kann ein Topic Type mehr als einem Topic zugewiesen werden, indem das betreffende den Topic Type repräsentierende Topic die Role „type“ mehrfach einnimmt.

Auch ist es möglich, Hierarchien von Types zu bilden, in dem einem Topic, das als Topic Type fungiert, selbst wieder ein Topic Type zugewiesen wird. Dieses Konstrukt ist jedoch mit Vorsicht zu gebrauchen, da zur Abbildung der Struktur einer Domäne ein semantisch ähnliches, bei näherer Betrachtung aber eine unterschiedliche Bedeutung tragendes Konstrukt zum Einsatz kommt. Grundsätzlich muss un-

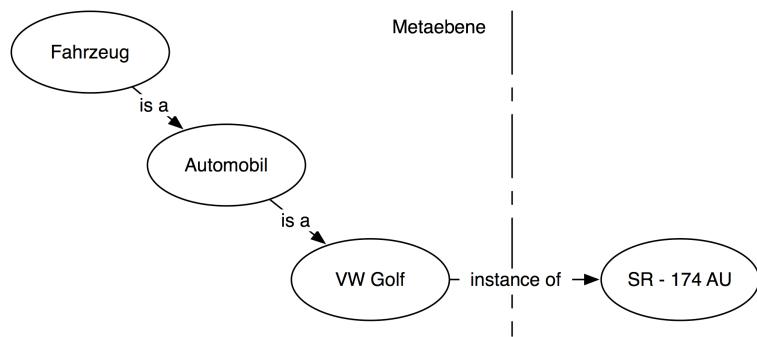


Abbildung 8.5.: Beziehungen in der Metamodellbildung in Topic Maps

terschieden werden, ob ein Topic eine konkrete Instanz eines anderen ist oder lediglich eine Spezialisierung darstellt. Im ersten Fall kommt eine „type-instance“ zum Einsatz, das übergeordnete Topic befindet sich semantisch auf einer anderen, abstrakteren Ebene und stellt eine Kategorie (also einen Topic Type) dar. Im Falle einer Spezialisierung kommt eine „supertype-subtype“-Association zum Einsatz, deren Rollen „supertype“ vom übergeordneten, allgemeineren bzw. „subtype“ vom untergeordneten, spezielleren Topic eingenommen wird. Hier befinden sich beide Topics semantisch auf einer Ebene, keines ist abstrakter als das andere. Der Unterschied liegt vielmehr in der mehr oder weniger konkreten Festlegung der durch die Topics repräsentierten Subjects. So ist wie in Abbildung 8.5 dargestellt das Topic „VW Golf“ ein Subtype des Topics „Automobil“ welches wiederum ein Subtype des Topics „Fahrzeug“ ist. Hier ist erkennbar, dass „Automobil“ insofern eine Spezialisierung von „Fahrzeug“ ist, als dass es im Allgemeinen motorisiert ist und vier Räder besitzt. „VW Golf“ ist wiederum eine Spezialisierung von Fahrzeug, die hinsichtlich der Form der Karosserie, der Anzahl der Türen und anderen Merkmalen mehr spezialisiert. Zwischen keinem der Topics findet jedoch eine Konkretisierung in dem Sinne statt, als dass auf der untergeordneten Seite von einem konkreten, real existierenden Fahrzeug die Rede wäre – dazu ist die „type-instance“-Beziehung zu verwenden. Die „subtype-supertype“-Beziehung ist transitiv, d.h. dass ein „VW Golf“ nicht nur ein „Automobil“ ist, sondern auch ein „Fahrzeug“. Für „type-instance“-Beziehungen ist diese Eigenschaft nicht gegeben.

Andere Types

Bei allen anderen Types (konkret also Topic Name Types, Association Types, Role Types und Occurrence Types) wird die Kategorie nicht durch eine separate Association abgebildet sondern durch eine im jeweiligen Informationselement enthaltene Referenz auf ein Topic, das als Type fungiert. Die Darstellung der Kategorie-

Information ist damit nicht so explizit wie bei Topic Types, wo sich direkt in der Repräsentation der eigentlichen Nutzinformation niederschlägt. Auch semantisch ist sind die hier behandelten Types gegenüber Topic Types insofern eingeschränkt, als das jedem Element exakt ein Type zugeordnet sein muss (der Type kann also nicht leer sein, auch können nicht mehrere Types zugeordnet werden). Wie oben bereits erwähnt sind Topic Types hier flexibler, einem Topic können beliebig viele oder auch keine Topic Types zugeordnet werden. Dies ist aber nur vordergründig eine Einschränkung. Topics dürfen wie oben beschrieben für jedes Subjekt nur einmal existieren. Hat aber ein Subject und damit ein Topic in unterschiedlichen Domänen unterschiedliche Bedeutungen, muss dies über mehrere Topic Types (in Verbindung mit Scopes, siehe Abschnitt 8.2.5) abgebildet werden. Alle anderen Informationskategorien in der Topic Map unterliegen nicht dieser Eineindeutigkeitsregel und können bzw. müssen, sollten sie unterschiedlichen Kategorien zuzuordnen sein, auch mehrfach vorhanden sein. Eine Assoziation, die einen anderen Namen trägt (also einer anderen Kategorie angehört) ist beispielsweise nicht identisch mit der ursprünglichen Assoziation, deren Name ebenfalls bereits durch die Zuordnung zu einer Kategorie festgelegt wurde.

Modellieren von Einschränkungen

Der Topic Map Standard erlaubt zwar die Angabe von Metamodellelementen (Types), ermöglicht es aber nicht Regeln anzugeben, anhand derer der semantisch korrekte Aufbau einer Topic Map geprüft werden. Ed ist beispielsweise möglich, einen Association-Type „hat Mitglieder“ zu definieren, der mittels den Roles „Organisationseinheit“ und „Mitarbeiter“ die Zuordnung von Mitarbeitern zu den Organisationseinheiten eines Unternehmens zuzuordnen. Es ist jedoch in der Topic Map nicht möglich zu spezifizieren, dass beispielsweise mindestens drei Mitarbeiter zugeordnet werden müssen oder dass es in dieser Beziehung nur eine Organisationseinheit geben darf. Weiters kann nicht spezifiziert werden, durch Topics welchen Types die jeweiligen Roles eingenommen werden dürfen – beispielsweise ist eine Zuordnung von Produktionsmitteln in der Role „Mitarbeiter“ zulässig bzw. kann sie nicht als unzulässig gekennzeichnet werden.

Ist eine derartige semantische Einschränkung bei der Topic Map Erstellung und die Einführung verbindlicher Strukturvorgaben notwendig, so muss dies außerhalb der Topic Map oder durch externe Interpretation spezifischer Topic Map Elemente geschehen. In ersterem Fall kann die noch nicht in finalem Zustand vorliegende TM-CL (Topic Map Constraint Language, (ISO JTC1/SC34, 2008)), eine Regelsprache zur Einschränkung der Repräsentationsmöglichkeiten in einer Topic Map, verwendet werden. Im zweiten Fall können die Metamodell-Elemente (also alle Topics, die als Types verwendet werden) durch zusätzliche Associations verknüpft werden, die

semantisch so interpretiert werden, dass sie eine zulässige Kombination von Topics der jeweiligen Kategorie anzeigen.

8.2.5. Statements und Scopes

Topic Maps bieten die Möglichkeit, Gültigkeitsbereiche für die in ihnen abgebildeten Informationen zu spezifizieren. Ein Gültigkeitsbereich definiert, in welchem Kontext eine Information gültig ist. Außerhalb dieses Kontext kann über die Gültigkeit keine Aussage getroffen werden. Der Gültigkeitsbereich wird als „Scope“ bezeichnet. Ein Scope kann für jedes „Statement“ in der Topic Map gesetzt werden. Statements sind alle „Aussagen“ über Topics, die in der Topic Map abgebildet werden, nicht aber Topics selbst. Als Statement werden Topic Names, Associations und Occurrences betrachtet. Roles und Variants besitzen keinen Scope, da sie keine direkte Aussage über Topics treffen, sondern nur im Zusammenhang mit Associations bzw. Topic Names existieren, deren Scope sich quasi auf sie vererbt.

Ein Scope für ein Statement wird durch die Angabe eines oder mehrerer Topics festgelegt. Wird kein Topic angegeben, so gilt der „unconstrained scope“, das Statement ist unbeschränkt gültig. Topics zur Abbildung von Scopes können explizit angelegt werden (z.B. Topics „Deutsch“ und „Englisch“, die Sprach-Scopes ermöglichen, Topics „Tierwelt“ und „Transportwesen“ zur Domänenabgrenzung – siehe Abbildung 8.6), es ist jedoch grundsätzlich auch möglich, die Gesamtheit der Topics einer Domäne zur Definition eines betreffenden Scopes heranzuziehen (also alle in der Topic Map vorhandenen Topics, die Tiere repräsentieren, als Scope zu verwenden, um den Gültigkeitsbereich „Tierwelt“ abzubilden). Obwohl der Topic Map Standard hier explizit offen bleibt, erscheint jedoch erstere Variante hinsichtlich der Verwaltbarkeit aber auch der semantischen Vollständigkeit wegen als ratsamer (das Konzept „Tierwelt“ käme ansonsten z.B. nicht notwendigerweise vor, sondern wäre nur implizit vorhanden, was eine Auswertung schwierig macht).

Wird mehr als ein Topic angegeben, so bilden alle Topics gemeinsam den Kontext, indem das Statement gültig ist. Bei der Auswertung des Gültigkeitsbereichs müssen also alle angegebenen Topics zutreffen, damit ein Statement gültig ist. Ein Statement dessen Scope die Topics „Tierwelt“ und „Deutsch“ enthält, ist also nur gültig, wenn beide Topics zutreffen (also z.B. zur Filterung ausgewählt wurden). Die gültigen Statements sind also die Schnittmenge jener der Statements, die im Scope „Tierwelt“ und im Scope „Deutsch“ gültig sind (siehe Abbildung 8.6). Die Angabe eines Scopes zu einem Statement, das in beiden Scopes (unabhängig voneinander gesehen) gültig sein soll, ist nicht direkt möglich. In diesem Fall muss das Statement zweimal jeweils unter Angabe eines der beiden Scopes eingefügt werden.

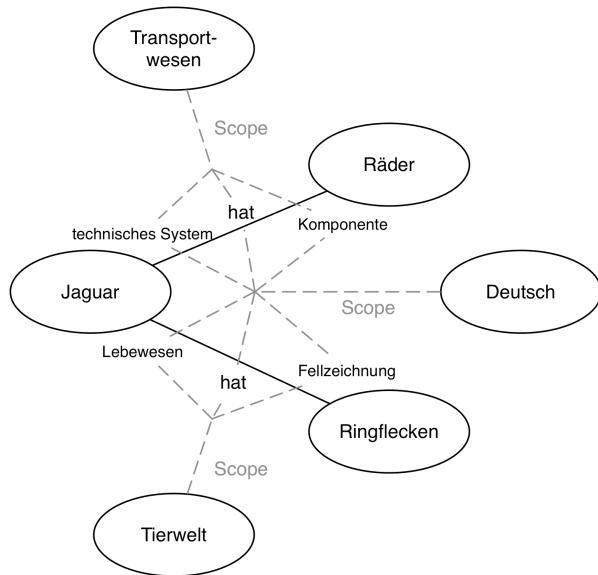


Abbildung 8.6.: Abbildung von Gültigkeitsbereichen durch Scopes

Für Topics selbst kann kein Scope angegeben werden. Das ist hinsichtlich des Topic Maps zugrunde liegenden Konzepts auch nicht sinnvoll, da Topics immer ein Phänomen der realen Welt, das Subject, repräsentieren, das selbst immer vorhanden ist und dessen Gültigkeit bzw. Existenz nicht von anderen Rahmenbedingungen wie anderen Subjects abhängt. Dementsprechend existiert ein Topic immer, lediglich seine Bezeichnung, die Beziehungen zu anderen Topics oder seine Occurrences können abhängig vom einem Scope unterschiedlich sein.

8.2.6. Reification

Wie bereits zu Beginn angeführt, können in einer Topic Map beliebige Inhalte abgebildet werden, jegliche Phänomene der realen Welt können durch Topics repräsentiert werden. Konsequenterweise können nun auch Elemente der Topic Map selbst oder sogar die Topic Map ansich durch ein Topic dargestellt werden. Eine derartige selbstbezügliche Abbildung wird als „Reification“ bezeichnet. Der Ausgangspunkt für eine Reification kann ein Statement sein oder eine Topic Map selbst. Topics können nicht verwendet werden, da ein sie repräsentierendes Topic semantisch äquivalent mit dem Ausgangspunkt wäre und damit zwei Topics vorhanden wären, die das gleiche Subject referenzieren. Nachdem die Abbildung zwischen Subject und Topic eineindeutig sein muss, ist ein derartiges Konstrukt nicht erlaubt.

In Abgrenzung zur Types (Association Types, Occurrence Types,...) repräsentiert ein reifizierendes Topic nicht die Kategorie eines Elements sondern das konkrete Element selbst. Es ist so möglich, einem konkreten Element wie einer Association zusätzliche Information hinzuzufügen (etwa eine Occurrence) oder auch eine bestehende Topic Map als Ganzes zu kapseln und durch das sie reifizierende Topic Bezug zu nehmen.

8.2.7. Merging

Unter „Merging“ versteht man die Vereinigung zweier voneinander getrennter Topic Maps zu einer gemeinsamen Map. Dabei ist vor allem die Eineindeutigkeitsregel zu beachten, für ein Subject darf also in der resultierenden Topic Map nur ein Topic existieren. Strukturell nicht kritisch, jedoch die Verwendbarkeit einschränkend sind mehrfach auftretende semantisch identische Statements. Soweit möglich, sollten auch diese Duplikate im Merging-Prozess entfernt werden.

Der Topic Map Standard definiert für jede Art von Element Regeln, anhand der festgestellt werden kann, ob zwei Elemente dieser Art identisch sind oder nicht. Ausgangspunkt sind immer die Topics, wobei beim Vergleich ausschließlich vom abgebildeten Subject ausgegangen werden muss. Sind die Subjects identisch, sind im Wesentlichen auch die beiden Topics identisch und können durch eine gemeinsame Topic ersetzt werden. Auf Basis der vereinigten Topic Menge werden nun die enthaltenen Statements verglichen. Damit Statements als identisch erkannt werden, müssen nicht nur ihr eigentlicher Inhalt sondern auch ihre Kategorie (Type), ihr Gültigkeitsbereich (Scope) und das/die ihnen zugeordnete(n) anderen Element(e) identisch sein. Eine Role ist also nur dann mit einer anderen Role identisch, wenn ihr Type, ihr Scope, die Association, der sie angehört und das referenzierte Topic identisch ist. Daraus folgt, das in der Merging-Reihenfolge zuerst die Topics, dann die eigenständigen Statements wie Topic Names, Associations und Occurrences und letztendlich die abgängigen Statements wie Roles und Variants behandelt werden müssen.

8.3. Abbildung von Modellen auf Topic Maps

Diagrammatische Modelle (Oppl und Stary, 2005) können direkt ohne zusätzliche Transformationen auf Topic Maps abgebildet werden. Derartige Modelle bestehen aus Knoten und Kanten, die Verwendung dieser im konkreten Anwendungskontext ist durch die Modellierungssprache festgelegt. In den folgenden Abschnitten wird nun beschrieben, wie die einzelnen Aspekte eines diagrammatischen Modells auf eine Topic Map abgebildet werden können.

8.3.1. Grundlegende Abbildung

Der nahe liegendste Ansatz, um diagrammatische Modelle auf Topic Maps abzubilden, werden nun die Knoten auf Topics, die Kanten auf Associations abzubilden (siehe Abbildung 8.7, Variante 1). Ist in den Kanten mehr Information als die bloße Anzeige der Verbindung von zwei Knoten abgebildet (wie z.B. in UML⁴ State Charts (Rumbaugh et al., 2004), bei denen die zu einem Zustandsübergang führenden Ereignisse inkl. Bedingungen und ablaufende Aktionen in den Kanten repräsentiert werden), so kann es sinnvoll sein, auch die Kanten auf Topics abzubilden, da diesen auf dem Wege der Occurrences zusätzliche Information zugewiesen werden kann (siehe Abbildung 8.7, Variante 2). Associations würden in diesem Fall verwendet, um die Knoten und Kanten des Modells zu verbinden, spielen also in der Repräsentation der Information nur noch eine untergeordnete Rolle. Ein Weg, die direkte Abbildung beizubehalten (indem Knoten auf Topics und Kanten auf Associations abgebildet werden) ist, zur Repräsentation der zusätzlich in den Kanten liegenden Information die Möglichkeit der Reification zu nutzen, den Associations also Topics zuzuweisen, die genutzt werden können, um diese zusätzliche Information zu verwalten (siehe Abbildung 8.7, Variante 3).

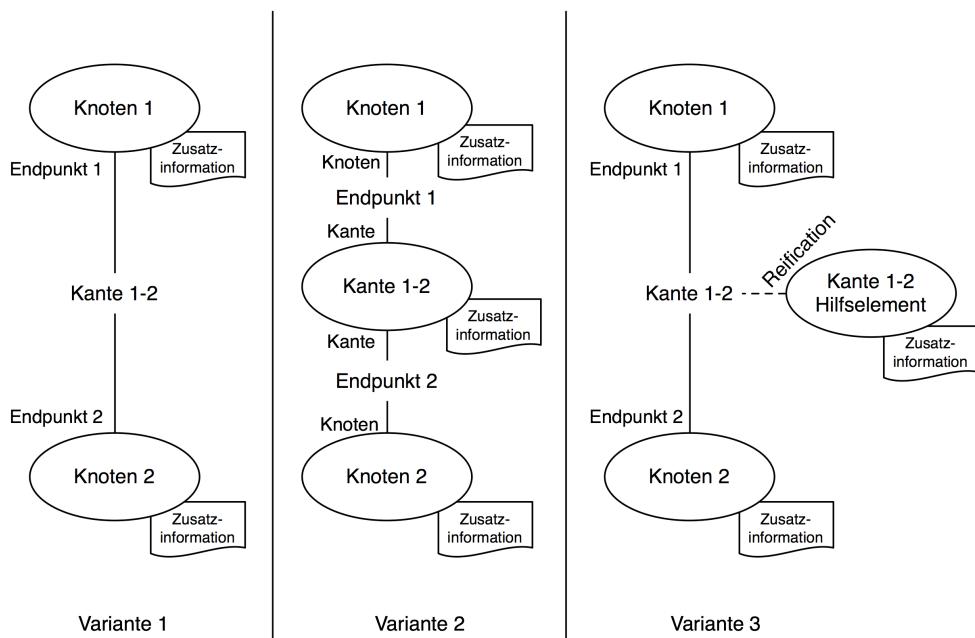


Abbildung 8.7.: Abbildung von Modellinformation in Topic Maps

⁴Unified Modelling Language

Die Bedeutung der Knoten des abzubildenden Modells im Kontext einer bestimmten Kante muss in den direkt abbildenden Varianten (in denen Kanten auf Associations abgebildet werden) durch Roles dargestellt werden. In der indirekten Abbildung (bei der Kanten auf Topics abgebildet werden), kann diese Information auch in den Associations repräsentiert werden, die die Topics die Knoten darstellen und jene die Kanten darstellen miteinander verbinden.

8.3.2. Abbildung des Metamodells

Da Topic Maps von vorne herein nicht auf eine bestimmte semantische Bedeutung der Topics und Associations festgelegt sind, muss auch das Meta-Model der abgebildeten Modellierungssprache mit eingebettet werden. Dies wird in Topic Maps durch die Einbindung von Types ermöglicht. In den nun folgenden Ausführungen wird nur noch auf die direkt abbildende Variante Bezug genommen, da deren Ausdrucksstärke durch die Möglichkeit zur Reification gegenüber dem indirekt abbildenden Ansatz nicht eingeschränkt ist und die unmittelbare Abbildung zu insgesamt kleineren, einfacher zu verwaltenden Topic Maps führt (für eine Kanten wird nur eine Association benötigt, im Gegensatz dazu benötigt der alternative Ansatz ein Topic und mindestens zwei Associations zu Abbildung einer Kante).

Die in der Modellierungssprache festgelegten Arten von Knotentypen (also den eigentlichen Modellierungselementen) werden auf Topic Types abgebildet. Durch die Referenzierung eines einen Knoten repräsentierenden Topics auf diesen Topic Type wird die Bedeutung zugewiesen.

Ebenso wird mit Kanten verfahren. Durch die Einführung von Topics die als Association Types und Role Types fungieren, kann die Bedeutung einer Kante im abzubildenden Modell definiert werden. Dazu wird ein Association Type festgelegt, der die eigentliche Bedeutung der Kante festlegt, die zugehörigen Role Types definieren, wie viele Endpunkte existieren und welche Bedeutung diese haben. Für eine konkrete Kante wird dann eine Association und eine der Anzahl der Endpunkte entsprechende Nummer an Roles erstellt, denen der jeweilige Association bzw. Role Type zugewiesen wird.

Die Verwendung von Occurrences und dementsprechend die Erstellung von Occurrence Types ist zur reinen Abbildung von diagrammatischen Modellen nicht notwendig. Die Verwendung von Occurrences kann aber sinnvoll sein, wenn aus dem ursprünglichen Modell ebenfalls Ressourcen referenziert werden, die für die weitere Verarbeitung des Modells notwendig sind. Occurrences werden dann im Sinne der Topic Map zur Referenzierung dieser Ressourcen verwendet, wobei sich die zu verwendenden Occurrence Types an der Bedeutung der Ressourcen im abzubildenden Modell orientiert.

8. Persistierung

Wie in Abschnitt 8.2.4 bereits beschrieben, bietet die Topic Map selbst jedoch keine Möglichkeit, eine explizite Zuordnung zwischen Topic Types, Association Types und Role Types zu definieren, so dass ein in einer Topic Map repräsentiertes Modell auf semantische Korrektheit hin überprüft werden kann. Die einzige ohne zusätzliche Information überprüfbarer Bedingungen sind die Prüfungen, ob alle Knoten und Kanten (also Topics, Associations und Roles) einer Kategorie (also einem Type) zugewiesen wurden.

Zusätzlich muss also eine externe Möglichkeit geschaffen werden, semantische Korrektheits-Bedingungen zu formulieren und zu überprüfen:

1. Zulässige Kategorien von Knoten (Topic Types)
2. Zulässige Kategorien von Kanten (Association Types, Role Types und eine Zuordnung zwischen diesen, die eine Aussage über die Anzahl und Bedeutung der Endpunkte der Kante zulässt)
3. Zulässige Verbindungen zwischen Knoten und Kanten (Zuordnung zwischen Endpunkten einer Kantenkategorie und den Knotenkategorien, die diese Endpunkte belegen dürfen – also eine Zuordnung zwischen Role Types und Topic Types)

Wie bereits oben beschrieben, können derartige Bedingungen innerhalb oder außerhalb der Topic Map formuliert werden. Die Interpretation der formulierten Bedingungen und deren Anwendung auf konkrete Anwendungsfälle muss immer von außerhalb der Topic Map durchgeführt werden. Hier wird der Ansatz der Repräsentation der Bedingungen innerhalb der Topic Map verfolgt, um durch die Übermittlung einer Topic Map nicht nur ein Modell an sich zu übertragen sondern auch jene Information zu liefern, die zur Interpretation derselben notwendig ist.

Die Formulierung der Bedingungen erfolgt auf Ebene der als Types eingesetzten Topics und vervollständigt so das in der Topic Map enthaltene Meta-Modell der Sprache in der das zu repräsentierende Modell erstellt wurde. Die Information über zulässige Knoten und Kanten wird über die Festlegung von entsprechenden Types definiert. Für jeden Typen wird ein Topic eingeführt, das aus dem auf die Topic Map abgebildeten Modell referenziert werden kann. Alle oben notwendigen Zuordnungen zwischen diesen Types werden über Associations abgebildet, die die als Types verwendeten Topics verbinden (siehe Abbildung 8.8).

Die Definition der Kantentypen erfolgt über eine Association mit mindestens drei Roles. Eine dieser Roles referenziert den Association Type, die anderen Roles verweisen auf die zu verwendenden Role Types, die zur Beschreibung der Endpunkte der Kante verwendet werden (wovon mindestens zwei vorhanden sein müssen). Die Angabe der Kardinalität (d.h. wie oft ein bestimmter Endpunkt im konkreten Modell auftreten darf) wird durch eine die jeweilige Role reifizierendes Topic festgelegt.

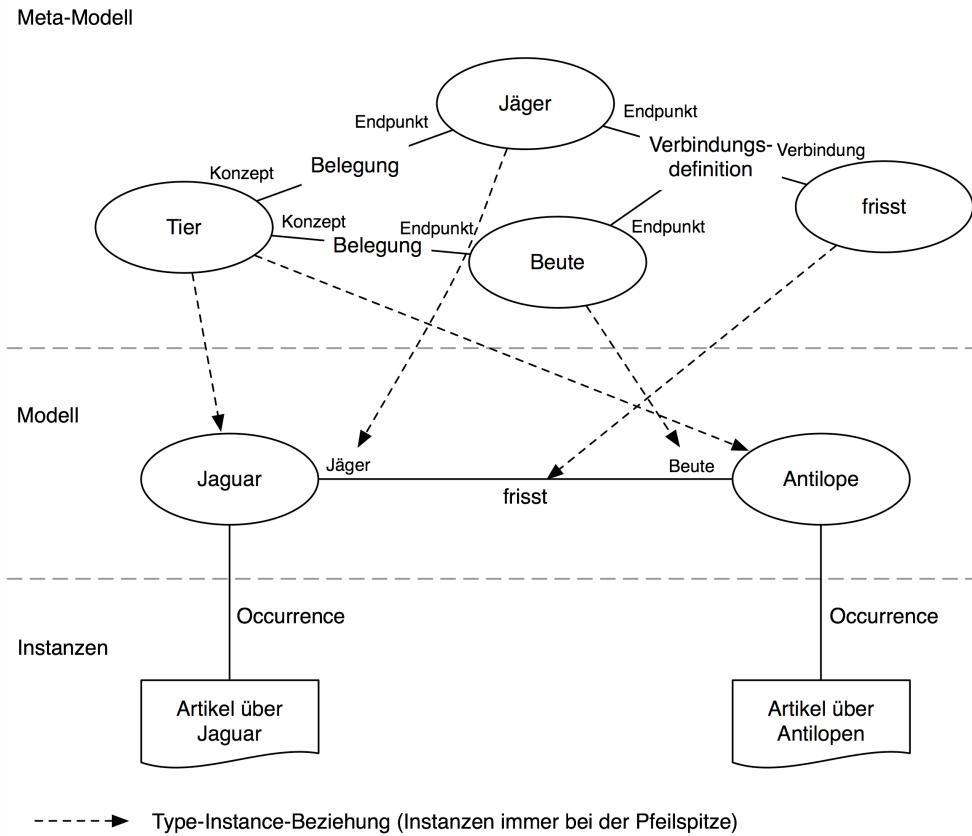


Abbildung 8.8.: Definition des Meta-Models (ohne Kardinalitäten)

Ähnlich wird die Zuordnung zwischen Endpunkten und Knotenkategorien realisiert. Zwischen den entsprechenden Topic Types und Role Types werden Associations erstellt, die festlegen, ob eine bestimmte Knotenkategorie einen Endpunkt einnehmen darf oder nicht. Dazu enthält die betreffende Association mindestens zwei Roles, von denen eine auf den Role Type verweist, der den Endpunkt realisiert und eine entsprechende Anzahl von Roles, an die die zulässigen Topic Types (also Knotenkategorien) angebunden werden.

Die eben beschriebenen Abbildungsvorschriften selbst werden ebenfalls in der Topic Map abgebildet. Die Topic Map enthält damit auch das Meta-Meta-Modell das die Elemente die zur Festlegung eines Meta-Models und deren Zusammenspiel festlegt. Eine so definierte Topic Map ist also semantisch vollständig definiert und ermöglicht eine Rekonstruktion eines Modells, das in einer im Vorfeld unbekannten Sprache modelliert wurde, sofern lediglich das Meta-Meta-Modell bekannt ist, dass zur Interpretation der Sprachbeschreibung (Meta-Modell) notwendig ist (siehe Abbildung 8.9).

8.3.3. Abgrenzung von Submodellen

In einem Modell kann es sinnvoll oder notwendig sein, unterschiedliche Modellbereiche voneinander abzugrenzen. Der Grund für die Abgrenzung kann ein inhaltlicher sein (z.B. eine Partitionierung nach Akteuren bei Aktivitätsdiagrammen (Rumbaugh et al., 2004)) oder aus dem Modellierungsvorgang heraus motiviert sein (etwa bei der Abgrenzung von Teilmodellen, die durch verschiedene Personen erstellt wurden). Außerdem ist es möglich, in einer Topic Map mehrere Modelle zu repräsentieren, die ebenfalls voneinander abgrenzt werden müssen.

Für diese Abgrenzung bietet sich die Verwendung von Scopes an. Scopes haben zwar keinen Einfluss auf die vorhandenen Topics, wirken jedoch auf die Statements und – hier relevant – vor allem auf die die Topics verbindenden Associations. Werden also Scopes eingesetzt, um Teilmodelle voneinander zu unterscheiden bzw. zu trennen, so können die im Moment nicht relevanten Teilmodelle zwar nicht „ausgeblendet“ werden (im Sinne von „temporär vollkommen entfernt“, durch die Entfernung der nicht relevanten Statements sind jedoch nur noch jene Topics untereinander verbunden, die dem aktuell betrachteten Submodell angehören. Ausgehend von einer beliebigen, im Scope gültigen Association oder einem Topic, das bekannterweise dem aktuellen Teilmodell angehört kann so der gesamte relevante Teil der Topic Map erschlossen werden.

Die Realisierung der Trennung zwischen Teilmodellen durch das Ausblenden irrelevanter Verbindungen zwischen Topics birgt einen weiteren potentiellen Vorteil. Werden in einer Topic Map mehrere untereinander zusammenhängende Modelle

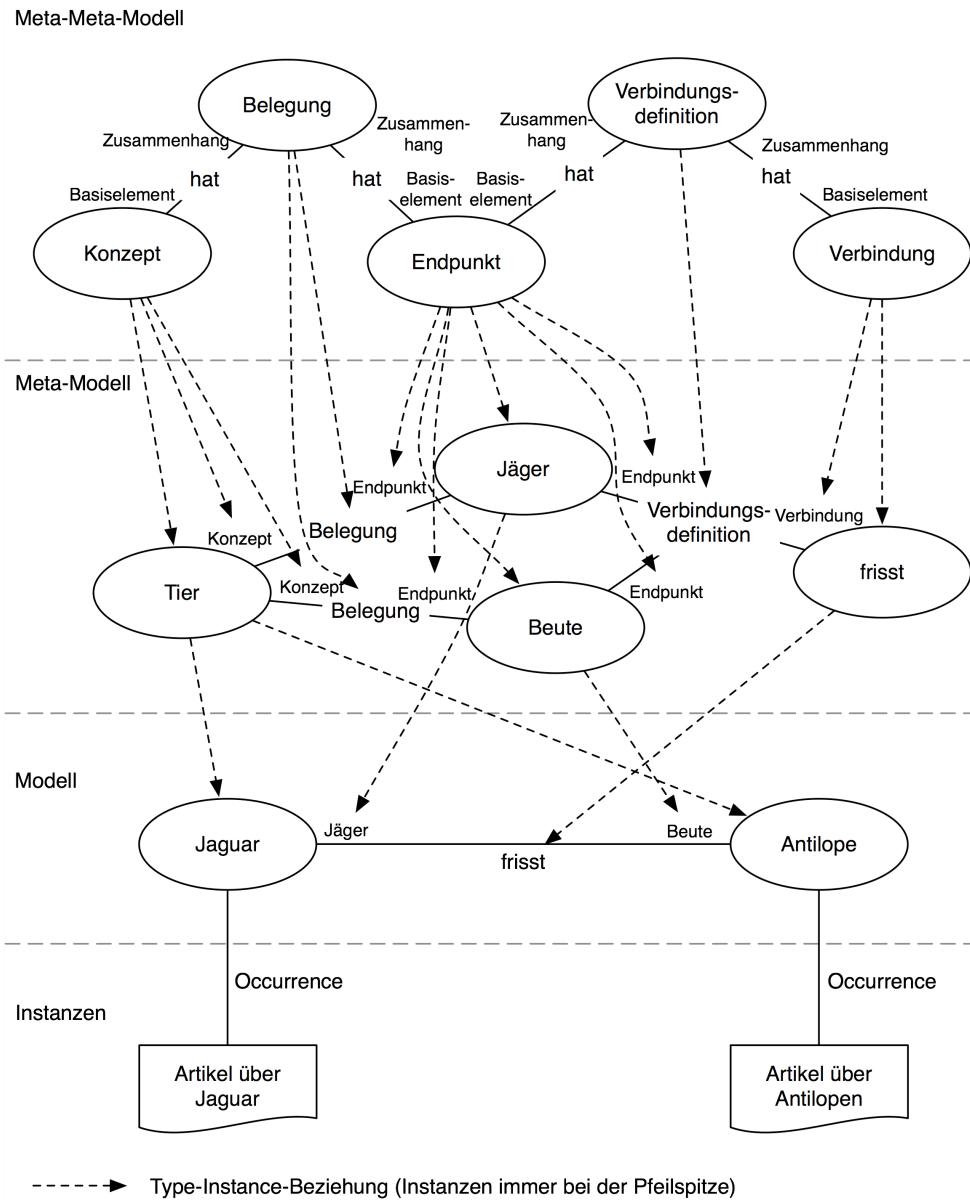


Abbildung 8.9.: Einbindung des Meta-Meta-Modells

abgebildet, so wird – der Grundforderung einer Topic Map nach Eineindeutigkeit entsprechend – jedes Element nur einmal abgebildet, egal ob es in nur einem Modell verwendet wird oder in mehreren. Durch den Einsatz von Scopes bilden in mehreren Modellen verwendete Elemente automatisch eine Schnittstelle, an der zwischen den Modellen navigiert werden kann. Dieser Vorteil muss in herkömmlichen Modellierungsansätzen mit mehreren untereinander verknüpften Modellen wie der UML (Rumbaugh et al., 2004) oder ARIS (Scheer, 2003) technisch durch explizite Verknüpfung bzw. Referenzierung der äquivalenten Modellelemente in den unterschiedlichen Modellen erzeugt werden.

Zur Kennzeichnung des Scopes muss zumindest ein Topic verwendet werden. Im Meta-Modell muss festgelegt werden, ob dazu ein Topic eines bestehenden Types verwendet wird oder ein neuer Type eingeführt werden, dessen Topics ausschließlich zum Aufspannen eines Scopes verwendet werden. Dazu wird im Meta-Meta-Modell ein Type „Partition“ eingeführt, der für alle Elemente des Meta-Modells verwendet werden muss, deren konkrete Instanzen einen Scope im Modell kennzeichnen sollen.

8.3.4. Flexibilisierung der Abbildung

Wie in Kapitel XY beschrieben, ist das Meta-Modell von Modellen die im vorgestellten Ansatz entstehen, nicht im vorhinein festgelegt. Die Modellierungssprache – also das Meta-Modell – wird während der Modellbildung semantisch definiert und ist damit erst zur Modellierungszeit bekannt. Das Metamodell kann außerdem während der Modellierung erweitert werden, es ist auch möglich, dass sich die Bedeutung bereits existierender Meta-Modell-Elemente ändert.

Für die Persistierung bedeutet dies, dass das Meta-Modell nicht im Vorhinein sondern erst zum Zeitpunkt der Speicherung festgeschrieben werden kann. Außerdem kann die Prüfung auf semantische Korrektheit des Modells ebenfalls nur durchgeführt werden, sobald das Modell definiert ist.

Um die geforderte Flexibilität bei der Sprachdefinition in Form und Zeitpunkt zu gewährleisten, wurde von Neubauer (2008) ein System entwickelt, dass es erlaubt, dynamisch zur Laufzeit Modellelemente zu definieren, die Regeln zu deren Verwendung festzulegen und diese ohne Unterbrechung des Modellierungsvorgangs unmittelbar zu verwenden (wobei auch die Prüfung der semantischen Korrektheit zur Laufzeit adaptiert wird). Die technische Umsetzung dieses Ansatzes wird in Abschnitt 8.4.2 beschrieben.

8.4. Technische Umsetzung der Persistierung von Modellen

Die oben beschriebenen Konzepte zur Persistierung von Modellen wurden wie die übrigen Software-Komponenten des Systems in Java implementiert. Die Basis der Persistenz-Komponente bildet eine Topic Map Engine, die im Rahmen einer Arbeit über flexible Content-Repräsentation vom Autor entwickelt wurde (Oppl, 2007a). Das in (Neubauer, 2008) entwickelte System zur Generierung und Verwendung dynamischer Metamodelle, das bereits in Abschnitt 8.3.4 konzeptionell beschrieben wurde, wird in der Folge hinsichtlich seiner technischen Umsetzung betrachtet. Basierend auf diesen beiden Komponenten wurde die eigentlichen Persistierung implementiert. Der dort verfolgte Ansatz ist Thema des letzten Teils dieses Abschnitts.

8.4.1. Topic Map Engine

Als Topic Map Engine bezeichnet man ein Software-Modul, mit dessen Hilfe Topic Maps verwaltet werden können und das im Normalfall auch Funktionalität zur Persistierung der Topic Map anbietet. Für den Topic Map Standard (ISO JTC1/SC34/WG3, 2008), der dieser Arbeit zugrunde liegt, war zum Zeitpunkt der Erstellung der Software nur eine derartige Engine vertrieben, die von Ontopia⁵ kommerziell vertrieben wird und keine offenen Schnittstellen bietet. Zu diesem Zeitpunkt noch nicht für den verwendeten Topic Map Standard verfügbar war die Open-Source-Engine TinyTIM⁶, die auf Basis der ebenfalls erst seit einigen Monaten verfügbaren Topic Map API⁷ eine offene Schnittstelle zur Verwaltung von Topic Maps in Java anbietet und unterschiedliche Formate zur Persistierung und zum Import unterstützt.

Für diese Arbeit wurde mangels verfügbarer Alternativen eine eigene Topic Map Engine erstellt, deren Funktionalität und Aufbau hier nur kurz umrissen werden soll. Die detaillierte Beschreibung der Implementierung und die allgemeine Verwendung der Engine sind in (Oppl, 2007a) beschrieben.

Kernkomponenten

Die Topic Map Engine bildet die Komponenten einer Topic Map direkt auf Java Klassen ab.

⁵<http://www.ontopia.com>

⁶<http://tinytim.sourceforge.net/>

⁷Application Interface

Einführung von Domänenmodellen

Schnittstelle zur Persistierung

Die Topic Map Engine bietet zur Persistierung ein einfaches Interface an, über welches Topic Maps auf der Engine exportiert werden können bzw. in die Engine geladen werden. Das Interface abstrahiert dabei von der konkreten Persistierungs-Technologie und erlaubt die Einbindung unterschiedlicher Ansätze zur Speicherung von Topic Maps.

Konkret wurden drei Implementierungen des Persistenz-Interfaces erstellt. Die erste Implementierung liest und schreibt Topic Maps von bzw. in im XTM⁸-Format (ISO JTC1/SC34/WG3, 2006) abgelegten XML-Dateien. So wird ein standardkonformer Datenaustausch zwischen Topic Map Engines ermöglicht. Die zweite Implementierung bindet die Topic Map Engine via Hibernate (Red Hat Middleware, 2007) an eine relationale Datenbank an. Im Gegensatz zur Speicherung in XML-Files ermöglicht die Ablage der Daten in einem RDBMS⁹ einen effizienteren, selektiven Zugriff, sodass nicht unter allen Umständen die gesamte Datenbasis einer Topic Map im Arbeitsspeicher gehalten werden muss.

Die dritte Implementierung des Persistenz-Interfaces ermöglicht lediglich den Export einer Topic Map in ein von GraphViz (Ellson et al., 2002) interpretierbares Format, das eine graphische Darstellung der Topic Map mit automatischer Anordnung der Topics ermöglicht. Da das GraphViz-Format jedoch nicht so ausdrucksstark ist wie der Topic Map Ansatz, geht beim Export Information verloren. Dadurch ist es nicht möglich, eine Topic Map aus einem GraphViz-File zu rekonstruieren, der Import in die Engine ist also nicht möglich.

Zusätzlich zur Ausgabe der gesamten Topic Map ist das GraphViz-Export-Modul auch in der Lage eine mittels HTML Imagemaps navigierbare Version der Topic Map zu exportieren, in der jeweils immer nur ein Topic mit dessen Kontext (also allen Topics, die mit dem zentralen Topic verbunden sind) dargestellt wird (siehe Abbildung XY). Durch die Imagemap werden die Kontext-Tops mit jenen Ansichten verknüpft, wo jeweils diese im Fokus der Betrachtung stehen. Damit ist eine schrittweise Navigation durch die Topic Map möglich.

⁸XML Topic Map

⁹Relationales Datenbank Management System

8.4. Technische Umsetzung der Persistierung von Modellen



Abbildung 8.10.: Ausschnitt einer mittels GraphViz visualisierten Topic Map

8.4.2. Dynamische Metamodelle

8.4.3. Persistierung

8.5. Export graphischer Repräsentationen

Neben der Persistierung der Modelle in Form einer Topic Map ist es auch sinnvoll, die Modelle in deren graphischer Form als Referenz abzulegen. Das hier entwickelte Werkzeug bedient sich der graphischen Ausgabefunktionalitäten, die der Java Klassenbibliothek mit der Version 1.4 hinzugefügt wurden, um das aktuelle Modell in unterschiedlichen Formen als Grafik auszugeben und zur späteren Referenz zu speichern.

8.5.1. Ausgabeformen

Zur Speicherung der graphischen Repräsentation eines Modells wird grundsätzlich die Visualisierung verwendet, die auf dem sekundären Ausgabekanal (also dem Bildschirm) zur Anwendung kommt. Beim Export sind nun unterschiedliche Modellaspekte zu berücksichtigen, die je nach intendiertem Verwendungszweck einzeln oder in Kombination in die Ausgabe eingehen können. Diese Aspekte sind im Einzelnen

- der aktuell auf der Oberfläche befindliche Modellzustand
- die hierarchisch in diesen eingebetteten Submodelle
- die Modellierungshistorie, also die Entwicklung des Modells über die Zeit

Die Darstellung dieser Aspekte in einer graphischen Repräsentation ist (außer im erstgenannten Fall) insofern komplex, als dass eine beliebig lange zeitliche Abfolge bzw. eine beliebig tief verschachtelte Hierarchie in den zweidimensionalen Raum abgebildet werden muss. Im Falle einer Kombination des zweit- und drittgenannten Aspektes müssen zwei Dimensionen zugleich abgebildet werden, was die Darstellung zusätzlich erschwert.

Der aktuell auf der Oberfläche befindliche Modellzustand wird exakt wie dargestellt in eine Grafik transformiert und als Bild abgespeichert. Zur Abbildung der Modellierungshistorie bietet sich an, die einzelnen gespeicherten Modellzustände chronologisch anzurufen. Der sich so ergebende Zeitstrahl beginnt links oben und setzt sich von links nach rechts und oben nach unten bis in die rechte untere Ecke fort, wo wiederum der aktuelle Modellzustand dargestellt wird. Die zweidimensionale Abbildung des Zeitstrahls erfolgt dabei derart, dass sowohl die horizontale als auch die vertikale Ausdehnung des resultierenden Bildes minimal sind (siehe Abbildung 8.11).

8.5. Export graphischer Repräsentationen



Abbildung 8.11.: Modellierungshistorie als exportierte Grafik

Bei der Darstellung eines Modells mit hierarchisch verschachtelten Teilmodellen bietet sich eine baumartige Darstellung mit dem aktuellen Modellzustand als Wurzelknoten an. Diese ist aufgrund der notwendigen Detaildarstellung der einzelnen Modelle jedoch platzintensiv und kann nur schwer als physisches Dokument abgelegt werden. Deshalb wurde alternativ die hierarchische Struktur auf eine der Darstellung der zeitlichen Modellentwicklung ähnliche Darstellungsform abgebildet. Dabei wird die Hierarchie flach ausgerollt, die Einbettungen zeigen sich durch einen graduell dunkler werdenden Modellhintergrund für tiefere verschachtelte Ebenen. Zusätzlich wird in jedes Submodell farblich abgesetzt auch das jeweilige Container-element eingebettet. Die Abfolge der einzelnen Modelle startet mit dem aktuellen Modellzustand als erstem Knoten. Dahinter wird das erste im aktuellen Modell eingebettete Teilmodell angezeigt. Besitzt dieses Teilmodell wiederum eingebettete Teilmobile, so werden diese in der Folge mit erneut abgedunkeltem Hintergrund dargestellt. Diese Form der Darstellung wird fortgesetzt bis keine weiteren eingebetteten Modelle mehr vorhanden sind. Es folgt (sofern vorhanden) das zweite Submodelle des aktuellen Modellzustandes (mit dem abgedunkelten Hintergrund der ersten Einbettungsebene). Diese Hierarchie wird wiederum bis zu den Endpunkten nach unten verfolgt, es folgt ggf. das dritte Submodell auf der ersten Einbettungsebene. Die sich so ergebende Linie an Modellzuständen wird wie der chronologischen Darstellung der Modellierungshistorie so umgebrochen dass sich sowohl in horizontaler als auch in vertikaler Richtung eine minimale Ausdehnung ergibt (siehe Abbildung 8.12).

In der komplexesten Ausprägung der Darstellung eines Modells als graphische Repräsentation werden sowohl die Historie der Modellentstehung als auch die Hierarchie der eingebetteten Submodelle zugleich dargestellt. Dies erfolgt hier durch die Kombination der beiden eben beschriebenen Ansätze. Die Darstellung der Modellierungshistorie bildet die oberste Ebene der Modelldarstellung. Beginnend mit dem ältesten gespeicherten Modellzustand werden die einzelnen Modelle in der Reihenfolge ihrer Entstehung abgebildet, wobei zu jedem Modellzustand unmittelbar folgend dessen eingebetteten Submodelle ausgegeben werden (deren Entstehung nicht mehr separat dargestellt wird). So ergibt sich eine kombinierte Linie aus chronologischer Modellentwicklung und eingebetteten Teilmodellen. Diese wird wie schon oben mit minimaler Ausdehnung in horizontaler wie vertikaler Richtung auf eine Grafik abgebildet. Die Entwicklung des Modells ist dabei wie gehabt von links oben nach rechts unten zu verfolgen, wobei nur jene Modellzustände mit weißem Hintergrund auf oberster Ebene der Zeitlinie angehören. Alle dunkler hinterlegten Modelle sind Teilmodelle und sind dem nach vorne nächstgelegenen weiß hinterlegten Modell zuzuordnen.

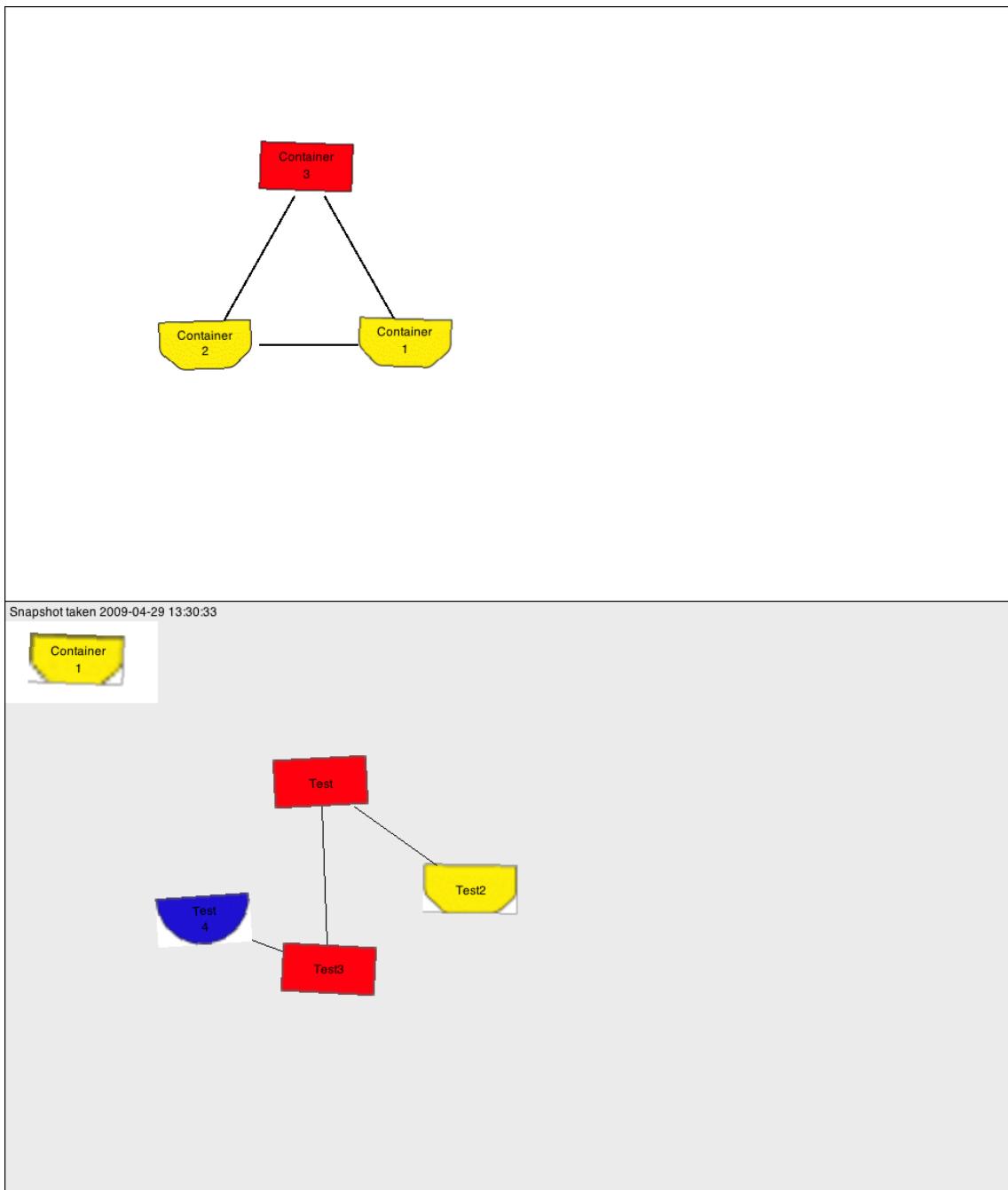


Abbildung 8.12.: Modell-Hierarchie als exportierte Grafik

8.5.2. Technische Umsetzung des graphischen Exports

Zur Umsetzung des graphischen Exports wird auf die mit der Java Plattform in der Version 1.4 eingeführten Klassen zu Verarbeitung von Grafikformen zurückgegriffen. Diese unterstützen standardmäßig gängige Grafikformate wie JPEG¹⁰, GIF¹¹ oder PNG¹². Im konkreten Fall wurde ein verlustfrei komprimierendes Format gewählt, verlustbehaftete Verfahren wie JPEG sind für die hier abzubildenden feinen Strukturen nicht geeignet, da es durch Kompressionsartefakte zu Qualitätsminderungen in der Darstellung von Details kommt.

Die Ausgabe einer Datei im gewählten Grafikformat wird vollständig von der Klasse `ImageIO` übernommen. Der Schnittstellen-Methode `write` sind als Parameter der Dateiname, das Grafikformat sowie ein Objekt der Klasse `BufferedImage` (allgemeiner: einer Klasse, die das Interface `RenderedImage` implementiert) zu übergeben, dass die eigentlichen Bilddaten enthält.

Das `BufferedImage`-Objekt kann durch einen Methoden-Aufruf mit der graphischen Repräsentation einer Klasse, die von der Java AWT¹³-Klasse `Component` abgeleitet ist, befüllt werden. Da alle graphischen Komponenten des JHotDraw-Frameworks Subklassen eben dieser Klasse sind (inklusive der Zeichenoberfläche selbst), können diese durch einen Aufruf ihrer `paint`-Methode in ein ausreichend großes `BufferedImage` (bzw. dessen `Graphics`-Objekt) geschrieben werden.

In den aus mehr als einem Teilmodell bestehenden Ausgaben (also der Historie, der hierarchischen Darstellung der Teilmodelle oder der Kombination dieser beiden Fälle) muss das Bild aus den graphischen Repräsentationen der einzelnen Teilmodelle zusammengesetzt werden. Dazu werden (im Falle der hierarchischen Teilmodelle rekursiv) die logischen Modelle erzeugt und bereits in der korrekten linearisierten Darstellungsform in einem Vektor gespeichert. Die einzelnen logischen Modelle werden dann in graphische Repräsentationen umgewandelt und in je einem `BufferedImage` gespeichert. Diese werden in der Folge so zusammengesetzt, dass die horizontale und vertikale Ausdehung der Gesamtfläche minimal ist.

8.6. Zusammenfassung

¹⁰Joint Photographic Experts Group (File Interchange Format)

¹¹Graphics Interchange Format

¹²Portable Network Graphics

¹³Abstract Window Toolkit

Teil III.

Evaluierung

Einleitung

Nach der nun erfolgten Beschreibung der Umsetzung des Werkzeugs wird in diesem Teil die Überprüfung der Verwendbarkeit des Werkzeugs und seiner Effekte behandelt. Ziel dieser Arbeit war es, explizite Articulation Work zu unterstützen. Letztendliches Ziel dieses Teils muss sein, diese Anforderung hinsichtlich ihrer Erfüllung oder Nicht-Erfüllung zu überprüfen.

Wie in Kapitel XY argumentiert, führt der Weg zur Unterstützung expliziter Articulation Work über die (kollaborative) Externalisierung mentaler Modelle. Die aus dieser Externalisierung resultierenden Strukturen sind ihrerseits diagrammatische Modelle. Die Qualität dieser Modelle ist vielschichtig bewertbar, im Kontext des hier verfolgten Verwendungszwecks sind aber einige Bewertungsdimensionen identifizierbar, die bei der Unterstützung von Articulation Work relevant sind. Diese Dimensionen werden ebenfalls hinsichtlich ihrer Ausprägung im hier vorgestellten Werkzeug zu bewerten sein.

Letztendlich wird die Externalisierung mentaler Modelle technisch durch ein Tabletop Interface unterstützt. Auch die technische Umsetzung bzw. deren Verständlichkeit und Verwendbarkeit hat Auswirkungen auf den Erfolg der Externalisierung und damit der durchgeführten Articulation Work. Das Werkzeug selbst und seine Nutzung muss also ebenfalls untersucht und im Kontext der Anforderungen in dieser Arbeit bewertet werden.

Entsprechend dieser Ausführungen wurde die hier beschriebenen Untersuchung durchgeführt. Sie gliedert sich in drei Teile, die sich mit dem Werkzeug selbst, den erstellten Modellen und den Auswirkungen durchgeführten expliziten Articulation Work auseinandersetzen. Die Struktur dieses Teiles spiegelt diese Aufteilung wieder. In Kapitel 9 wird das Werkzeug aus Sicht seiner Eigenschaften als Tabletop Interface theoretisch-konzeptionell betrachtet und aus den in diesen Bereich verfügbaren Analyse- und Beschreibungsframeworks mögliches Verbesserungspotential identifiziert. In Kapitel XY werden Design und Umsetzung jener Tests beschrieben, in denen die Benutzbarkeit und Verständlichkeit des Werkzeugs überprüft wurden. Die Überprüfung der Effekte des Werkzeugs beginnt im darauf folgenden Kapitel XY, in dem die erstellten Modelle und deren Entstehungsprozess Gegenstand der Betrachtung sind. Letztendlich wird in Kapitel XY auf die Wirkung des Werkzeugs auf Articulation Work und damit letztendlich auf die im jeweiligen Anwendungsfall zu erzielenden Effekte eingegangen.

9. Konzeptionelle Einordnung

An dieser Stelle erfolgt ein Rückgriff auf die in Abschnitt 5.2 beschriebenen konzeptionellen Beschreibungs-Ansätze für Tangible User Interfaces. Das in den letzten drei Kapiteln beschriebene Werkzeug wird hier im Lichte eben dieser Ansätze betrachtet bzw. wo möglich mittels der vorgeschlagenen Schemata beschrieben. Damit werden zwei Ziele verfolgt. Einerseits soll die Praxistauglichkeit der konzeptionellen Ansätze überprüft werden, indem sie auf ein aktuelles, im Vergleich zu den beispielhaft in den Artikeln angegebenen Systemen komplexes und flexibles System angewandt werden. Andererseits soll durch die konzeptionelle Betrachtung des Werkzeugs das Design und die Umsetzung auf Inkonsistenzen geprüft werden und Potential für Verbesserungen des Werkzeugs identifiziert werden. Die hier abgeleiteten Potentiale werden in einem weiteren Schritt den Erkenntnissen aus der praktischen Evaluierung des Systems gegenübergestellt. So wird es möglich, auch die potentiellen Auswirkungen der Anwendung eines der hier vorgestellten konzeptionellen Ansätze bei der Konzeption und Umsetzung eines TUI betrachtet werden.

Dieses Kapitel folgt im Aufbau der Struktur des Abschnitts 5.2 und betrachtet in der Folge jeden der vorgestellten Ansätze in einem separaten Abschnitt. Dabei werden jeweils die Konzepte des Ansatzes auf das Werkzeug angewandt (Unterabschnitt "Abbildung") und in der Folge die Erkenntnisse über die Eignung des Ansatzes und mögliches Verbesserungspotential des Werkzeugs angeführt (Unterabschnitt "Bewertung"). Abschließend erfolgt eine zusammenfassende Betrachtung der Ergebnisse hinsichtlich der beiden oben formulierten Ziele.

9.1. Einordnung in den Bricks-Designraum

Grundlage der Betrachtungen in diesem Abschnitt ist das Konzept der Tangible Bits (Fitzmaurice et al., 1995), der in Abschnitt 5.2.1 beschrieben wird.

9.1.1. Abbildung

Aus den Erfahrungen mit der Erstellung des „Bricks“-Systems leiten Fitzmaurice et al. (1995) einen „Design Space“ ab, der 13 Merkmale definiert, die bei der Kon-

9. Konzeptionelle Einordnung

zeption eines TUI beachtet werden sollten bzw. die sich zur gegenüberstellenden Bewertung von TUIs eignen. Neben den Merkmalen geben die Autoren auch mögliche Ausprägungen an, die gemeinsam den Design Raum abstecken. Für das hier vorgestellte Werkzeug ergibt sich folgende Zuordnung:

In der Kategorie *Brick's internal ability* ist das System bzw. die eingesetzten Tokens als „inert“ zu klassifizieren, da die physischen Objekte keine Elektronik aufweisen und somit kein aktives Verhalten zeigen können. Bezieht man die unmittelbare Umgebung der Elemente, auf die Information projiziert werden kann, in die Beurteilung mit ein, so ist die Einstufung „simple expressions“ zu rechtfertigen.

In der Kategorie *Input & Output* werden eingabeseitig in unterschiedlichen Interaktionen folgende Parameter kontinuierlich oder ereignisgesteuert erfasst: X-Y-Position (kont.) , Rotation (kont.) , Tastatureingabe (ereignisgest.) und Kamerabild (ereignisgest.). Ausgabeseitig werden folgende Parameter eingesetzt: Zustand auf der physischen Oberfläche, Projektion und Bildschirm.

In der Kategorie *Spatially aware* ist (bezogen auf die Tokens) die Ausprägung „unaware“ zu wählen, da die physischen Elemente selbst keine Möglichkeit zur Feststellung der Konfiguration ihrer Umgebung haben. Bezogen auf das Gesamtsystem ist die Ausprägung „mutual awareness“ zu wählen, da den softwareseitigen Repräsentationen der Elemente durch das im Hintergrund arbeitende Lokalisierungssystem durchaus bekannt ist, wo sie sich befinden und welche Elemente sich in ihrer Nähe befinden.

In der Kategorie *Communication* ist keine Einordnung möglich, da die physischen Elemente des Systems nicht untereinander kommunizieren.

In der Kategorie *Interaction time span* sind die Ausprägungen „quick“ und „interaction cache“ zu wählen. Die meisten Interaktionen laufen ereignisbasiert ab und sind mit der Auslösung wieder abgeschlossen („quick“). Manche Interaktionen (z.B. Herstellung einer Verbindung, Kontrolle der Modellierungshistorie,...) benötigen jedoch längere Zeit bzw. die Manipulation mehrerer Tokens („interaction cache“).

In der Kategorie *Bricks in use at same time* ist bei Betrachtung des gesamten Systems die Ausprägung (Größenordnung) „5-10“ zu wählen, da potentiell so viele Modellierungselemente gleichzeitig (auch durch mehrere Personen) benutzt werden. Bezogen auf die Werkzeug-Tokens (also die unmittelbar funktionsauslösenden physischen Elemente, die eigentlich von diesem Ansatz eigentlich betrachtet werden) sind die Ausprägungen „1“ oder „2“ (je nach Interaktion) zu wählen.

In der Kategorie *Function assignment* ist bezogen auf die Werkzeug-Tokens die Ausprägung „permanent“ zu wählen, da diesen die durch sie ausgelöste Funktion fix zugeordnet ist.

In der Kategorie *Interaction representations* ist die Ausprägung „balanced“ insofern für das vorliegende Werkzeug passend, als dass die physischen und digitalen Elemente einander in der Repräsentation des Systemzustandes gleichberechtigt ergänzen.

In der Kategorie *Physical & virtual layers* erfüllt das System durch die beiden Ausgabekanäle beide Ausprägungen („direct“ für die Tischoberfläche, „indirect“ für den sekundären Ausgabekanal).

In der Kategorie *Bond between physical & virtual layers* sind physische und virtuelle Ebene im hier vorgestellten System „tightly coupled“, da die Repräsentationen auf beiden Ebenen stets synchron sind.

In der Kategorie *Operating granularity* ist aufgrund der physischen Natur des Systems (Tisch) die Ausprägung „Desktop“ zu wählen, wobei dies auch mit der vorgesehenen Auflösung des Tracking („fraction of inches accuracy“) korreliert.

In der Kategorie *Operating surface type* erfüllt das System die Anforderungen der Ausprägung „dynamic“, da sich die Oberfläche durch die Projektion von Information zur Laufzeit ändert.

In der Kategorie *Operating surface texture* ist die Ausprägung „continuous“ auszuwählen, da die Elemente frei auf der Oberfläche platziert werden können.

9.1.2. Bewertung

Der von Fitzmaurice et al. (1995) aufgespannte Design Raum für Graspable User Interfaces ist aus dem von den Autoren entwickelten System abgeleitet und weist mangels zum Zeitpunkt der Erstellung vorhandenen Alternativen starke Spezifika auf, die den Ansatz zur Einordnung anderer Systeme nur bedingt geeignet machen. Insbesondere die Ableitung mehrerer Eigenschaften aus der Tatsache, dass die physischen Elemente des System (die „Bricks“) aktive Bausteine sind und auf eine Oberfläche bewegt werden, schränkt die Verwendbarkeit einiger Kategorien im Allgemeinen ein (z.B. *Communication* oder *Spatially aware*). Außerdem ist der Design Raum auf Systeme ausgerichtet, die physische Elemente als Eingabewerkzeuge verwenden und berücksichtigt keine Elemente, die ausschließlich zur Informationsrepräsentation verwendet werden.

Das hier vorgestellte Werkzeug konnte weitgehend in den Design Raum eingeordnet werden, wenn auch die Verwendung passiver Tokens nicht vorgesehen ist und damit einige Kategorien nicht sinnvoll belegt werden können. Ansätze zur Verbesserung des Werkzeugs können nicht abgeleitet werden, da der Designraum für Detailbetrachtungen zu unspezifisch bleibt und seine generelle Ausrichtung nicht vollständig auf die Eigenschaften des Werkzeugs abgebildet werden können.

9.2. Bestimmung der Eigenschaften des Graspable User Interfaces Ansatz

Grundlage der Betrachtungen in diesem Abschnitt ist das Konzept der Tangible Bits (Fitzmaurice, 1996), der in Abschnitt 5.2.2 beschrieben wird.

9.2.1. Abbildung

Fitzmaurice (1996) definiert fünf Eigenschaften, die ein „Graspable User Interface“ ausmachen und legt mögliche Ausprägungen fest, deren Werte für ein konkretes System eine Aussage über dessen „Graspability“ zulässt. Für das hier vorgestellte Werkzeug sind folgende Einstufungen argumentierbar:

In der Kategorie *Space-multiplexing* ist die höchste Ausprägung „permanent, never reassigned“ auszuwählen. Dies ist der Fall, weil sämtlichen Werkzeugtokens des Systems eine vorgegebene Funktion zugewiesen ist, die sich während der Laufzeit nicht ändert.

In der Kategorie *Concurrency* ist hinsichtlich der Verwendung von Werkzeugen die Ausprägung „occasionally 2“ zu wählen. Die Eigenschaften beziehen sich ausschließlich auf Werkzeuge zur Manipulation digitaler Information, so dass die Modellierungstokens, die den Systemzustand repräsentieren, außer acht gelassen werden. „Occasionally 2“ trifft dann deswegen zu, weil sowohl bei der Verbindungsherstellung als auch zum Auslösen der Wiederherstellungsunterstützung jeweils zwei Werkzeug-Token gleichzeitig eingesetzt werden müssen, in allen anderen Fällen aber nur ein Werkzeug zum Einsatz kommt. Berücksichtigt man die Modellierungstokens (als Werkzeuge zur Manipulation des Systemzustandes), wäre die höchste Ausprägung „more than 3“ zu wählen.

In der Kategorie *Physical form* ist die höhere Ausprägung „specific“ auszuwählen, da für jede Funktionalität ein spezifisches Werkzeug-Token existiert.

In der Kategorie *Spatially aware* ist hinsichtlich der Werkzeugtokens eine Mischform zwischen „unaware“ und „aware“ zu wählen. Bei einem Teil der Werkzeugtokens ist die konkret ausgeführte Funktion von dessen Position bzw. Nähe zu Modellierungstokens abhängig (bei Markierungstokens) oder wird durch dessen Orientierung beeinflusst (beim Historien-Kontroll-Token). Die übrigen Werkzeug-Tokens sind jedoch insofern „spatially unaware“, als das ihre Positionierung auf der Oberfläche keinen Einfluss auf deren Funktionalität hat. Die Modellierungstokens und die einbettbaren Tokens wären bei Berücksichtigung als „spatially aware“ zu klassifizieren.

In der Kategorie *Spatial reconfigurability* ist das System in die Ausprägung „track“ einzuordnen, da die einzelnen Werkzeuge nicht unabhängig von der Oberfläche und der in ihr integrierten optischen Tracking-Funktion verwendet werden können.

9.2.2. Bewertung

Die Eigenschaften von Graspable User Interfaces erlauben eine allgemeine Bewertung eines TUI. Sie sind nicht geeignet, um eine detaillierte Analyse oder Spezifikation durchzuführen. Dieser Aspekt – die Eigenschaften des Gesamtsystems – wird jedoch bei den meisten anderen Frameworks außer acht gelassen, so dass eine Einordnung in das vorgeschlagene Schema durchaus sinnvoll sein kann.

Für das hier vorgestellte Werkzeug wurden durchwegs Ausprägungen identifiziert, die auf eher hohe „Graspability“ hinweisen. Ein Nachteil des Ansatzes liegt in der Fokussierung auf reine „Steuerungs“-TUIs, die zur Kontrolle oder Manipulation digitaler Information verwendet werden. Würde der Repräsentations-Aspekt eines TUI (stärker) berücksichtigt, würden sich wie oben beschrieben zum Teil höhere Ausprägungen in einzelnen Kategorien ergeben. Verbesserungspotential kann ob der allgemeinen und relativ abstrakten Beschreibung des Werkzeugs mit diesem Ansatz nicht abgeleitet werden.

9.3. Betrachtung im Lichte des Tangible Bits Ansatzes

Grundlage der Betrachtungen in diesem Abschnitt ist das Konzept der Tangible Bits (Ishii und Ullmer, 1997), der in Abschnitt 5.2.3 beschrieben wird.

9.3.1. Abbildung

Das hier vorgestellte Werkzeug kann hinsichtlich seiner Funktion als eine Instanz des Konzepts „Interactive Surface“ betrachtet werden. Die „Surface“ ist hierbei eine Tischoberfläche, auf der interagiert wird. Die im Rahmen der Beschreibung des „metaDESK“ (Ullmer und Ishii, 1997) als Beispiel für eine „Interactive Surface“ eingeführten TUI-Elemente finden zum Teil auch im hier vorgestellten Werkzeug Anwendung.

Die Modellierungstokens und einbettbaren Tokens des Werkzeugs sind *Phicons*, also passive Träger von digitaler Information. Die Werkzeugtokens zur Manipulation des Modells entsprechen *Phandles*, also Elemente, die dazu verwendet werden,

9. Konzeptionelle Einordnung

digitale Information zu verändern bzw. festzulegen. Jene Werkzeugtokens, die der Steuerung der Systemfunktionen dienen, sind hingegen als *Instruments* zu klassifizieren. *Lenses* und *Trays* kommen im Werkzeug nicht zum Einsatz.

Hinsichtlich der Metaphorik unterscheiden Ullmer und Ishii (1997) zwischen unterschiedlichen Abstraktionsebenen von Phicons (*generic – symbolic – model*), wobei im vorliegenden System ob der offenen Semantik die Modellierungstokens ausschließlich *generic Phicons* sind bzw. sein können. Die Werkzeugtokens sind zumeist als *symbolic Phicons*, im Falle des Löschtokens – dem Radiergummi – eher als *model Phyicon* zu klassifizieren.

9.3.2. Bewertung

Für die Bewertung des Werkzeugs ist vor allem dessen Gegenüberstellung zu den vorgeschlagenen Elementen einer „Interactive Surface“ von Interesse. Hier zeigt sich, das die unterschiedlichen Arten von Tokens, die im Werkzeug eingeführt wurden, feingranular auf die unterschiedlichen Element-Arten von (Ishii und Ullmer, 1997) abbildbar sind. Insbesondere die explizite Unterscheidung zwischen *Phandles* und *Instruments* ist eine Alleinstellungsmerkmal der hier vorgeschlagenen Systematik.

Eine mögliche Lücke, die Erweiterungspotential für das Werkzeug anzeigen könnte, ist die Abwesenheit von TUI-Elementen, die als *Lenses* oder *Trays* zu klassifizieren sind. Insbesondere *Trays* erscheinen für die explizite Interaktion mit einzelnen Tokens – etwa der Benennung oder der Einbettung von Zusatzinformation – als geeignet. Die dazu notwendigen Interaktionsabläufe würden expliziter auf den Vorgang der Zuordnung von Information eingehen und sich stärken von anderen Interaktionen unterscheiden, die anderen Zwecken, z.B. der Herstellung von Verbindungen zwischen Modellierungstokens, dienen.

9.4. Einordnung in das Ordnungssystem von Holmquist et al.

Grundlage der Einordnung in diesem Abschnitt ist der Ansatz von (Holmquist et al., 1999), der in Abschnitt 5.2.4 beschrieben wurde.

9.4.1. Abbildung

Die von Holmquist et al. verwendete Terminologie ist im Wesentlichen direkt auf jene abbildbar, die in dieser Arbeit verwendet wurde. Die Modellierungstokens und einbettbaren Tokens entsprechen im Wesentlichen *Tokens*. Dies ist dadurch begründbar, dass die Art eines Modellierungstokens in einem Modell immer im gleichen Zusammenhang mit der Art der Information steht, die durch dieses repräsentiert wird. Eine Eigenschaft, die eher *Containern* zuzuordnen ist, ist jedoch die dynamische Festlegbarkeit der Bedeutung einer Art von Modellierungstokens - die physischen Elemente ansich sind vor Beginn der Modellbildung generisch (also *Container*), werden aber im Zuge der Modellierung mit Bedeutung belegt (die dann für alle Instanzen dieser Art von Modellierungstokens gilt) und sind dann eher als *Tokens* zu klassifizieren.

Die Werkzeugtokens des hier vorgestellten Systems entsprechen in ihrer Konzeption den *Tools*. Sie manipulieren digitale Information, lösen Aktionen aus oder versetzen das System in einen anderen Zustand und entsprechen damit exakt der Definition von *Tools*, die von den Autoren gegeben wird.

Information Faucets sind im Kontext des hier vorgestellten Systems einerseits die Tischoberfläche, über die Information zu Modellierungstokens abgerufen werden kann, andererseits ist die Registrierungskamera ein klassisches Faucet im Sinne der Definition, da sie dem Abruf oder der Assoziation von Information an ein Token dient, sobald dieses in den Erfassungsbereich der Kamera gerät.

9.4.2. Bewertung

Die konzeptionellen Elemente des hier vorgestellten Systems sind also auf das Ordnungsstem von Holmquist et al. (1999) abbildbar. Die Problematik der nicht eindeutigen Zuordnung von Modellierungstokens zur Kategorie *Tokens* oder *Constraints* ist einerseits auf eine der grundlegenden Design-Paradigmen des hier entwickelten Werkzeugs – der Flexibilität der Abbildung – zurückzuführen, weist aber andererseits auch auf mögliches Verbesserungspotential hin.

Durch die Flexibilisierung nicht nur der Bindung zwischen physischen Elementen und digitaler Repräsentation sondern auch der Verwendung von unterschiedlichen physischen Elementen selbst könnten Modellierungstokens eher *Token*-artiger werden. Indem Modellierende eigenen physische Elemente (auf ihrem Arbeitskontext) einbringen können, könnte die Erfassbarkeit der Bedeutung der physischen Repräsentation unter Umständen verbessert werden können.

9.5. Einordnung in das Object-Meaning-Kontinuum

Grundlage der Einordnung in diesem Abschnitt ist der Ansatz von Underkoffler und Ishii (1999), der in Abschnitt 5.2.5 beschrieben wurde.

9.5.1. Abbildung

Das Object-Meaning-Kontinuum ist eines der ersten Ansätze, die die physischen Objekte eines TUI nicht strikten Kategorien zuordnen sondern auf einem Kontinuum anordnen. Dabei wird kein kategorischer Unterschied zwischen informationsrepräsentierenden Objekten und Werkzeug-Objekten gemacht – sowohl in der Mitte des Kontinuums als auch an den Enden verschwimmt die Grenzen zwischen Objekt als reiner Repräsentation und reinem Werkzeug.

In der Folge werden die Objekte des Werkzeugs in das Kontinuum eingeordnet, wobei zur einfacheren Anwendbarkeit die entlang des Kontinuums von den Autoren definierten Ausprägungen verwendet werden.

Die Ausprägung *object as noun* kommt im Werkzeug nicht zum Einsatz. Keines der physischen Elemente hat eine direkte Entsprechung in der realen Welt – durch die geforderte Flexibilität der Abbildung wäre das auch nicht möglich.

Die Elemente die zur Modellbildung verwendet werden – also Modellierungstokens und einbettbare Tokens – sind der Ausprägung *object as attribute* zuzuordnen, da in die Farbe bzw. Form der Tokens Bedeutung (nämlich die Semantik des jeweiligen Art von Tokens) codiert ist. Bei Modellierungstokens ist zu beachten, dass die Zuordnung der Bedeutung dynamisch zu Laufzeit erfolgt, der physischen Eigenschaft des Objekts also erst durch die Benutzer konkret Bedeutung zugewiesen wird.

Keines der Objekte des Werkzeug ist aufgrund seines Designs der Ausprägung *object as pure object* zuzuweisen. Diese Zuordnung kann jedoch dynamisch bei der Modellbildung für Modellierungstokens eintreten, wenn die Benutzer den unterschiedlichen Objektarten keine Bedeutung zuordnen und diese beliebig mit Information belegen.

Die Tokens, die im Werkzeug nicht unmittelbar zur Modellbildung verwendet werden, verteilen sich zwischen den beiden funktional abstrahierten Ausprägungen des Kontinuums. Der Ausprägung *Object as Verb* sind das Historiensteuerungs-Token und das Markierungstoken zur Herstellung einer gerichteten Verbindung zuzuordnen. Für das Historiensteuerungs-Token gilt dies, da dessen Drehbewegung zur zeitliche Navigation auf die Bewegungen eines Uhrzeigers abbildbar ist. Das

Markierungstoken zur Herstellung einer gerichteten Verbindung weist eine Pfeilspitze als Grundfläche auf, wodurch eine physische Eigenschaft Hinweise auf die Funktion des Tokens gibt (hier allerdings grenzwertig, da die dreieckige Grundfläche nicht eindeutig als Pfeilspitze zu erkennen ist). Die übrigen Tokens sind eher im Bereich des *object as reconfigurable tool* anzusiedeln, da ihrer äußere Form oder ander physische Eigenschaften keine Hinweise auf deren Funktionalität geben. Dies gilt für die allgemeinen Markierungstokens, das Snapshot-Token und das Wiederherstellungs-Token.

Einen Spezialfall bildet das Löschtoken, das mit dem Radiergummi als Repräsentation eine *object as verb*-Einordnung suggeriert (Radiergummi zum Löschen von Verbindungen), tatsächlich das System aber lediglich in einen Löschmodus versetzt, in dem Verbindungen mittels anderer Interaktionsabläufe gelöscht werden können. Hinsichtlich seiner tatsächlichen Verwendung ist das Löschtoken also als *object as reconfigurable tool* zu klassifizieren.

9.5.2. Bewertung

Das von den Autoren vorgeschlagene Kontinuum eignet sich, um die Elemente eines TUI hinsichtlich deren Bedeutung und Verwendung einzurichten. Diese Einordnung kann nützlich sein, um Elemente zu identifizieren, deren tatsächliche Verwendung im TUI nicht mit der wahrgenommenen Bedeutung übereinstimmt. Dazu müssen die Elemente unabhängig von der konkreten Implementierung klassifiziert werden (ggf. von nicht am Design und der Entwicklung beteiligten Personen) und das Ergebnis der umgesetzten Funktionalität gegenübergestellt werden.

Für das hier vorgestellte Werkzeug ist eine derartige Diskrepanz wie oben bereits beschrieben am Löschtoken zu erkennen. Dieses suggeriert eine Verwendbarkeit im Sinne von *object as verb*, setzt aber tatsächlich die augenscheinliche Funktion (Löschen) nicht um (bzw. ist auf eine andere Funktion – Löschmodus aktivieren – abgebildet) und ist deshalb lediglich als *object as reconfigurable tool* einzurichten. Eine „Aufwertung“ des Löschtokens im Sinne einer Hinterlegung mit der tatsächlichen Lösch-Funktion würde eine erwartungskonforme Verwendbarkeit eher sicherstellen und so zur Verbesserung des Gesamtsystems beitragen.

9.6. Betrachtung im Lichte des MCRpd-Modells

Grundlage der Einordnung in diesem Abschnitt ist der Ansatz von Ullmer und Ishii (2000), der in Abschnitt 5.2.6 beschrieben wurde.

9.6.1. Abbildung

Wird das erstellte Werkzeug dem MCRpd-Modell gegenübergestellt, so ist erkennbar, dass die Eigenschaften des Werkzeugs augenscheinlich nicht den Anforderungen des MCRpd-Modells an ein Tangible User Interface genügen. Das Werkzeug verfügt über einen Ausgabekanal – den Bildschirm – der nicht an die physische Repräsentation gekoppelte ist. Bei näherer Betrachtung erscheint eine vollständige Einordnung jedoch argumentierbar. All jene Interaktionen, die mit der eigentlichen Modellierung zusammenhängen, genügen den Anforderungen des MCRpd-Modells ohne Einschränkungen. Die Manipulation des *Model* (im MCRpd-Modell) erfolgt über die Tischoberfläche, die gleichzeitig dazu verwendet wird, den Systemzustand zu manifestieren. Dem MCRpd-Modell entgegenzulaufen scheinen jene Interaktionsabläufe, die den sekundären Ausgabekanal einbeziehen. Dabei sind zwei Fälle zu unterscheiden. Bei der Einbettung von Information in Modellelemente wird die sekundäre Oberfläche zur Auswahl der anzubindenden Ressource und damit als GUI benutzt. Eine Einordnung in das MCRpd-Modell ist hier damit nicht möglich. Bei der Betrachtung der Modellierungshistorie wird die sekundäre Oberfläche als alleiniges Ausgabemedium benutzt, der Systemzustand wird durch das runde Navigationstoken auf der Oberfläche beeinflusst. Dies verletzt grundsätzlich den Aufbau des MCRpd-Modells, betrachtet man jedoch die daraus abgeleiteten Kern-Charakteristika von TUIs, so kann festgestellt werden, dass diese dennoch nicht verletzt sind. Das in Frage zu stellende Charakteristikum ist jene mit der Forderung nach Kopplung zwischen der physischen Repräsentation des *Models* (*REP-P*) und der intangiblen, digitalen Manifestation von Modellaspekten in der realen Welt (*REP-D*). Die Autoren fordern von dem Zusammenhang zwischen *REP-P* und *REP-D* jedoch ausschließlich, dass er „*perceptually coupled*“ sein müsse, die Kopplung also von den Benutzern als solche wahrgenommen werden müsse. Betrachtet man das runde Navigationstoken als *REP-P* und die Ausgabe am sekundären Ausgabekanal als *REP-D*, so ist diese Kopplung feststellbar, da sich *REP-D* immer in Abhängigkeit von *REP-P* verändert. Insofern ist das MCRpd-Modell nicht verletzt, das Werkzeug weist die von den Autoren als Kern-Charakteristika von Tangible User Interfaces bezeichneten Eigenschaften auf.

Hinsichtlich der Kategorien von TUIs, die von den Autoren festgelegt werden, ist das System der Kategorie *relational* zuzuordnen. Das hier vorgestellte Werkzeug ist nicht *spatial*, da die Position der verwendeten Tokens relativ zum Referenzrahmen (der Tischoberfläche) keine spezifische Bedeutung haben. Die Bedeutung ist viel mehr in den Beziehungen der Tokens untereinander codiert, was wiederum für ein *relationales* System sprechen würde. Gleichzeitig kann damit die Kategorie *associative* ausgeschlossen werden, da in System dieser Art keine Beziehungen zwischen Tokens berücksichtigt werden. Da die Beziehungen zwischen Tokens nur digital und

nicht physisch abgebildet werden, ist die Bedingung für ein *konstruierendes* System nicht erfüllt. *Constructive* wäre das Werkzeug dann, wenn der Modellzustand vollständig durch physische Elemente und Verbindungen abgebildet wäre.

9.6.2. Bewertung

Das *MCRpd*-Modell ist ein im Vergleich zu anderen Ansätzen eher abstraktes, konzeptionelles Modell zur Beschreibung eines Tangible User Interfaces. Trotzdem – oder auch deswegen – eignet es sich gut zur Reflexion der Eigenschaften eines TUIs bzw. zur Prüfung der Konsistenz der vorgesehenen Benutzerinteraktionen.

Das Werkzeug konnte in die Logik des Modells eingeordnet werden, wobei bei der Beschreibung der Interaktion zur Steuerung der Modellierungshistorie verstärkter Argumentationsbedarf herrschte. Dies kann auf eine möglicherweise zu schwache Kopplung zwischen *REP-P* und *REP-D* hinweisen. Tatsächlich wird bei der Kontrolle der Modellierungshistorie auf der Tischoberfläche kein Feedback ausgegeben, ob das Steuerungs-Token erkannt wurde und in welchem Zustand es sich aktuell befindet. Die Kopplung könnte etwa in Form einer Darstellung des aktuell dargestellten Zeitpunkts in der Modellierungshistorie rund um das Kontroll-Token angezeigt werden, was die Kopplung zwischen den beiden Komponenten der Repräsentation verstärken würde.

9.7. Einordnung in den Tokens+Constraints Kontext

Grundlage der Einordnung in diesem Abschnitt ist der Ansatz von (Ullmer et al., 2005), der in Abschnitt 5.2.7 beschrieben wurde.

9.7.1. Einordnung

Eine unmittelbare Einordnung des hier vorgestellten Werkzeugs in den Tokens+Constraints-Ansatz ist nicht möglich, da einerseits kein allgemeines Schema zur Betrachtung vorgeschlagen wird und das Werkzeug seiner Konzeption nach nicht dem Verständnis eines Tokens+Constraints-System nach (Ullmer et al., 2005) entspricht. Dazu müsste es physische Constraints aufweisen, die den Interaktionsraum der informationstragenden Tokens physisch einschränken. Das einzige nach dieser Definition identifizierbare Constraint des Werkzeugs ist der aktive Bereich der Tischoberfläche, die den Modellierungsraum beschränkt. Diese Einschränkung ist aber im Ver-

9. Konzeptionelle Einordnung

gleich zu den Beispielen für Constraints, die die Autoren angeben, eher wenig strikt und lässt viel Interaktionsraum.

Am ehesten ist das vorliegende System als eine „interactive surface“ mit Aspekten einer „constructive assembly“ zu klassifizieren. Die Qualifikation als „interactive surface“ erscheint ob der Interaktion mit physischen Blöcken auf einer digital augmentierten Oberfläche naheliegend. „Constructive assembly“-Aspekte sind im Bereich der Einbettung von informationstragenden Tokens in Modellierungs-Tokens zu finden. Die Zuordnung wird dabei durch das Hineinlegen eines Tokens in ein anderes ausgedrückt, wodurch die konkrete Semantik in der Beziehung zwischen den beiden Tokens abgebildet ist. Generell wird die Semantik des abzubildenden Modells in der räumlichen und logischen Konfiguration der Modellierungs-Tokens zueinander abgebildet, was ebenfalls einem „constructive assembly“-Aspekt entspricht.

Nicht unmittelbar in Zusammenhang mit dem Tokens+Constraints-Ansatz stehen die fünf Fragen von (Bellotti et al., 2002), die beim Design einer Benutzungsschnittstelle berücksichtigt werden sollten. In Ullmer et al. (2005) werden diese Fragen für den dort vorgestellten Ansatz beantwortet, an dieser Stelle sollen sie im Lichte des hier vorgestellten Systems betrachtet werden.

Address Das System interpretiert alle Interaktionen, die mit Modellierungs- oder Werkzeugtokens unmittelbar auf der Tischoberfläche ausgeführt werden, als an es gerichtet. Andere Interaktionen werden ignoriert und können auch technisch nicht erfasst werden.

Attention Der Einsatz jedes Tokens löst unmittelbar eine Reaktion auf den Ausgabekanälen des Systems aus. Benutzer erhalten also direktes Feedback auf erkannte Interaktionen. Eine Verzögerung zwischen Ein- und Ausgabe tritt lediglich bei der Markierung von Elementen auf, die zur Robustheit gegen Fehlerkennungen erst erfasst wird, wenn das Eingabe-Token länger als 500 ms vorhanden ist.

Action Befehle an das System können generell an einem beliebigen Punkt der Oberfläche abgesetzt werden, sofern es sich um allgemeine Kommandos handelt, die das Gesamtsystem betreffen. Befehle, die einem bestimmten Objekt zuzuordnen sind (z.B. Auswahl oder Verbindungsherstellung) werden durch räumliche Nähe zugeordnet.

Alignment Nach Ausführung einer Aktion befindet sich das System immer in einem stabilen Zustand, anhand dessen Visualisierung die Benutzer erkennen können, ob die intendierte Aktion korrekt erkannt und ausgeführt wurde.

Accident Missverständnisse zwischen System und Benutzern können auf unterschiedliche Arten aufgelöst werden. Bei Benutzeraktionen, die mit einem Timeout belegt sind (z.B. Auswahl) ist es ausreichend, dieses Timeout abzuwarten,

wodurch das System wieder in den Ausgangszustand wechselt. Missverständnisse, die zu permanenten Zustandsänderungen führen, können entweder explizit durch die gegenteilige Interaktion rückgängig gemacht werden (z.B. Löschen einer versehentlich hergestellten Verbindung) oder mittels der Wiederherstellung eines gespeicherten Systemzustandes korrigiert werden, in dem das Missverständnis noch keine Auswirkung hatte bzw. nicht aufgetreten war.

9.7.2. Bewertung

Der Tokens+Constraints-Ansatz ermöglicht in der vorliegenden Form keine direkte Abbildung des Werkzeugs auf seine Konzepte. Wertvoller für die Betrachtung sind an dieser Stelle die Ordnungsschemata und Fragestellungen, die von Ullmer et al. (2005) im Kontext des Ansatzes erarbeitet bzw. beantwortet werden.

Von Interesse sind insbesondere die fünf Fragen für das Design von Benutzungsschnittstellen, die von (Bellotti et al., 2002) gestellt werden. Bei der Beantwortung dieser Fragen für das vorliegende System ist durchaus Verbesserungspotential zu identifizieren. Am offensichtlichsten zeigt sich das am Beispiel des Feedbacks des Systems an Benutzer über einen erkannten Interaktionswunsch. Hier kommt es beim Einsatz des Markierungstokens technisch bedingt zu Verzögerungen, die – ob der ansonsten unmittelbaren Reaktion des Systems – Unsicherheit bei den Benutzern erzeugen kann. Auch die Auflösung von Missverständnissen ist im Moment sub-optimal gelöst, da in jedem Fall entweder Zeitverlust auftritt oder mindestens zwei Interaktionsschritte zur Korrektur einer Fehlinterpretation notwendig sind. Hier wäre unter Umständen die Einführung einer expliziten „Undo“-Funktionalität sinnvoll.

9.8. Einordnung in das Framework nach Koleva et al.

Grundlage der Einordnung in diesem Abschnitt ist der Ansatz von (Koleva et al., 2003), der in Abschnitt 5.2.8 beschrieben wurde.

9.8.1. Abbildung

Das Framework eignet sich zur Einordnung einzelner Aspekte eines Tangible User Interfaces, aufgrund seiner Ausrichtung auf die Brücke zwischen realer und digitaler Welt insbesondere für die Betrachtung der eingesetzten Tokens und deren Verwendung zur Repräsentation und Manipulation des Systemzustandes. In Tabelle 9.1

9. Konzeptionelle Einordnung

werden die Tokens in die Kategorien entlang des Kohärenz-Kontinuums eingeordnet und hinsichtlich der Eigenschaften ihrer Brückenfunktion in die digitale Welt betrachtet.

Tabelle 9.1.: Beurteilung des Werkzeugs hinsichtlich des Degree of Coherence

Element	Kategorie	Trans- for- mation	Sensing of Interaction	Konfig- urierbar- keit	Lebens- dauer	Auto- nomie
Modell- ierungs- token	Proxy	lit.	X-Y-Position und Rotation, Öffnungssta- tus	fixiert	temp.	abh.
einbett- bares Token	Identifier	transf.	Präsenz, Con- tainer	fixiert	temp.	unabh.
Mark- ierungs- token	Specialized Tool	transf.	X-Y-Position	fixiert	temp.	unabh.
Löschtoken	Projection	transf.	Präsenz	fixiert	perm.	unabh.
Snapshot- token	Specialized Tool	transf.	Präsenz	fixiert	perm.	unabh.
Historien- navigations- token	Specialized Tool	transf.	Rotation	fixiert	perm.	unabh.
Wieder- herstellungs- token	Specialized Tool	transf.	Präsenz	fixiert	perm.	unabh.

lit.literally, transf.transformed, konfig.konfigurierbar, temp.temporär,
perm.permanent, abh.abhängig, unabh.unabhängig

Die Kardinalität wurde hier nicht gesondert betrachtet, die die Kardinalität immer 1:1 ist, also eine eindeutige Zuordnung zwischen realem Objekt und digitaler Repräsentation gegeben ist. Im Übrigen verzichten auch Koleva et al. (2003) auf die Einordnung in diese Kategorie, da sie generell nur geringen Unterscheidungswert hat. Hinsichtlich der Source of Link, die in der Tabelle ebenfalls nicht angegeben ist (und von den Autoren ebenfalls nicht verwendet wird), ist zu erwähnen, dass das Werkzeug durchaus einen Aspekt aufweist, bei dem der Source of Link die digitale Welt ist. Im Rahmen der Wiederherstellungsunterstützung gibt das System Anweisungen zur Manipulation der realen Welt, wodurch sich der Informationsfluss umkehrt. Da jedoch kein physisches Element direkt manipuliert wird, ist eine Einord-

nung in das oben angeführte Schema nicht möglich (der Link ist lediglich indirekt vorhanden).

9.8.2. Bewertung

Das von (Koleva et al., 2003) vorgeschlagene Framework ermöglicht die Klassifikation eines Tangible Interfaces über den Aspekt der Stärke der Bindung zwischen digitaler und realer Welt. Die Autoren nehmen damit eine zu diesem Zeitpunkt neue Perspektive ein, der noch keine große Aufmerksamkeit geschenkt wurde. Durch die Vernachlässigung der Interaktion am Tangible User Interface stellt eine Analyse unter Einsatz der im Framework vorgeschlagenen Kategorien und Merkmalen nur einen Teilaspekt des Gesamtsystems dar. Trotz dieser Einschränkung stellt das Framework ob seiner detaillierter Betrachtung der Eigenschaften der Verknüpfung von physischen Objekten mit digitaler Information einen potentiellen Mehrwert dar beim Design oder der Analyse von TUIs dar.

Insbesondere ermöglicht das Framework, nicht ausgeschöpftes Kohärenz-Potential zu identifizieren. Im konkreten Fall des hier vorgestellten Werkzeugs lässt sich das am Beispiel des Löschtokens zeigen. Dieses physisch durch einen Radiergummi repräsentierte Token wird im Moment lediglich als Schalter verwendet. Das System wird in den Löschmodus versetzt, sobald das Token auf der Oberfläche erkannt wird. Das Token ist als *Projection* einzufordern, die das Token mit der Information des aktivierten oder deaktivierten Löschmodus verbindet. Obwohl hoch kohärent, ist das Token trotzdem suboptimal eingesetzt, da es in der Praxis als Werkzeug wahrgenommen wird, das zum Löschen einer spezifischen Verbindung verwendet werden kann (*Specialized Tool*). An diesem Beispiel lassen sich zwei Aspekte zeigen, die bei der Verwendung des Frameworks beachtet werden müssen. Zum einen ist hohe Kohärenz nicht für jeden Anwendungsfall anstrebenswert, da bei Werkzeugen im Allgemeinen eine nicht permanente Bindung verwendet wird. Zum anderen zeigt sich die Unvollständigkeit der Analyse mittels dem Framework, da die Metaphorik des physischen Elements, also seine Bedeutung in der Interaktion, nicht berücksichtigt wird. Beide Aspekte – Kohärenz und Metaphorik – berücksichtigt erst (Fishkin, 2004) in der von ihm vorgeschlagenen Taxonomie (siehe Abschnitt 5.2.11 und 9.11).

9.9. Spezifikation des TAC-Schemas nach Shaer et al.

Grundlage der Einordnung in diesem Abschnitt ist der Ansatz von Shaer et al. (2004), der in Abschnitt 5.2.9 beschrieben wurde.

9.9.1. Abbildung

Das „Token and Constraints“-Schema (TAC) erlaubt es, ein Tangible Interface sowohl hinsichtliche dessen Struktur als auch dessen Verwendung zu beschreiben. In Tabelle 9.2 wird das Schema auf das hier vorgestellte Werkzeug angewandt.

Tabelle 9.2.: Spezifikation des Werkzeug mittels TAC-Schema

TAC	Struktur			Verhalten	
	Token	Constraint	Variable	Aktion	Feedback
1	Modellierungs-token	Oberfläche	Modell-element	Auflegen	Modellelement anzeigen
				Bewegen	Modellelement bewegen
				Entfernen	Modellelement entfernen
2	einbettbares Token	Modellierungs-token	Modell-element	Hineinlegen	Daten einbetten
				Herausnehmen	Container-Kopplung aufheben
3	einbettbares Token	Registrierungskamera	einbettbares Modell-element	Vor die Kamera halten	ungebunden: Datenbindung auslösen; gebunden: Gebundene Daten anzeigen
4	Markierungs-token	Modellierungs-token	Modell-element	Neben Modellierungs-token platzieren	Markierung anzeigen
5	Markierungs-token	Modellierungs-token, markiertes Modellierungs-token	Verbindung	Neben unmarkiertem Modellierungs-token platzieren	Verbindung herstellen und anzeigen

Tabelle 9.2.: (Fortsetzung)

TAC	Struktur		Verhalten		
	Token	Constraint	Variable	Aktion	Feedback
6	Tastatur	markiertes Modellierungs-token	Modell-element	Tastatur-eingabe	Benennung des markierten Modellelements
7	Tastatur	Verbindungen, kein markiertes Modellierungs-token	zuletzt hergestellte Verbindung	Tastatur-eingabe	Benennung der zuletzt hergestellten Verbindung
8	Löschtoken	Oberfläche	Modell-element	Auflegen	Löschmodus aktivieren
				Entfernen	Löschmodus deaktivieren
9	Snapshot-token	Oberfläche	Modellierungs-historie	Auflegen	Aktuellen Modellzustand sichern, Blitz anzeigen
10	Historienkontroll-token	Oberfläche	Modellierungs-historie	Auflegen	Letzten gespeicherten Snapshot anzeigen
				Drehen	Durch die gespeicherten Snapshots navigieren
				Entfernen	Aktuelles Modell anzeigen
11	Wiederherstellungs-token	Oberfläche, vorhandenes Historienkontroll-token	Modell-zustand	Auflegen	Aktuell angezeigten Snapshot wiederherstellen

Jene Interaktionsabläufe, bei denen das System die Aktionen der Benutzer anleitet (z.B. bei der Unterstützung der Wiederherstellung) können in diesem Schema nicht abgebildet werden, da die Constraints keine physischen Objekte sondern lediglich projizierte Information sind.

9.9.2. Bewertung

Das TAC-Schema eignet sich für eine umfassende Spezifikation des Struktur und des Verhaltens eines Tangible User Interfaces. Das vorgeschlagene Schema geht jedoch (wie die meisten anderen Ansätze auch) davon aus, dass das TUI vor allem zur Informationseingabe verwendet wird und das sich der Systemzustand und dessen Manifestierung am Interface in Abhängigkeit dieser Eingaben ändern. Nicht abbildbar sind Interaktionen, die vom System ausgelöst bzw. kontrolliert werden, bei denen also die *Variable* das aktive und nicht das manipulierte Element ist (im Gegensatz zum zuvor vorgestellten *Degree of Coherence*-Ansatz der mit der *Source of Link*-Eigenschaft explizit auf diesen Aspekt eingeht – siehe Abschnitt 5.2.8).

Entsprechend dieser Einschränkung eignet sich das TAC-Schema weitgehend für die Spezifikation des hier vorgeschlagenen Werkzeuges. Lediglich die Wiederherstellungsunterstützung kann nicht abgebildet werden, da sie vom System gesteuert wird. Beim Einsatz zur Spezifikation eines TUI oder bei der Untersuchung desselben hinsichtlich möglichem Verbesserungspotential ist vor allem auf die möglichen Constraints eines Tokens zu achten. Dabei ist es hilfreich, unterschiedliche Constraints bezüglich der von ihnen vorgegebenen oder durch sie ermöglichten Aktionen zu betrachten. Als Beispiel im konkreten System kann wiederum das Löschtoken verwendet werden. Diese wird in der aktuellen Implementierung mit dem Constraint „Oberfläche“ verwendet, um den Löschmodus zu aktivieren (wenn es aufgelegt wird) bzw. zu deaktivieren (wenn es entfernt wird). Setzt man das Löschtoken nun in ein TAC mit dem Constraint „Verbindung“, ergeben sich neue Möglichkeiten der Interaktion. Ein Aufsetzen des Löschtokens auf eine Verbindung könnte diese unmittelbar löschen und würde so den notwendigen Interaktionsablauf massiv vereinfachen. Das mit diesem Constraint auch die Metapher des verwendeten Radiergummis sinnbringend verwendet wird, ist ein Nebeneffekt, der jedoch im TAC-Schema nicht repräsentiert wird. Vielmehr ist die Metaphorik Ausgangspunkt für eine sinnvolle und verständliche Auswahl möglicher Constraints für ein Token.

9.10. Einordnung in die Kategorien von TUI-Anwendungen

Grundlage der Einordnung in diesem Abschnitt ist der Ansatz von (Klemmer et al., 2004), der in Abschnitt 5.2.10 beschrieben wurde.

9.10.1. Abbildung

Das Werkzeug weist Aspekte einer *spatial application* auf, da es die Platzierung von physischen Elementen auf einer Oberfläche als Grundlage der Interaktion mit dem System heranzieht. Da aber die Beziehung zwischen den Elementen sowohl für die Informationsrepräsentation als auch für die Steuerung des Systems wesentlich ist, ist auch eine Einordnung in die Kategorie *topological application* argumentierbar. Letztendlich referenzieren manche physische Elemente auch auf digitale Information, was das Kriterium für die Einordnung in die Kategorie *associative application* ist. Lediglich die Kategorie *Forms* trifft nicht auf das hier vorgestellte Werkzeug zu.

9.10.2. Bewertung

Das hier vorgestellte Werkzeug lässt sich nicht eindeutig einer der von Klemmer et al. (2004) identifizierten Applikations-Kategorien zuordnen. Die (von den Autoren als solche bezeichnete) „Taxonomie“ eignet sich demnach nicht, um komplexere Systeme zu klassifizieren, die sowohl Informationsrepräsentation als auch Systemsteuerung durch physische Elemente abwickeln.

9.11. Einordnung in die Taxonomie von Fishkin

Grundlage der Einordnung in diesem Abschnitt ist der Ansatz von (Fishkin, 2004), der in Abschnitt 5.2.11 beschrieben wurde.

9.11.1. Abbildung

Das in dieser Arbeit entwickelte Werkzeug überspannt aufgrund seiner komplexen Struktur in beiden von Fishkin vorgeschlagenen Dimensionen zur Klassifikation von Tangible Interfaces mehrere Ausprägungen. Um eine umfassende und ins Detail gehende Einordnung vornehmen zu können, werden im Folgenden Einzelaspekte des Systems betrachtet und eingeordnet. Während die Dimension "Embodiment" bereits in Kapitel 7 betrachtet wurde, um eine strukturierte Zuordnung der Ausgabekanäle vornehmen zu können, werden hier die einzelnen Funktionalitäten des Systems (siehe Abschnitt 6.3) jeweils beiden Dimensionen zugeordnet (siehe Tabelle 9.3)

9. Konzeptionelle Einordnung

Tabelle 9.3.: Einordnung des Systems in die Taxonomie nach Fishkin

	Embodiment	Metaphor
Platzieren und Benennen von Modellelementen	distant, nearby (Tastatur), full (Haftnotiz)	verb (Tastatur), verb + noun (Haftnotiz)
Erstellen von Verbindern	nearby	verb bis noun+verb (Werkzeug- tokens), verb (räumliche Nähe)
Löschen von Verbindern	environmental bis nearby	noun
Einbetten von Information	full	noun + verb
Abrufen von Information	distant	verb
Erstellen von Snapshots	environmental bis nearby	none
Navigation in der Modell-Historie	distant	verb
Wiederherstellen eines Modell- Zustandes	nearby	noun + verb

Beim *Platzieren und Benennen von Modellelementen* ist die Benennung auf zwei Arten möglich, die unterschiedlich in die Taxonomie einzuordnen sind. Bei Benennung mittels Auswahl und Tastatur ist durch die Projektion der Benennung die Embodiment-Ausprägung "nearby" zu wählen. Der Vorgang der Auswahl und Benennung kann als analog zur realen Welt gesehen werden, die eingesetzten Werkzeuge sind aber generischer Natur – Metaphor ist also als "verb" zu klassifizieren. Bei der Benennung mittels Haftnotitz ist durch die unmittelbar auf den Tokens angebrachten Benennungen Embodiment "full", Der Vorgang des Beschriftens wird analog zur realen Welt durchgeführt, auch die Informationsträger (Haftnotizen) entsprechen jenen der realen Welt, Metaphor ist also "verb + noun", wobei der notwendige Vorgang der expliziten Erfassung einer Beschriftung durch das System eine Klassifikation "full" verhindert und sogar die Einstufung "noun + verb" etwas abschwächt (keine Analogie des Vorgangs zur realen Welt).

Zur *Herstellung von Verbindern* existieren ebenfalls zwei Möglichkeiten. In beiden Fällen ist durch die Projektion der Verbindung die Ausprägung in Embodiment "nearby", sie unterscheiden sich jedoch hinsichtlich "Metaphor". Bei der Verwendung von Werkzeugtokens ist der Vorgang der Auswahl der Endpunkte analog zur realen Welt zu sehen und somit als "verb" einzustufen. Die Verwendung von spe-

zifischen Werkzeugtokens zur Herstellung gerichteter Verbinder zeigt sogar Züge von "noun + verb", da die durch das Token dargestellte Pfeilspitze eine Analogie zur realen Welt bildet.

Das *Löschen von Verbindern* wird durch das Lösch-Token vorgenommen. Dieses ist durch einen Radiergummi symbolisiert, der jedoch nicht als solche eingesetzt wird sondern das System nur in einen Löschmodus versetzt. Die Klassifikation in Metaphor ist demnach "noun". Die Visualisierung des Löschezustandes erfolgt unspezifisch durch die Umfärbung der gesamten Tischoberfläche, womit ein Embodiment von "nearby" oder "environmental" (aufgrund der Unspezifität) gerechtfertigt wäre.

Einbetten von Information erfolgt durch die Verwendung der Modellierungstokens als Container und Hineinlegen von kleineren Tokens. Embodiment ist in diesem Fall "full", die die Einbettung physisch nachvollzogen wird. Metaphor ist durch die Analogie des "Hineinlegens" von Information in "Container" in die Ausprägung "noun + verb" einzuordnen.

Das Abrufen von Information wird über den sekundären Ausgabekanal abgewickelt und ist daher in Embodiment als "distant" einzuordnen. Der Vorgang des Herausnehmens von Information aus einem Container existiert analog zur realen Welt, das bei diesem Vorgang im Zentrum stehende Objekt, das einbettbare Token, ist jedoch generisch und weist nicht auf die Art der eingegebenen Information hin. Eine Klassifikation von "verb" in Metaphor erscheint daher gerechtfertigt.

Beim Erstellen von Snapshots wird die gesamte Tischoberfläche als Feedbackkanal genutzt. Insofern ist Embodiment wie im Falle des Löschens von Verbindern im Bereich "environmental" bis "nearby" anzusiedeln. Das Snapshot-Token selbst ist ein generisches Objekt, das keine Analogie zur realen Welt aufweist. Metaphor ist daher "none".

Die Navigation in der Modell-Hierarchie erfolgt mit dem runden Navigations-Token. Zur Ausgabe der gespeicherten Modell-Zustände wird der sekundäre Ausgabekanal verwendet. Embodiment ist deshalb "distant". Metaphor beschränkt sich auf "verb", da der Drehvorgang zur Navigation analog zum Einstellen einer Uhr erfolgt, das Token selbst aber bis auf seine runde Form generisch ist.

Das Wiederherstellen eines Modellzustandes erfolgt durch spezifische Anweisungen auf der Modellierungsoberfläche. Embodiment ist also als "nearby" einzustufen. Der Vorgang der Wiederherstellung erfolgt durch Verschieben der Modellierungstokens, was im Wesentlichen analog zur realen Welt abläuft. Da unmittelbar die Objekte manipuliert werden, kann Metaphor als "noun + verb" eingestuft werden.

9.11.2. Bewertung

Die Taxonomie nach Fishkin ermöglicht eine strukturierte Erfassung einzelner Aspekte eines Tangible User Interfaces. Eine aussagekräftige Gesamteinordnung ist nur bei einfachen TUIs möglich, komplexe, mit vielen Interaktionsmöglichkeiten ausgestattete Systeme tendieren dazu, ein sehr breites Spektrum der Taxonomie abzudecken. Für die detaillierte Betrachtung eines komplexen Gesamtsystems erscheint die Taxonomie dennoch geeignet, da einerseits aus den einzelnen Teileinordnungen für den jeweiligen Anwendungsfall ggf. Verbesserungspotentiale abgeleitet werden können und andererseits (nach der Betrachtung des hier entwickelten Systems) scheint, als ob ein die Taxonomie breit abdeckendes Gesamtsystem potentiell Inkonsistenzen im Interaktionsdesign aufweist bzw. unterschiedliche Interaktionsparadigmen vermischt wurden. Vor allem "Ausreißer" aus einem vorwiegend einheitlichen Gesamtbild scheinen einer näheren Betrachtung hinsichtlich eines möglichen Redesigns wert.

Konkret können diese Vermutungen im vorliegenden System vor allem an der Konzeption des Lösch-Tokens und des Snapshot-Tokens festgemacht werden. Der Großteil der Interaktionen mit dem System beinhaltet in der Dimension Metaphor den "verb"-Aspekt (zu etwa gleichen Teilen ausschließlich und in der Kombination mit "noun"). Die Funktionalitäten, die die beiden erwähnten Tokens einbeziehen, laufen diesem Trend entgegen und zeigen in Metaphor die Ausprägung "noun" bzw. "none". Tatsächlich zeigt sich in der Praxis, dass die Anwendbarkeit dieser Tokens von Benutzern missverstanden bzw. nicht verstanden wird. Ein Redesign dieser Tokens mit expliziterer bzw. eher aktivitätsorientierter Metaphor erscheint deshalb untersuchenswert.

Zusammenfassend scheint die Taxonomie vor allem im Zusammenhang mit der Sicherung von konsistenter Interaktion an der Benutzungsschnittstelle sinnvoll anwendbar zu sein. Der Mehrwert des Ansatzes zeigt sich hier nicht so sehr in den absoluten Ausprägungen auf den beiden Dimensionen sondern vielmehr in den relativen Unterschieden, die zwischen den einzelnen Teilen des Tangible User Interfaces auftreten.

9.12. Betrachtung im Lichte der Retrospektive von Ishii

Grundlage der Einordnung in diesem Abschnitt ist der Ansatz von (Ishii, 2008), der in Abschnitt 5.2.12 beschrieben wurde.

9.12.1. Einordnung

Ishii (2008) spricht in seiner umfassenden Darstellung der Entwicklung des Forschungsgebiets „Tangible User Interfaces“ eine Vielzahl von Aspekten an, die der Konzeptbildung im Gebiet dienlich sind. Er spart jedoch Ansätze aus, die analytisch oder von Seiten der Spezifikation an TUIs herangehen. Trotzdem ist eine Einordnung in die unterschiedlichen angesprochenen Aspekte zum Teil möglich. Ein Großteil der Ergebnisse wurden ob des zusammenfassenden Charakters des Artikels bereits in früheren Abschnitten bearbeitet und argumentiert. Diese werden hier nur noch zusammenfassen erwähnt.

Für die Einordnung des Systems in das MCRit-Framework sei an dieser Stelle auf die bis auf die veränderte Nomenklatur identische Betrachtung des MCRpd-Frameworks in Abschnitt 9.6 verwiesen.

Hinsichtlich der Kategorisierung von TUIs ist das System identisch zu den in Abschnitt 9.7 Kategorien einzuordnen. Die bei (Ishii, 2008) angegebenen Kategorien stellen lediglich eine Erweiterung (bzw. Verbreiterung des Betrachtungsbereichs) der in (Ullmer et al., 2005) identifizierten Kategorien dar. Die zusätzlichen Kategorien haben jedoch für das hier vorgestellte Werkzeug keine Relevanz, so dass eine weiterführende Neubewertung unterbleiben kann.

Die von Ishii angeführten grundlegenden Eigenschaften und Merkmale eines TUI ermöglichen in Ermangelung der Angabe von konkreten Merkmalsausprägungen oder Beurteilungskriterien keine Einordnung des Werkzeugs. Die angegebenen Aspekte überschneiden sich jedoch stark mit den Eigenschaften und Merkmalen der in Abschnitt 9.1 und 9.2 betrachteten Ansätze von (Fitzmaurice et al., 1995) und (Fitzmaurice, 1996).

9.12.2. Bewertung

Die umfassende Retrospektive der Entwicklung des Forschungsgebiets der „Tangible User Interfaces“, die von (Ishii, 2008) vorgenommen wird, ermöglicht einen breiten Blick auf das aktuelle Feld und identifiziert außerdem nach wie vor offene Forschungs-Punkte. Für die Beurteilung eines TUI ist die Arbeit so wie die Ansätze, auf denen sie aufbaut und die sie integriert, nur bedingt geeignet. Dies liegt in der abstrakt-konzeptionell bleibenden Beschreibung der einzelnen Aspekte begründet, die nur bedingt eine argumentierbare Einordnung ermöglichen. Detailaspekte eines Systems bleiben unberücksichtigt, Ziel des Artikels ist es nicht, eine Analyse- oder Spezifikationsframework einzuführen (tatsächlich wird dies als eine der offenen Forschungspunkte genannt).

9. Konzeptionelle Einordnung

Für das Werkzeug können aus oben genannten Gründen aus der Betrachtung im Lichte dieses Ansatzes keinerlei Potentiale für Verbesserung abgeleitet werden. Die mögliche globale Einordnung des Gesamtsystems wurde bereits in anderen Abschnitten auf Basis der diesem Ansatz zugrunde liegenden Arbeiten vorgenommen und bringt an dieser Stelle keine neuen Erkenntnisse.

9.13. Zusammenfassung

In diesem Kapitel wurde das hier vorgestellte Werkzeug in die in Abschnitt 5.2 beschriebenen Ansätze zur konzeptionellen Betrachtung eines Tangible User Interface eingeordnet. Damit wurden zwei Zielsetzungen verfolgt. Einerseits sollten die vorgestellten Ansätze hinsichtlich ihrer grundsätzlichen Eignung zur Ausdrucksstärke bei der Beschreibung eines Tangible User Interfaces überprüft werden. Andererseits sollte aus der strukturierten konzeptionellen Betrachtung des Werkzeugs mögliche Inkonsistenzen in Design und Implementierung identifiziert werden und ggf. daraus Maßnahmen zur Verbesserung des Werkzeugs abgeleitet werden. Diese beiden Aspekte werden in den folgenden beiden Abschnitten getrennt betrachtet.

9.13.1. Eignung der konzeptionellen Ansätze zur Beschreibung

9.13.2. Verbesserungspotential für das Werkzeug

10. Überblick über die empirische Untersuchung

In diesem Kapitel wird ein Überblick über die in dieser Arbeit durchgeführte empirische Untersuchung gegeben. Dabei wird auf die einzelnen zu untersuchenden Aspekte, deren theoretische Grundlagen und die Durchführung der Untersuchung gegeben.

Die im Rahmen der empirischen Evaluierung zu untersuchenden Aspekte sind Gegenstand des ersten Abschnitts. Neben einer wiederholenden grundlegenden Betrachtung werden hier die jeweiligen Untersuchungsfragen festgelegt. Eine nähere Betrachtung der einzelnen Aspekte, die Festlegung der Methodik und deren Operationalisierung im Rahmen des konkreten Untersuchungsdesigns erfolgt im Rahmen der übrigen Kapitel in diesem Teil der Arbeit.

Im zweiten Abschnitt wird ein Überblick über das globale Untersuchungsdesign gegeben. Auf Basis der zu evaluierenden Aspekte werden die konkret durchgeführten Teile der Evaluation (im Folgenden: "Evaluierungsblöcke") beschrieben und den Aspekten zugeordnet. Diese Evaluierungsblöcke werden überblicksweise hinsichtlich der intendierten Ziele, der Aufgabenstellung und der jeweiligen Anzahl der Teilnehmer beschrieben. Die Beschreibung bildet die Grundlage für die Beschreibung der Evaluierung der zu prüfenden Aspekte in den folgenden Kapiteln.

10.1. Untersuchungsaspekte

Ziel dieser Arbeit ist die Unterstützung von expliziter Articulation Work. Eine Möglichkeit, explizite Articulation Work zu unterstützen, ist die Externalisierung und Abstimmung der mentalen Modelle über den betreffenden Arbeitsvorgang, die den Handlungen der beteiligten Personen zugrunde liegen (siehe Kapitel XY). Die Externalisierung mentaler Modelle ist mittels unterschiedlicher Methoden möglich, wobei sich Ansätze, die auf der Abbildung mentaler Modelle in diagrammatischen Strukturen basieren, als gut geeignet erwiesen haben (siehe Kapitel XY). Zwei derartige Methoden sind Concept Mapping und Strukturlegetechniken, die beide Vor- und Nachteil hinsichtlich des Einsatzes in kollaborativen Szenarien zeigen (siehe

Kapitel XY). In dieser Arbeit wird deshalb versucht, die Vorteile der beiden Ansätze methodisch zu vereinigen und zur Vermeidung der Nachteile durch ein Tabletop Interface zu unterstützen (siehe Kapitel XY).

Anhand dieser Argumentationskette zeigt sich, dass zwischen der Zielformulierung und dem konkreten Werkzeug zur Zielerreichung einige argumentative Schritte liegen, die vorerst lediglich (aus der Literatur begründete) Annahmen darstellen. Im Zuge der Evaluation der Ergebnisse dieser Arbeit müssen nun diese Schritte einzeln betrachtet werden und hinsichtlich der jeweiligen Zielerreichung überprüft werden. Untersuchungsgegenstand ist dabei jeweils das erstellte Werkzeug, die betrachteten Aspekte unterscheiden sich je nach Argumentationsschritt. Die Untersuchungsfragen, die die Argumentationsschritte abdecken sind:

- Unterstützen Werkzeug und Methode Articulation Work? (Aspekt: Werkzeug)
- Erlauben Werkzeug und Methode die Abbildung semantisch offener diagrammatischer Modelle? (Aspekt: Modell)
- Sind das Werkzeug und dessen Komponenten verständlich und wie intendiert einsetzbar? (Aspekt: Articulation Work)

Diese Fragen decken die Aspekte der oben beschriebene Argumentationskette ab, die Detaillierung der Fragestellungen ist in den folgenden Abschnitten beschrieben. Die Beschreibung der zu prüfenden Hypothesen sowie die Operationalisierung der Untersuchungsfragen erfolgt in den Kapitel XY bis XY.

10.1.1. Evaluierung des Werkzeugs

Die Evaluierung des Werkzeugs an sich beschäftigt sich mit der Beantwortung der dritten Untersuchungsfrage. Diese zielt auf die Verständlichkeit des Werkzeugs im weiteren Sinn ab. Unter Verständlichkeit im weiteren Sinn ist hier zu verstehen, dass einerseits geprüft werden muss, ob die Bedeutung und grundlegende Verwendung der Komponenten des Werkzeugs von Benutzern erfasst und verstanden werden und ob andererseits die Interaktionsabläufe, die zur Auslösung bzw. Abwicklung einer Funktion des Werkzeugs führen, für Benutzer verständlich und nachvollziehbar sind. Mögliche Metriken sind hier die Faktoren zur Evaluierung interaktiver Systeme nach Shneiderman (REF).

Neben der quantitativen Bewertung anhand dieser Metriken ist bei der Untersuchung dieses Aspektes vor allem auch das qualitative Feedback der Benutzer notwendig, um Ansatzpunkte zur Verbesserung der Verwendbarkeit des Werkzeugs zu erhalten. Diese Anregungen können im Sinne eines iterativen Designprozesses umgesetzt und deren Auswirkungen erneut einer Evaluierung unterzogen werden. Neben der Erhebung dieser zusätzlich funktionalen Anforderungen für einen iterati-

ven Designprozess sind in diesem Zusammenhang auch Hinweise hinsichtlich nicht-funktionaler Aspekte des Systems zu berücksichtigen, die der Verwendbarkeit negativ beeinflussen bzw. auch unkritisch sein können.

Die Verwendbarkeit des Werkzeugs kann nicht entkoppelt von der Anwendungsdomäne betrachtet werden, muss also im Kontext der Aufgabe, für die es eingesetzt wird, gesehen werden. Das Werkzeug ist zwar grundsätzlich für die Repräsentation beliebiger diagrammatischer Modelle ausgelegt, eignet sich aufgrund der unterschiedlichen Anforderungen jedoch nicht gleich gut für alle möglichen Anwendungsfälle (so sind z.B. ausschließlich Verbindungen mit zwei Endpunkten erstellbar, Verbindung mit mehr Endpunkten werden nicht unterstützt). Die Prüfung der Verwendbarkeit des Werkzeugs kann hier fokussiert auf die in dieser Arbeit verfolgten Anwendungsfälle durchgeführt werden, die im Bereich der konzeptionellen Netze (im Wesentlichen Varianten von Concept Maps) und im Bereich der Abbildung von Arbeitsvorgängen (im Wesentlichen kausale Zusammenhänge mit Kontextinformation) zu finden sind. Die Unterstützung anderer Anwendungsfälle ist möglich und unter Umständen erstrebenswert, stellt jedoch kein Beurteilungskriterium dar.

10.1.2. Evaluierung der Modellrepräsentationen

Der zweite zu evaluierende Aspekt sind die mit dem Werkzeug erstellten Modelle, die als Mittel zur Durchführung expliziter Articulation Work dienen. Eine wesentliche Eigenschaft, die Modelle dabei aufweisen müssen, ist die Adäquatheit der Modellierungssprache hinsichtlich der durch die Benutzer zu repräsentierenden Information. Diese Eigenschaft wird in der vorliegenden Arbeit durch den in Kapitel XY beschriebenen Ansatz der semantischen Offenheit abgedeckt, der jedoch vor allem hinsichtlich der intersubjektiven Verständlichkeit der Modelle und deren Eindeutigkeit nicht nur Vorteile bringt. Grundlegende ist in dieser Phase zu evaluieren, ob die erstellten Modelle den im Rahmen des Einsatzes zur Unterstützung von Articulation Work intendierten Zweck erfüllen. Dabei sind sowohl das Modell als auch das (hier von den Benutzern festgelegte) Metamodell zu betrachten. Anhaltspunkte zur Identifikation der zu evaluierenden Objekte sowie zum Vorgehen bieten hier der Ansatz der "Interactive Process Models" (Jørgensen, 2004) und die "Grundsätze der ordnungsgemäßen Modellierung" (Becker et al., 2000) sowie von diesen Arbeiten abgeleitete Ansätze.

Die eben beschriebenen Ansatzpunkte erlauben eine Evaluierung der erstellten Modelle hinsichtlich der Abbildbarkeit der Kernaspekte von "Articulation Work" im engeren Sinne (Strauss' "salient dimensions": "*who, where, when, what and how*" (Fjuk et al., 1997)), decken also im Wesentlichen eine an organisationalen Abläufen orientierten Sicht auf Modelle ab. Im Sinne der Offenheit der Abbildung müssen

aber auch Modelle berücksichtigt werden, die nicht diese "salient dimensions" zur Grundlage haben, also "Concept Maps" (Novak und Cañas, 2006) im allgemeinen Sinn sind und damit die Abbildung mentaler Modelle nicht nur über unmittelbare Arbeitsaspekte sondern über beliebige Sachverhalte erlauben (Ifenthaler, 2006). Dabei sind Metriken notwendig, die die erstellten Modelle selbst betrachten und deren Eigenschaften und Verwendung beim Concept Mapping bzw. im Rahmen von Strukturlegetechniken berücksichtigen.

Wie bereits im letzten Abschnitt angeführt, ist auch bei diesem Aspekt der Evaluierung der in dieser Arbeit verfolgte Anwendungszweck des Werkzeugs (bzw. hier: der Modelle) zu berücksichtigen. Dies ist insofern ein einschränkender Faktor, als dass hier Modelle lediglich im Kontext der Externalisierung mentaler Modelle und zur Unterstützung von Articulation Work berücksichtigt werden. Das Werkzeug selbst erlaubt auch die Erstellung von Modellen zu anderen Anwendungszwecken, die jedoch hier nicht weiter berücksichtigt werden.

10.1.3. Evaluierung der Articulation Work

Letztendlich muss auch die durchgeführte Articulation Work selbst beurteilt werden. In der Literatur zum Thema "Articulation Work" werden zumeist lediglich das Phänomen "Articulation Work" und dessen konkrete Ausprägungen beschrieben (siehe Kapitel XY), Ansätze zur Bewertung des Erfolgs von Articulation Work sind jedoch selten zu finden. Aus der Verschränkung zwischen Articulation Work und Production Work, also jenem Anteil der Arbeit, der unmittelbar der Zielerreichung dient, die von mehreren Autoren, unter anderem (Fujimura, 1987) und (Strauss, 1993), erwähnt wird, lassen sich jedoch Ansatzpunkte ableiten.

Articulation Work tritt immer dann auf, wenn eine Zielerreichung in der Production Work aufgrund von Unklarheiten oder Problemen zwischen den beteiligten Individuen nicht möglich ist. Ein erfolgreicher Abschuss der Production Work bei am Beginn oder während der Arbeit bestehenden Unklarheiten weißt also unter Umständen auf erfolgreich durchgeführte Articulation Work hin. Articulation Work manifestiert sich im Arbeitsprozess auf unterschiedliche Arten, so dass bei der Evaluierung hinsichtlich der Auswirkungen des Werkzeugs diese von den übrigen Einflussfaktoren (also auf anderen Wegen durchgeführte Articulation Work) getrennt werden muss. Dazu ist eine Betrachtung des gesamten Arbeitsablaufs unter Berücksichtigung von Production und Articulation Work notwendig. Metriken, die bei der Bewertung des Erfolgs von Articulation Work zu berücksichtigen sind, sind also einerseits im Ergebnis des Arbeitsprozesses, andererseits auch im Arbeitsprozess selbst zu finden.

Ein zweiter Ansatzpunkt zur Bewertung des Erfolgs von Articulation Work liegt in den Aussagen von Strauss (1993) hinsichtlich der wahrgenommenen "Problematik" einer Arbeitssituation, die Articulation Work notwendig macht. Diese Wahrnehmung ist individueller Natur, d.h. Articulation Work ist dann notwendig, wenn zumindest einer am Arbeitsablauf beteiligten Person Aspekte der Arbeit unklar sind oder problematisch erscheinen. Im Gegenzug ist keine Articulation Work notwendig bzw. diese abgeschlossen, wenn alle beteiligten Personen die Situation als unproblematisch empfinden bzw. mit den im Rahmen der (expliziten) Articulation Work erzielten Ergebnissen zufrieden sind. Hier liegt der Ansatzpunkt für eine Evaluierung des Erfolgs der durchgeführten Articulation Work, der auf diese auf Basis der individuellen Wahrnehmungen der beteiligten Personen beurteilt.

10.2. Globales Untersuchungsdesign

Die oben beschriebenen Aspekte müssen nun im Rahmen einer empirischen Untersuchung getestet werden. Während die detaillierten Untersuchungsdesigns in den folgenden Kapiteln, die sich jeweils einem der drei zu evaluierenden Aspekte widmen, beschrieben werden, wird an dieser Stelle ein Überblick über das globale Untersuchungsdesign und die im Rahmen der Evaluierung durchgeführten Anwendungen des Werkzeugs gegeben.

Im ursprünglichen globalen Untersuchungsdesign war vorgesehen, jedem der zu untersuchenden Aspekte einen Block an Anwendungen des Werkzeugs mit einer auf den jeweiligen Aspekt abgestimmten Aufgabenstellung zuzuordnen. Nach Durchführung der ersten beiden Blöcke wurde offensichtlich, dass sich während der Evaluierung zusätzlich Hypothesen zu einem Aspekt bildeten, die – um sie in die Evaluierung einfließen zu lassen – in einem späteren Block getestet werden mussten. Außerdem wurde offensichtlich, dass vor allem zur Evaluierung des Werkzeugs in allen Blöcken Verbesserungspotential identifiziert werden konnte bzw. Anregungen der Anwender rückgemeldet wurden, die zum Teil im Rahmen des iterativen Entwicklungsprozesses in das Werkzeug einflossen und entsprechend in einem späteren Block erneut getestet werden musste.

Letztendlich wurden die Blöcke, sofern die jeweilige Aufgabenstellung geeignet war, für die Evaluierung mehrerer bzw. aller Aspekte herangezogen. Bei der nun folgenden Beschreibung der Anwendungs-Blöcke wird deshalb jeweils angegeben und begründet, inwieweit diese in die Evaluierung welcher Blöcke einfließen. Ein Überblick über das globale Untersuchungsdesign mit einer erneuten, überblicksweisen Zuordnung zwischen den zu evaluierenden Aspekten und den Anwendungsblöcken wird in Abschnitt 10.4 gegeben.

10.2.1. Block I: Technische Evaluierung

Die Intention von Block I war die grundlegende Verständlichkeit und Verwendbarkeit des Werkzeugs zu klären. Fokus dieses Blocks an Anwendungen des Werkzeugs war also die Untersuchung der Eigenschaften des Werkzeugs selbst. Zusätzlich sollte explorativ Hypothesen für die übrigen zu evaluierenden Aspekte gebildet werden.

Kontext

Die Untersuchung wurde im Rahmen einer Diplomarbeit durchgeführt (REF Bohninger), wobei die Untersuchungen in keinen einheitlichen realen Arbeitskontext eingebettet waren. Allerdings war die Aufgabenstellung so formuliert, dass die erstellten Modelle aus den Arbeitskontexten der jeweiligen Teilnehmer stammten.

Aufgabenstellung und Ablauf

Den modellierenden Teilnehmern wurde mitgeteilt, dass sie einen Aspekt aus ihrem täglichen Arbeits- oder Privatleben abbilden sollten, der regelmäßig auftritt oder bereits mehrmals für Probleme sorgte. Die bewusste Offenheit der Aufgabenstellung sollte dabei bewirken, dass sich die Teilnehmer nicht zu sehr auf den abzubildenden Sachverhalt, sondern eher auf den Abbildungsprozess selbst fokussierten.

Nur die Hälfte der Teilnehmer erstellte Modelle. Die zweite Hälfte wurde zur Überprüfung der Verständlichkeit der Modelle sowie der Verwendung des Werkzeugs zur kollaborativen Modellierung herangezogen. Dazu wurde nach Abschluss einer Modellbildung jeweils ein nicht modellierender Teilnehmer an die Modellierungsfläche gebeten und gebeten, die Abbildung zu interpretieren. Die Beurteilung der Adäquatheit dieser Interpretation erfolgte durch den modellierenden Teilnehmer.

In einer dritten Phase wurden beide Teilnehmer aufgefordert, dass Modell gemeinsam zu reflektieren und gegebenenfalls zu verändern, um es den Ergebnissen der Reflexion anzupassen. In dieser Phase war das vorrangige Ziel, die Verwendung des Werkzeugs bei der Veränderung von Modellen und dessen kollaborativer Anwendung zu testen.

Anwendungen und Teilnehmer

Insgesamt wurden neun Anwendungen des Werkzeug wie oben beschrieben durchgeführt. Zusätzlich wurde das Untersuchungsdesign im Rahmen von drei Anwendungen getestet (Pretest), woraus hinsichtlich der technischen Eigenschaften des

Werkzeugs ebenfalls bereits Erkenntnisse gewonnen werden konnten. Insgesamt nahmen also 24 Personen an diesem Block von Anwendungen teil, 6 davon in der Pretest-Phase.

Die Teilnehmer stammten aus unterschiedlichen beruflichen Hintergründen (DETAILS) und unterschieden sich auch in Art der höchsten abgeschlossenen Ausbildung (DETAILS). Die Altersspanne lag zwischen XY und XY Jahren, XY Teilnehmer waren weiblich, XY männlich.

Die Modellierungsphasen dauerten im Schnitt XY Minuten, die kürzeste Modellbildung dauerte XY Minuten, die längste XY Minuten. Die Interpretations- und Reflexionsphasen (nicht separat aufschlüsselbar, da zum Großteil ineinander übergehend) dauerten im Schnitt XY Minuten.

Verwendung der Ergebnisse

Die Ergebnisse dieses Blocks flossen in die Evaluierung des Werkzeugs und in die Hypothesenbildung hinsichtlich der erstellten Modelle ein. Für die Evaluierung der Modelle konnten erste Erkenntnisse hinsichtlich der Verständlichkeit der mittels offener Semantik gewonnen werden. Keine Ergebnisse brachte dieser Block für die Evaluierung der durchgeführten Articulation Work.

10.2.2. Block 2: Aushandlung von Zusammenarbeit I

In Block 2 lag der Fokus der Evaluation erstmals auf der Unterstützung von Articulation Work. In diesem Rahmen wurden auch die Verwendbarkeit des Werkzeugs im praktischen Anwendungskontext und die Eigenschaften der erstellten Modelle sowie deren Rolle im Prozess der expliziten Articulation Work untersucht.

Kontext

Block 2 wurde im Rahmen eines Seminars aus Wirtschaftsinformatik mit Studierenden derselben Studienrichtung durchgeführt. Die im Seminar zu erstellenden wissenschaftlichen Arbeiten wurden von den Studierenden in Gruppen zu 2-3 Personen ausgearbeitet. Die Gruppen wurden so gebildet, dass sich die Teilnehmer nicht persönlich kannten oder zumindest nicht bereits in anderen Kontexten zusammen-gearbeitet hatten. Ziel dieser Maßnahme war die Vermeidung der Verfälschung der Untersuchungsergebnisse durch bereits eingespielte Gruppen (Erfahrungen in Seminaren der Vorjahre zeigen tendentiell schlechtere Ergebnisse bei der Zusammenarbeit von einander nicht persönlich bekannten bzw. nicht eingespielten Teilnehmern).

Im Rahmen des Seminars wurden sechs Forschungsgebiete ausgewählt, die in Zusammenhang mit der Erstellung und Verwendung sozio-technischer Systeme stehen (konkret: Organisationales Lernen, eLearning, CSCW, Mentale Modelle, Articulation Work und semantische Contentanreicherung). Den Gruppen wurden jeweils zufällig zwei dieser Themen zugewiesen, die Aufgabe für die wissenschaftliche Arbeit war das Finden und Beschreiben einer möglichen Verknüpfung oder eines möglichen Zusammenhangs zwischen diesen Themen. Dieser Zusammenhang sollte im Zentrum der Seminararbeit stehen und aus beiden Grundlagen-Themen argumentiert sein. Ziel dieser Maßnahme war es, die Seminararbeit so offen wie möglich zu gestalten und eine Themenfindungs bzw. -konkretisierungsprozess in den Ablauf zu integrieren. Außerdem wurde so ein Setting geschaffen, in dem eine strikte Arbeitsteilung der Gruppenteilnehmer ohne weitere Zusammenarbeit sich tendenziell stark auf das Ergebnis auswirkt und sich konkret der fehlenden oder schwachen Verknüpfung der Grundlagen-Themen zeigt.

Aufgabenstellung und Ablauf

Das Werkzeug wurde im Rahmen des Seminars für jede Gruppe zweimal eingesetzt. Die erste Anwendung fand zu Beginn des Seminars nach der Themenzuteilung statt. Die Aufgabe war die Aushandlung der Modalitäten der Zusammenarbeit mit der Zielsetzung, das an der resultierenden wissenschaftlichen Arbeit die Kooptorenchaft nicht mehr zu erkennen sein sollte (etwa durch plötzlich wechselnde Schreibstile oder Brüche in der Argumentationskette). Den Teilnehmern wurde das Werkzeug und dessen Funktionen vorgestellt und ohne weitere Vorgaben zur Verfügung gestellt (insbesondere wurden weder Vorgaben hinsichtlich der Topologie des zu erstellenden Modells oder der Bedeutung der Modellierungselemente gemacht).

In der zweiten Anwendung wurde der Zusammenarbeitsprozess reflektiert und gegebenenfalls eine Adaption vereinbart. Die zweite Anwendung fand in der Mitte des Semesters nach Abschluss der Literaturrecherche und der Grobkonzeption, aber vor der Erstellung der eigentlichen wissenschaftlichen Arbeit statt. Konkrete Zielsetzung für die Teilnehmer war hier, auf Basis der bisherigen Erfahrungen die weitere Zusammenarbeit zu vereinbaren. Das Werkzeug wurde ohne neuerliche Vorstellung und ohne Vorgaben zur Verfügung gestellt.

Anwendungen und Teilnehmer

Insgesamt nahmen an diesem Block 19 Personen in 9 Gruppen zu 2 bzw. einmalig 3 Personen teil. Jede der Gruppen setzte das Werkzeug zweimal ein, wodurch insgesamt 18 Anwendungen die Grundlage für die Auswertung der Ergebnisse bilden.

Die Teilnehmer waren allesamt Studierende der Wirtschaftsinformatik im zweiten Studienabschnitt, 18 Personen waren männlich, eine weiblich. Vier Personen hatten insofern Erfahrung mit wissenschaftlichen Arbeiten bzw. den konkreten Anforderungen in der betreffenden Lehrveranstaltungen, als dass sie bereits zuvor eine Lehrveranstaltung gleichen Typs besucht hatten.

In der ersten Runde dauerten die Anwendungen durchschnittlich XY Minuten, in der zweiten Runde lediglich XY Minuten.

Verwendung der Ergebnisse

Die in diesem Block erhobenen Daten fließen in die Auswertung alle drei zu evaluierenden Aspekte ein. Zur Auswertung hinsichtlich des Erfolgs von Articulation Work liegen neben den Aufnahmen der Modellierungsvorgänge und den erstellten Modellen selbst auch Prozessreflexionen der Teilnehmer über den Erstellungsprozess der Seminararbeiten sowie die Seminararbeit ansich vor. Die Auswirkungen von Articulation Work können also am Ergebnis (im Vergleich zu Ergebnissen auf Lehrveranstaltungen mit identischem Konzept) und am subjektiv wahrgenommenen Verlauf des Erstellungsprozesses der Arbeit bewertet werden.

Hinsichtlich des Auswertung des Modell-Aspektes wird durch diesen Block die Betrachtung von Modellen ermöglicht, die im Kontext der Arbeitsabstimmung erstellt wurden, also im Wesentlichen der Definition von Vorgehen und Schnittstellen dienen. Untersucht werden hier Aufbau und Inhalt der Modelle, wobei besonderes Augenmerk auf der Prozess und Ergebnis der Bedeutungszuweisung zu den Modellelementen liegt.

Im Rahmen der Werkzeug-Evaluation bringt dieser Block die ersten Hinweise auf die Anforderungen an das Werkzeug bei der Verwendung desselben im Rahmen einer realen Aufgabenstellung. Außerdem wurde in diesem Block erstmals ein durchgängig kollaboratives Szenario eingesetzt, bei dem immer mindestens zwei Personen gleichzeitig das Werkzeug verwenden.

10.2.3. Block 3: Concept Mapping I

Der Fokus von Block 3 lag auf der Erstellung von semantisch vernetzten Strukturen im Allgemeinen, wobei das Konzept der Concept Maps als ein etabliertes Werkzeug zur Externalisierung mentaler Modelle eingesetzt wurde. Inhaltlich fokussierte dieser Block nicht auf die Unterstützung von Articulation Work im engeren Sinne, wohl aber auf die Externalisierung und Abstimmung mentaler Modelle, was wie in Kapitel XY beschrieben ein Mittel zur Unterstützung expliziter Articulation Work ist. Im Zentrum der Aufmerksamkeit steht in diesem Block also die Evaluierung der

erstellten Modelle und der Nutzen des Werkzeugs zur Aushandlung einer einheitlichen auf einen gegebenen Sachverhalt.

Kontext

Der dritte Block wurde im Rahmen einer Lehrveranstaltung zur Schulung von Methoden der Prozess- und Kommunikationsmodellierung durchgeführt. Diese Lehrveranstaltung ist Teil der im Curriculum definierten Basiskompetenz Wirtschaftsinformatik und wird von Studierenden im zweiten bis dritten Studiensemester besucht.

Im Rahmen der Lehrveranstaltung wurden drei unterschiedliche Prozessmodellierungssprachen (SeeMe (Herrmann et al., 2004a), Subjekt-orientierte Modellierung mittels JPass REF und EPK's aus dem ARIS-Konzept (Scheer und Nuettgens, 2000)) eingeführt und praktisch an einem durchgängigen Beispiel angewandt. Diese Sprachen unterscheiden sich sowohl im Anwendungsgebiet, in den abgebildeten Aspekten des realen Prozesses sowie in der Darstellungsform des Modells. Ziel der letzten Teilaufgabe, die unter Einsatz des hier vorgestellten Werkzeugs durchgeführt wurde, war bei den Studierenden ein Verständnis für die Unterschiede und Gemeinsamkeiten zwischen diesen Sprachen zu erzeugen und sie in die Lage zu versetzen, für einen gegebenen Anwendungsfall eine adäquate Sprache auszuwählen.

Aufgabenstellung und Ablauf

Die Aufgabe zur Erstellung der Concept Map umfasste zwei Teile, wobei im zweiten Teil das Tabletop Interface eingesetzt wurde. Die Aufgabenstellung lautete in beiden Teilen, eine Concept Map zu erstellen, die die wesentlich erscheinenden Eigenschaften der vorgestellten Sprachen sowie deren Gemeinsamkeiten und Unterschiede darstellt. In der ersten Phase war diese Aufgabe von den Studierenden individuell zu lösen, wobei die Concept Map auf Papier oder mit Hilfe des Werkzeugs CMapTools² (Cañas et al., 2004) am Rechner erstellt werden konnte.

In der zweiten Phase wurden Gruppen zu je drei Teilnehmern gebildet, die nun ihre individuellen Sichten konsolidieren und jeweils eine gemeinsame Concept Map zur gleichen Aufgabenstellung unter Einsatz des hier vorgestellten Werkzeugs erstellen sollten. Die Gruppen wurden zufällig zusammengesetzt, den Teilnehmern war während der individuellen Phase die Zuteilung nicht bekannt, so dass eine Abstimmung vor Anwendung des Werkzeugs weitgehend ausgeschlossen werden kann.

¹Ereignisgesteuerte Prozesskette

²<http://cmap.ihmc.us>

Anwendungen und Teilnehmer

An den Anwendungen, die in diesem Block durchgeführt wurden, nahmen insgesamt 54 Personen teil, die in 18 Gruppen einmalig mit dem Werkzeug arbeiteten. Alle Teilnehmer waren Studierende der Wirtschaftsinformatik im ersten Studienabschnitt (1-4 Semester), 8 waren weiblich, 46 männlich. Keinem der Teilnehmer war der Ansatz des Concept Mapping vor Beginn der betreffenden Aufgabe bekannt, Erfahrungen mit Prozessmodellierungssprachen (also dem Gegenstand der Concept Map) sammelten alle Teilnehmer erstmals im Rahmen der Lehrveranstaltung, in der dieser Evaluierungs-Block durchgeführt wurde.

Den Teilnehmern wurde das Werkzeug vor Beginn der Anwendung demonstriert und in sämtlichen Anwendungsaspekten erklärt. Die Anwendungen selbst dauerten durchschnittlich XY Minuten, wobei die kürzeste Anwendung XY Minuten, die längste XY Minuten dauerte.

Verwendung der Ergebnisse

Die Daten, die aus diesem Block gewonnen werden konnten, gehen in die Evaluierung des Modell-Aspekts ein. Hier können einerseits wiederum die erstellten Modelle hinsichtlich Struktur, Inhalt und semantischen Zuweisungen untersucht werden. Der Modellierungsgegenstand ist in diesem Fall jedoch anders gelagert als im vorhergehenden Fall, anstelle eines Arbeitsabstimmung ist hier ein Vergleich von Konzepten durchzuführen. Andererseits können hier die Abstimmungsprozesse der individuellen mentalen Modelle insofern betrachtet werden, als dass für jede Gruppe neben dem kollaborativ erstellten Ergebnis auch noch die individuellen Concept Maps vorliegen und ausgewertet werden können.

Wie bereits in den zuvor beschriebenen Blöcken können auch hier wieder Erkenntnisse hinsichtlich der Verwendung des Werkzeugs gewonnen werden. Aufgrund der der Aufgabe innenwohnenden Wichtigkeit der Verbindungen zwischen Konzepten wird vor allem deren Verwendung bzw. der Vorgang deren Erstellung zu betrachten sein.

Der Aspekt Articulation Work bleibt in diesem Block im engeren Sinne außen vor, als dass kein Arbeitskontext vorliegt, keine aufzulösende Problematik vorliegt und keine Zusammenarbeit auszuhandeln ist. Insofern wird dieser Aspekt in diesem Block nicht explizit behandelt. Aufgrund der Durchführung sämtlicher Schritte, die zur Unterstützung expliziter Articulation Work notwendig sind (Externalisierung, Abstimmung) können aber die einzelnen Anwendungen zur Hypothesenbildung für den Evaluierungs-Aspekt Articulation Work herangezogen werden.

10.2.4. Block 4: Aushandlung von Zusammenarbeit 2

Block 4 deckt die erste Anwendung des Werkzeugs im realen Unternehmenskontext ab. Im Rahmen einer Diplomarbeit REF wurde das Werkzeug zur Offenlegung unmittelbar relevanter bzw. urgenter Fragestellungen eingesetzt, die im Rahmen eines Workshops zu den Abläufen in und zur Struktur der IT-Abteilung einer Unternehmensgruppe aus dem Bildungsbereich auftraten. Fokus dieses Blocks war die Untersuchung der Einsetzbarkeit des Werkzeugs im praktischen Kontext und dessen tatsächlicher Unterstützungsleistung für Articulation Work. Dazu wurde neben der Begleitung der eigentlichen Modellierungssession in zeitlichem Abstand auch eine Erhebung der wahrgenommenen Wirkungen auf die Arbeitspraxis durchgeführt.

Kontext

Das Werkzeug wird im Kontext einer österreichweit tätigen Unternehmensgruppe im Aus- und Weiterbildungsbereich eingesetzt. Konkret kam das Werkzeug bei einem Workshop zum Einsatz, der von der Abteilung für technisches Produkt- und Service-Management in der konzernweiten IT-Abteilung abgehalten wurde. Die Abteilung hat rund 30 Mitarbeiter, die sich in insgesamt 5 Unterabteilungen gliedern. Zusätzlich ist ein Mitarbeiter abteilungsweit für die Qualitätssicherung der Arbeitsabläufe verantwortlich. Dieser leitete die Workshops, bei denen das Werkzeug zum Einsatz kam und wählte in Abstimmung mit den jeweils betroffenen Kollegen die Themenauswahl durch.

Aufgabenstellung und Ablauf

In unterschiedlichen Konstellationen mit Gruppengrößen von 2 bis 6 Personen wurden an zwei Workshop-Terminen Themen aus dem täglichen Arbeitskontext behandelt. Dabei wurden zum Einen Unterabteilungs-interne oder -übergreifende Arbeitsabläufe abgebildet und ausgehandelt, die als potentiell problematisch oder neu einzurichten wahrgenommen wurden. Zum Anderen wurde die wahrgenommene Struktur einer Unterabteilung selbst und deren Außenbeziehungen abgebildet, reflektiert und zwischen den Mitgliedern derselben abgestimmt.

Bei Aufgaben der ersten Kategorie begann die Bearbeitung jeweils mit der kooperativen Repräsentation des Ist-Standes und damit einem Abgleich der individuellen Sichten auf den aktuellen Arbeitsablauf. In weiterer Folge wurde anhand des Modells mögliches Optimierungspotential diskutiert und das Modell ggf. dementsprechend adaptiert.

Bei der Darstellung der Struktur einer Unterabteilung wurden im ersten Schritt die relevanten organisationalen Einheiten und Rollen gesammelt und auf der Ober-

fläche platziert. In weiterer Folge war die Aufgabe die Zusammenhänge innerhalb der Unterabteilung und deren Beziehungen nach außen durch räumliche Anordnung der definierten Einheiten sowie deren Kommunikationskanäle explizit durch Assoziationen darzustellen. Ziel war eine Repräsentation des Ist-Zustands der Abteilung, die soweit abgestimmt wurde, dass alle Teilnehmer ihre individuelle Sicht auf das Modell abbilden konnten.

Anwendungen und Teilnehmer

Am ersten Workshop-Tag nahmen insgesamt 6 Teilnehmer an 5 Modellierungsdurchgängen in Gruppen von 2 bis 5 Personen teil. Beim zweiten Workshop nahmen insgesamt 8 Teilnehmer an ebenfalls 5 Modellierungsdurchgängen teil. Insgesamt beschäftigten sich 8 Aufgaben mit konkreten Arbeitsabläufen, 2 Aufgaben widmeten sich der Struktur von Unterabteilungen. Die Gruppengröße variierte zwischen 3 und 6 Personen. 10 Teilnehmer nahmen an mehr als einem Modellierungsdurchgang teil, eine Person war an beiden Workshop-Tagen beteiligt.

Durch die Einbindung aller Unterabteilungen kamen Teilnehmer mit unterschiedlichem fachlichen Hintergrund zu Einsatz. Etwa die Hälfte der Teilnehmer war der Gruppe der Techniker oder Softwareentwickler zuzuordnen. Die andere Hälfte setzte sich aus Mitarbeiter im Support, Verkauf, Einkauf sowie der internen Verrechnung zusammen. Eine Teilnehmerin war weiblich, alle anderen Teilnehmer waren männlich.

Allen Teilnehmern wurde einmalig das Werkzeug und dessen Bedienung vorgestellt. Die Modellierungsdurchgänge dauerten zwischen 25 Minuten und etwa 1,5 Stunden. Sämtliche Teilnehmer wurden nach ihrer letzten Teilnahme an einem Durchgang mittels einem Fragebogen (siehe Anhang XY) sowohl nach der Nützlichkeit des Werkzeugs als auch nach dem wahrgenommenen Nutzen des inhaltlichen Ergebnisses befragt. Um die mittelfristigen Auswirkungen der durchgeföhrten Modellierungsdurchgänge beurteilen zu können, wurde acht Wochen nach dem zweiten Workshop erneut eine Befragung durchgeführt, in der die wahrgenommene Auswirkungen thematisiert wurden (siehe Anhang XY).

Verwendung der Ergebnisse

Die Daten, die das Ergebnis dieses Blocks bilden, werden zur Evaluierung des Aspekts „Articulation Work“ eingesetzt. Betrachtet werden dabei die wahrgenommenen und beobachtbaren Veränderungen am Arbeitsprozess, der unter Einsatz des Werkzeugs reflektiert wurde.

Neben diesem Aspekt werden auch die erstellten Modelle, das in diesem Fall wieder aus der Domäne der Arbeitsabstimmung stammen, betrachtet und hinsichtlich ihrer Struktur und Semantik ausgewertet.

Der Werkzeug-Aspekt wird in diesem Teil der Untersuchung nicht gesondert betrachtet, Verbesserungs- und Erweiterungspotential wird nur bei Erwähnung oder offensichtlichen Bedienungsfehlern bzw. Verständnisschwierigkeiten explizit identifiziert.

10.2.5. Block 5: Concept Mapping 2

In Block 5 wird im Wesentlichen der Evaluierungs-Blocks 3 (siehe Abschnitt 10.2.3) inhaltlich erneut durchgeführt (die Modellierungsaufgabe ist identisch). Im Gegensatz zu Block 3, wo die grundlegende Eignung des Werkzeugs zum Concept Mapping im Mittelpunkt stand, wird in Block 5 eine vergleichende Studie durchgeführt, die die Eignung des Tabletop Interfaces zum kollaborativen Concept Mapping mit jener der rechner-basierten CMap Tools (Cañas et al., 2004) vergleicht.

Kontext

Die Anwendungssituation ist in diesem Block identisch mit dem in Abschnitt 10.2.3 beschriebenen Kontext (Lehrveranstaltung im Curriculum Wirtschaftsinformatik zur Schulung von Ansätzen in der Prozess- und Kommunikationsmodellierung).

Der Ablauf der Lehrveranstaltung unterschied sich nur insofern von jenem in Block 3, als dass für jede Modellierungssprache separat eine Reflexion in Gruppen zu zwei Studierenden durchgeführt wurde. In diesen Reflexion wurden die eigenen Anwendungen der jeweiligen Sprache mit einer Musterlösung gegenübergestellt und hinsichtlich ihrer Korrektheit und dem Vorgehen bei der Modellierung betrachtet.

Aufgabenstellung und Ablauf

Die Aufgabenstellung ist identisch mit jener in Block 3. Ziel ist es, die drei vorgestellten Prozessmodellierungssprachen hinsichtlich ihrer als wesentlich empfundenen Eigenschaften und deren Gemeinsamkeiten und Unterschiede zu betrachten und in einer Concept Map abzubilden. Das Vorgehen unterscheiden sich jedoch wegen der unterschiedlichen Zielsetzung der Untersuchung von jenem in Block 3.

Nach Abschluss der letzten Reflexionsphase (also nach drei Modellierungsphasen und drei Reflexionsphasen) wurde eine Gruppeneinteilung für die kollaborative Erstellung der Concept Map vorgenommen. Die Gruppen wurden aus jeweils zwei zufällig ausgewählten Studierenden gebildet. In der Untersuchung erhielt nun eine

Hälften der Gruppen den Auftrag, die Aufgabenstellung unter Verwendung des Tabletop Interfaces durchzuführen, die andere Hälften verwendete das rechner-basierte Werkzeug CMapTools (Cañas et al., 2004), um die Concept Map zu erstellen. Die Gruppen wurden zufällig einem Werkzeug zugeordnet und führten die Aufgabenstellung in beiden Fällen kollaborativ in einer kontrollierten Umgebung durch. Im Gegensatz zu Block 3 entfiel hier die explizit geforderte individuelle Vorbereitungsphase, um eine stärkere inhaltliche Auseinandersetzung mit den Inhalten während der Modellierung zu fördern.

Anwendungen und Teilnehmer

An der Untersuchung nahmen 49 Studierende in 23 Gruppen teil, wobei 11 Gruppen die Aufgabenstellung unter Verwendung des hier vorgestellten Werkzeugs und 12 Gruppen unter Verwendung der CMapTools durchführten. Die Teilnehmer waren allesamt Studierende der Wirtschaftsinformatik in der ersten Phase des Bakkelauratsstudiums (erstes bis drittes Semester), 40 Teilnehmer waren männlich, 9 weiblich. Keiner der Teilnehmer hatte Vorkenntnisse in der Prozessmodellierung oder im Concept Mapping.

Den Teilnehmern wurde das Werkzeug vor Beginn der Anwendung demonstriert und in sämtlichen Anwendungsaspekten erklärt. Die Anwendungen selbst dauerten durchschnittlich XY Minuten, wobei die kürzeste Anwendung XY Minuten, die längste XY Minuten dauerte.

Verwendung der Ergebnisse

Die in diesem Block erhobenen Daten fließen in vorrangig in den Modell-Aspekt der Evaluierung ein. Hier wird eine vergleichende Studie durchgeführt, die das Ziel hat, die Eignung der beiden verwendeten Ansätze für die Externalisierung von mentalen Modellen gegenüberzustellen. Grundlage dieser Beurteilung ist das erstellte Modell, außerdem wird der auch Modellierungsprozess in der Auswertung berücksichtigt.

Hinsichtlich des Werkzeug-Aspekts wird in diesem Block neben der Identifikation von Verbesserungspotential und Verständnisschwierigkeiten auch die Zufriedenheit mit dem Werkzeug bzw. dessen Akzeptanz bei den Benutzern explizit erhoben.

Der Aspekt Articulation Work wird hier wie schon in Block 3 und aus den dort angeführten Gründen (siehe Abschnitt 10.2.3) nicht weiter berücksichtigt.

10.3. Eingesetzte Werkzeuge und Verfahren

10.3.1. Werkzeuge

10.3.2. Korrelationstests

10.3.3. Signifikanztests

Shapiro-Wilk-Test

Der Shapiro-Wilk-Test (Shapiro und Wilk, 1965) testet eine Verteilung auf „Nicht-Normalität“ (d.h. die Nullhypothese ist, dass die Verteilung nicht normalverteilt ist). Für eine Wahrscheinlichkeit $p < 0.05$ kann daher davon ausgegangen werden, dass die geprüfte Verteilung nicht normalverteilt ist. Dieser Test eignet sich auch für kleine Stichproben (ab $n > 3$).

Er wird hier eingesetzt, um zu prüfen, ob der t-Test eingesetzt werden kann oder nicht (da dieser Normalverteilung der Parameter voraussetzt).

Wilcoxon-Test

für zwei abhängige oder unabhängige Stichproben

Kruskal-Wallis-Test

für drei oder mehr unabhängige Stichproben

t-Test

10.4. Zusammenfassung

In diesem Kapitel wurde das globale Untersuchungsdesign zur Evaluierung der hier vorgestellten Arbeit beschrieben. In den ersten Abschnitten wurden die zu evaluierenden Aspekte identifiziert und beschrieben. Im Rahmen dieser Beschreibungen wurden auch mögliche Ansatzpunkte für die konkrete Untersuchung angeführt, die die Basis für die detaillierte Konzeption der Evaluierung dieser Aspekte in den Kapiteln XY bis XY bildet.

Im folgenden Abschnitt wurden die einzelnen im Rahmen der Evaluierung durchgeführten Untersuchungen angeführt. Diese Untersuchungen fokussieren jeweils auf

einen der zu evaluierenden Aspekte. Ihnen liegt jeweils ein konkretes Szenario zu Grunde, das in einer Reihe von Anwendungen des Werkzeugs durch verschiedene Benutzer in Modelle umgesetzt wird. Je nach Fokus der Untersuchung werden vor- und nachgelagerte bzw. parallel ablaufende Aktivitäten in die Untersuchung mit einbezogen.

Die ursprüngliche Zuordnung zwischen den zu evaluierenden Aspekten und den einzelnen Evaluierungs-Blöcken ist in Tabelle 10.1 nochmals überblicksweise angeführt. Die Zuordnung hatte jeweils Einfluss auf das Szenario, in dem das Werkzeug angewandt wurde sowie auf das Untersuchungsdesign.

Tabelle 10.1.: Ursprüngliches globales Untersuchungsdesign

	Werkzeug	Modell	Articulation Work
Block 1	x		
Block 2			x
Block 3		x	
Block 4			x
Block 5		x	

Im Zuge der Durchführung der Evaluierung erwies sich die strikte Zuordnung eines Blocks zu genau einem zu evaluierenden Aspekt als nicht durchführbar. Tatsächlich liefern Untersuchungen zu einem (im Sinne der Zielhierarchie) übergeordneten Aspekte (von "unten" nach "oben": Werkzeug – Modell – Articulation Work) immer auch Erkenntnisse zu den untergeordneten zu evaluierenden Aspekten. Die Zuordnung der Evaluierungs-Blöcke zu den Aspekten verändert sich also wie in Tabelle 10.2 angegeben. Diese Zuordnung liegt auch den oben angeführten Beschreibungen der Blöcke zugrunde, in denen jeweils die Beiträge eines Blocks zu den zu evaluierenden Aspekten angegeben wurden.

Tabelle 10.2.: Einfluss der Untersuchungen auf die zu evaluierenden Aspekte

	Werkzeug	Modell	Articulation Work
Block 1	x		
Block 2	x	x	x
Block 3	x	x	
Block 4	x	x	x
Block 5	x	x	

10. Überblick über die empirische Untersuchung

In den folgenden Kapiteln wird nun die Evaluierung der einzelnen Aspekte über die Evaluierungs-Blöcke hinweg im Detail beschrieben. Dabei werden die Hypothesenbildung bzw. die Entwicklung der Hypothesen über die Zeit, die möglichen Ansätze zur Evaluierung der jeweiligen Hypothesen sowie das Untersuchungsdesign das zur Prüfung des Hypothesen führt, beschrieben. Die Kapitel schließen jeweils mit einer Zusammenfassung der Ergebnisse der Hypothesenprüfung und einer Bewertung dieser Ergebnisse im Kontext der globalen Zielsetzung, also der Unterstützung von expliziter Articulation Work.

II. Evaluierung der Verwendbarkeit des Werkzeugs

Im ersten empirischen Teil der Evaluierung wurde die grundlegende Verständlichkeit und Verwendbarkeit des Werkzeug geprüft. Ziel war es hier, konzeptionelle und technische Eigenschaften bzw. Verhaltensweisen des Werkzeugs zu identifizieren, die den Modellierungsprozess behindern oder unterbrechen. Darunter fällt grundsätzlich jede Eigenschaft und jede Verhaltensweise, die die Benutzer zwingt, sich mit dem technischen System an sich zu beschäftigen und von der Erfüllung der eigentlichen Aufgabe ablenkt bzw. diese unterbricht.

Die Untersuchung wurde daneben auch genutzt, um explorativ die inhaltliche Verwendung des Systems zu untersuchen (d.h. wie es für seinen eigentlichen Verwendungszweck, die Modellierung, eingesetzt wurde) und Hypothesen abzuleiten, die in weiteren Schritten getestet werden konnten.

II.I. Hypothesen

In diesem Abschnitt werden die Hypothesen angeführt und begründet, die in diesem Teil der empirischen Untersuchung geprüft werden. Die hier angegebenen Hypothesen gehen auf die Eigenschaften des Werkzeugs in der Verwendung durch die Benutzer ein. Bei der Hypothesenbildung wird auf den Verwendungszweck des Werkzeugs, die Unterstützung der Bildung diagrammatischer Modelle, Rücksicht genommen – die Modelle selbst sind jedoch nicht Gegenstand der Betrachtung, sondern werden erst im nächsten Kapitel behandelt. Nicht berücksichtigt wird außerdem die Verwendung zur Unterstützung von Articulation Work – die Implikationen des Werkzeugs auf diese sind Gegenstand von Kapitel 13.

II.I.I. Konzeptionell begründete Hypothesen

Die folgenden Hypothesen wurden aus der Aufgabenstellung (siehe Kapitel XY) sowie den Anforderungen an das Werkzeug (siehe Kapitel XY) abgeleitet. Neben der Formulierung der Hypothese ist jeweils die Begründung aus der Konzeption des Werkzeugs angeführt.

Der grundlegende Anspruch des Werkzeugs ist es, explizite Articulation Work zu unterstützen. Wie in Teil I dieser Arbeit beschrieben, wird dies über die Externalisierung und Aushandlung von mentalen Modellen realisiert. Ein gängiges Mittel, um mentale Modelle zu repräsentieren, sind diagrammatische Modelle, worunter die Ergebnisse der vorgeschlagenen Methoden zur Externalisierung – Concept Mapping und Strukturlegetechniken – fallen. Das Werkzeug muss also die Repräsentation diagrammatischer Modelle unterstützen.

Hypothese 1 *Das Werkzeug ermöglicht die Repräsentation diagrammatische Modelle.*

„Articulation Work“ ist immer in einen kooperativen Arbeitszusammenhang eingebettet. Die Kollaboration findet dabei nicht nur im produktiven Teil der Arbeit statt, sondern hat immer auch Auswirkungen auf die „Articulation Work“. Jede Unterstützung von „Articulation Work“ muss damit auch in kooperativen Szenarien einsetzbar sein. Dies gilt auch für das hier vorgestellte Werkzeug, das die kooperative Bearbeitung einer Aufgaben (hier: der Externalisierung und Abstimmung mentaler Modelle) ermöglichen muss.

Hypothese 2 *Das Werkzeug ermöglicht kollaboratives Arbeiten an einer Aufgabe.*

Die Aspekte von Arbeit, die im Rahmen von „Articulation Work“ abzustimmen sind, sind unterschiedlicher Natur. Naheliegend ist eine Abstimmung der Abläufe und Schnittstellen zwischen Personen, aber auch nicht-prozedurale Information wie das Verständnis der Struktur und Elemente eines Arbeitszusammenhangs kann Gegenstand von Articulation Work sein. Gleichermaßen gilt für die im Rahmen der Articulation Work abzustimmenden mentalen Modelle – diese bilden die Basis für Handlungsentscheidungen, umfassen aber im Allgemeinen (in Abgrenzung zu Schemata) nicht nur handlungsleitende Information sondern auch Kontextinformation, die die Bewertung der wahrgenommenen Situation ermöglicht. Dementsprechend muss ein Werkzeug zu Unterstützung von expliziter Articulation Work und damit der Externalisierung von mentalen Modellen die Verwendung in unterschiedlichen Kontexten, d.h. für unterschiedliche zu externalisierenden Informationsstrukturen, die in mentalen Modellen abgebildet sind, ermöglichen.

Hypothese 3 *Das Werkzeug ist gleichwertig für Modellierungsaufgaben in unterschiedlichen Kontexten einsetzbar.*

Die ersten drei hier formulierten Hypothesen sind unmittelbar aus der globalen Zielsetzung abgeleitet und bilden die grundlegenden Anforderungen an das Werkzeug bei der Unterstützung von Articulation Work ab. Die nun folgenden Hypothesen sind konzeptionell nicht mehr direkt auf die globale Zielsetzung ausgerichtet sondern stellen auf Funktionalität des Werkzeugs ab, die den Modellbildungsprozess unterstützen soll.

Die erste dieser Hypothesen bildet eine wesentliche Funktionalität des Werkzeugs, nämlich die Möglichkeit durch die Entstehungsgeschichte des erstellten Modells zu navigieren, ab. In der dieser Funktionalität zugrundeliegenden Literatur ((Shipman und Hsieh, 2000), (Klemmer et al., 2002)) wird diese als wesentlich bezeichnet, wenn Externalisierungsprozesse unterbrochen werden, kollaborativ durchgeführt werden oder Dritten die Möglichkeit gegeben werden soll, die Entstehung des Modells nachzuvollziehen. Allen drei Aspekten liegt die Annahme zugrunde, dass in der Historie des Externalisierungsprozesses die dem Ergebnis zugrundeliegenden Ideen und Annahmen zu erkennen sind. Im Kontext dieser Arbeit bedeutet dies, dass aus der Nachverfolgung der Historie des externalisierten Modells die diesem zugrundeliegenden mentalen Modelle verständlich und nachvollziehbar werden.

Hypothese 4 *Die Reflexion des Modellierungsverlaufs ermöglicht das Verständnis der dem Modell zugrundeliegenden mentalen Modelle.*

Auf Basis der Möglichkeit zur Navigation durch die Entstehungsgeschichte des Modells besteht auch die Möglichkeit, vergangene Modellzustände wiederherzustellen. Das Werkzeug unterstützt dabei die Benutzer durch die Ausgabe von schrittweisen Anweisungen, die den aktuellen Modellzustand in den wiederherzustellenden Zustand überführen. Allgemein bietet diese Funktionalität die Möglichkeit, erkannte Fehler im Modell zu korrigieren, ohne dabei bereits repräsentierte Information zu verlieren. Im kollaborativen Einsatz ermöglicht diese Funktionalität, alternative, individuelle Sichten auf den abzustimmenden Sachverhalt zu repräsentieren und dabei die Möglichkeit bieten, einen für alle Beteiligten akzeptablen Ausgangspunkt wiederherzustellen

Hypothese 5 *Die Möglichkeit der Wiederherstellung vergangener Modellzustände fördert die Bereitschaft alternative Repräsentationen auszuprobieren.*

Die letzten beiden Hypothesen dieses Abschnitts sind ausschließlich auf die Verwendung des Werkzeugs an sich ausgerichtet und stehen nicht im Kontext von Articulation Work oder der Unterstützung der Externalisierung mentaler Modelle. Hypothese 6 steht für den in der Zielsetzung formulierten Anspruch, dass das Werkzeug in den Hintergrund treten muss und die Beschäftigung mit der eigentlichen

Aufgabe nicht behindern darf. Dabei wird hier nicht auf den konkreten Anwendungsfall – die Erstellung von Modellen – eingegangen sondern lediglich die allgemeine Funktionsfähigkeit und Bedienbarkeit des Werkzeugs betrachtet. Ersteres ist Gegenstand der Evaluierung der erstellten Modelle, die in Kapitel 12 beschrieben werden.

Hypothese 6 *Das Werkzeug behindert die Modellbildung nicht.*

Hypothese 7 geht davon aus, dass bei wiederholten Verwendung des Werkzeugs Lern- und Gewöhnungseffekte auftreten, die die Verwendung erleichtern, beschleunigen und zu weniger Fehlbedienung führen. Dies ist ein Effekt, der bei jedem Werkzeug zu erwarten ist, dessen zugrundeliegenden Konzepte den Benutzern bewusst sind. Von dieser Voraussetzung kann durch die inhaltliche Einführung der Benutzer in die das Werkzeug prägenden und motivierenden Ideen ausgegangen werden. Damit wäre zu erwarten, dass das Werkzeug bei wiederholtem Einsatz in den späteren Anwendungen effizienter (im Sinne von schneller und Fehlbedienungen vermeidend) verwendet wird.

Hypothese 7 *Wiederholte Verwendung des Werkzeugs führt zu schnellerer Modellbildung und weniger Fehlbedienungen.*

11.1.2. Explorativ gebildete Hypothesen

Neben den aus der Aufgabenstellung abgeleiteten Hypothesen wurden einige Hypothesen auch während der Durchführung der einzelnen Evaluierungs-Blöcke gebildet. Diese Hypothesen sind spezifischer auf einzelne Aspekte des Werkzeugs abgestellt und decken beobachtete Auffälligkeiten und Missverständnisse in der Verwendung des Werkzeugs ab.

Die erste in diesem Zusammenhang beobachtete Auffälligkeit betrifft die Herstellung von Verbindern zwischen einzelnen Modellelementen. Wie in Abschnitt 6.3.3 beschrieben, existieren zwei Möglichkeiten, diese Funktion auszuführen. Einerseits können die beiden Modellelemente, die verbunden werden sollen, mit Markierungstokens ausgewählt werden, worauf hin eine Verbindung hergestellt werden. Andererseits können Verbinder auch durch das Zusammenführen der zu verbindenden Blöcke (bis sich deren Breitseiten berühren) hergestellt werden. In der ersten Implementierung des Werkzeugs, die im Evaluierungs-Block 1 und im ersten Teil des zweiten Blocks verwendet wurde, war lediglich die erste Variante verfügbar. Die Möglichkeit zur Herstellung von Verbindern wurde in den in diesen Blöcken durchgeföhrten Anwendungen kaum eingesetzt. Dies führte einerseits zur Bildung der Hypothese 14

(siehe Abschnitt 12.1.2), andererseits wurde bei ersten Auswertungen der Beobachtungen der im Verhältnis zum übrigen Modellierungs-Prozess hohe Zeit-Aufwand bei der Herstellung von Verbindern offensichtlich. Dieser Aufwand ist den Maßnahmen zur Stabilisierung der Erkennungsleistung des Werkzeugs geschuldet und kann mit den eingesetzten Interaktionsablauf nicht reduziert werden. Aufgrund einer Anregung eines Untersuchungsteilnehmers wurde deshalb die oben beschriebene zusätzliche Möglichkeit zur Herstellung von Verbindungen implementiert. Zu untersuchen ist nun, ob diese Maßnahme die Nutzung von Verbindern bei der Modellbildung tatsächlich erhöht.

Hypothese 8 *Die Einführung der alternativen Möglichkeit zur Verbindungsherstellung erhöht die Nutzung von Verbindern bei der Modellerstellung.*

Die zweite hier aufgestellte Hypothese betrifft eine Auffälligkeit bei der Verwendung des Löschtokens. Das Löschtoken wird verwendet, um das Werkzeug in einen Modus zu versetzen, in dem Verbinder gelöscht werden können. Schon die konzeptionelle Einordnung des Werkzeugs in Kapitel 9 zeigte Potential für Missverständnisse in der Verwendung dieses Tokens (siehe z.B. die Abschnitte 9.9 und 9.11). Zusammengefasst liegt die aus der Theorie ableitbare Problematik darin, dass durch die äußere Form des Tokens – einem Radiergummi – eine Metapher für dessen Verwendung („ausradieren“ von Elementen) suggeriert wird, die in dieser Form im Werkzeug nicht umgesetzt ist, da das Token lediglich als Schalter fungiert. Erste Beobachtungen deuteten darauf hin, dass die Verwendung des Löschtoken tatsächlich unverständlich oder missverständlich zu sein scheint. Die zugehörige Hypothese ist positiv formuliert, zu erwarten wäre demnach, dass sie verworfen werden muss.

Hypothese 9 *Das Löschtoken ermöglicht intuitives Löschen von Modellelementen.*

I I.2. Untersuchungsdesign und Durchführung

In diesem Abschnitt wird auf Basis der eben formulierten Hypothesen ein Untersuchungsdesign abgeleitet und die Durchführung der Untersuchung beschrieben. Im ersten Schritt werden die Hypothesen hinsichtlich der Möglichkeiten ihrer Beurteilung betrachtet und dabei die zu erhebenden Variablen identifiziert. Im nächsten Schritt folgt die Operationalisierung, in der die konkrete Messung bzw. Beurteilung der einzelnen Variablen festgelegt wird. Aus dieser Operationalisierung können die durchzuführenden Untersuchungen abgeleitet werden. An dieser Stelle erfolgt auch die Einordnung in die in Kapitel 10 beschriebenen Evaluierungsblöcke. Im letzten Teil dieses Abschnitts wird die eigentliche Durchführung beschrieben, wobei im

Speziellen auf deskriptive Parameter der Untersuchung eingegangen wird, die im Kontext der Werkzeugverwendung von Interesse sind, aber nicht oder nur als Teil der Berechnungsgrundlage in die Auswertung der Hypothesen eingehen.

11.2.1. Operationalisierung

In diesem Abschnitt wird für jede Hypothese identifiziert, welche Variablen bei ihrer Beurteilung berücksichtigt werden müssen. Für jeder der Variablen wird in der Folge die konkrete Mess- und Auswertungsmethode festgelegt (auf Basis der in Abschnitt 10.3 beschriebenen Verfahren) und jene Evaluationsblöcke festgelegt, die für die jeweilige Untersuchung herangezogen wurden.

Bei der Definition der Variablen für die einzelnen Hypothesen sind unabhängige, abhängige und Stör-Variablen zu unterscheiden. In einer empirischen Untersuchung beschreiben *unabhängige Variablen*, jene Parameter die in einer Untersuchung bewusst variiert bzw. festgelegt werden können, um eine Hypothese zu testen zu. Die Auswirkungen der Festlegung der unabhängigen Variablen zeigt sich an den *abhängigen Variablen*. Die Messung dieser Variablen erlaubt es, die jeweilige Hypothese zu beurteilen. *Störvariablen*, sind unabhängige Variablen, die jedoch nicht frei gewählt werden können, sondern als grundsätzlich unbekannter Einflussfaktor auf die abhängigen Variablen wirken. Bei der Beurteilung der Hypothesen muss der Einfluss von Störvariablen möglichst minimiert bzw. bei der Auswertung berücksichtigt werden.

Im Überblick sind folgende unabhängige Variablen in dieser Untersuchung zu berücksichtigen:

- Werkzeug
- Methodik
- Aufgabe
- Modellierungsvorkenntnisse

Abhängig von der jeweils zu testenden Hypothese können diese Variablen tatsächlich als unabhängige Variablen fungieren oder Störvariablen sein, deren Einfluss minimiert werden muss. Die abhängigen Variablen sind spezifisch für die jeweilige Hypothese festgelegt und werden in den jeweiligen nun folgenden Abschnitten eingeführt.

Repräsentation diagrammatischer Modelle

Gegenstand dieses Abschnitts ist die Operationalisierung der Hypothese 1. Gegenstand der Überprüfung ist hier die Eignung des Werkzeugs für die Repräsentation diagrammatischer Modelle.

Die **unabhängige Variablen** sind in diesem Fall das *Werkzeug* und die *Modellierungsaufgabe*. Das Werkzeug wird dabei in allen berücksichtigten Untersuchungen eingesetzt. Die Modellierungsaufgaben müssen so formuliert sein, dass es grundsätzlich möglich ist, sie durch die Beschreibung in einem diagrammatischen Modell zu erfüllen. Keinen Einfluss auf die Untersuchung haben die eingesetzte Methodik sowie eventuell vorhandene Modellierungsvorkenntnisse, da die grundsätzlich Möglichkeit der Erstellung diagrammatischer Modelle unabhängig von der Art der Verwendung und von der Kompetenz der Benutzer ist.

Die **abhängigen Variable** ist hier die *Repräsentation*, die mit Hilfe des Werkzeugs erstellt wurde. Ein diagrammatisches Modell zeichnet nach (Larkin und Simon, 1987) aus, dass in ihm Konzepte und deren Zusammenhänge visuell-graphisch dargestellt werden können (in Abgrenzung zu textuellen Beschreibungen). Zur Bewertung der Hypothese werden deshalb die erstellten Repräsentationen herangezogen und überprüft, ob sie den Anforderungen an ein diagrammatisches Modell – das Vorhandensein von Konzepten und Beziehungen zwischen diesen – erfüllen.

Kollaboratives Arbeiten

Gegenstand dieses Abschnitts ist die Operationalisierung der Hypothese 2. Dabei wird überprüft, ob das Werkzeug kollaboratives Arbeiten an einer Modellierungsaufgabe erlaubt.

Unabhängige Variablen sind hier das *Werkzeug*, die *Methodik* sowie die *Modellierungsaufgabe*. Während Werkzeug und Methode wie in den vorhergehenden Kapiteln konzipiert eingesetzt werden, muss eine Modellierungsaufgabe gewählt werden, die kollaboratives Arbeiten sinnvoll ermöglicht. Etwaige Modellierungsvorkenntnisse haben keinen Einfluss auf die Beurteilung der hier betrachteten Hypothese.

Als **abhängige Variablen** eignen sich in diesem Fall die *Zeitverteilung der Beteiligung* der einzelnen Benutzer am Modellierungsvorgang, die *Anzahl der Übergaben der Modellierungsinitiative* sowie das *Verhalten der Benutzer bei simultaner Manipulation* eines Modells auf der Modellierungsoberfläche. Die erstgenannte Variable kann quantitativ gemessen werden, wobei eine tendenziell zeitlich gleichverteilte Einbindung der Beteiligten in die Modellbildung für die Annahme der Hypothese spricht. Die Anzahl der Übergaben der Modellierungsinitiative zeigt an, wie oft bei der Modellierung die Initiative zwischen den Beteiligten gewechselt hat. Hohe Zahlen sprechen hier für eine Annahme der Hypothese. Zusätzlich kann mittels die

dritten Variable qualitativ beurteilt werden, ob und wie simultane Manipulation des Modells durch mehrere Benutzer möglich ist.

Einsetzbarkeit in unterschiedlichen Kontexten

Gegenstand dieses Abschnitts ist die Operationalisierung der Hypothese 3. Diese Hypothese zielt dabei auf die Eignung des Werkzeugs zur Modellbildung in unterschiedlichen Kontexten, d.h. für unterschiedliche Modellierungsaufgaben.

Die **unabhängigen Variablen** sind dabei wiederum das *Werkzeug* und die *Methodik*, die wie konzipiert eingesetzt werden, sowie die *Modellierungsaufgabe*, als jene Variable, die im Zuge der Untersuchung variiert wird. Etwaige *Modellierungsvorkenntnisse* können insofern als **Störvariable** wirken, als dass sie die Wahrnehmung der Eignung des Werkzeugs für eine bestimmte Aufgabe positiv oder negativ beeinflussen.

Abhängige Variablen sind in diesem Fall die *Wahrnehmung der Eignung* durch die Benutzer, die qualitativ beurteilt wird, und die *Korrelation der Größe der erstellten Modelle mit der benötigten Modellierungsdauer*. Korreliert die Modellgröße positiv mit der Modellierungsdauer, so ist der Zeitanteil, der zu Beschäftigung mit dem Werkzeug selbst (und nicht mit der Modellierungsaufgabe) tendenziell stabil. Daraus kann abgeleitet werden, dass das Werkzeug die verglichenen Modellierungsaufgaben gleich gut (oder schlecht) unterstützt.

Reflexion des Modellierungsverlaufs

Gegenstand dieses Abschnitts ist die Operationalisierung der Hypothese 4. Dabei wird überprüft, ob die Möglichkeit zur Reflexion des Modellierungsverlaufs das Verständnis der zugrundeliegenden mentalen Modelle ermöglicht bzw. verbessert.

Sowohl *Werkzeug* und *Methodik* sind als **unabhängige Variablen** zu identifizieren. Die Modellierungsaufgabe und Modellierungsvorkenntnisse haben keine Auswirkung auf die Untersuchung.

Als **abhängige Variable** kann hier der *wahrgenommene Erfolg bei der Vermittlung eines mentalen Modells* herangezogen werden. Zur Beurteilung desselben wird eine Modellierungssituation geschaffen, in der eine Person für sich individuell das Werkzeug zur Externalisierung eines mentalen Modells benutzt. In einem zweiten Schritt wird eine zweite, zuvor nicht beteiligte Person aufgefordert, den Inhalt der auf der Modellierungsoberfläche vorhandenen Repräsentation zu interpretieren, wobei der Modellierungsverlauf herangezogen werden darf, eine Interaktion mit dem ursprünglichen Modellierer jedoch nicht gestattet ist. Im dritten Schritt beurteilt der ursprüngliche Modellierer die Adäquatheit der Interpretation und trifft

so eine Aussage über den Erfolg des Transfers des mentalen Modells. In Kombination mit der Information über Art und Ausmaß der Benutzung der Möglichkeit zum Zugriff auf den Modellierungsverlaufs lässt sich eine qualitative Aussage über die Effekte dieser Funktionalität treffen.

Wiederherstellung vergangener Modellzustände

Gegenstand dieses Abschnitts ist die Operationalisierung der Hypothese 5. Gegenstand der Überprüfung ist die Verwendung der Wiederherstellungsfunktionalität zum Zwecke der versuchsweisen Veränderung des Modells.

Als **unabhängige Variablen** wirken hier wie zuvor *Werkzeug* und *Methodik*, die *Modellierungsaufgabe* hat insofern Auswirkungen, als dass sie so gestaltet sein muss, dass sinnvoll unterschiedliche Repräsentationen gebildet werden können. Modellierungsvorkenntnisse haben keine Auswirkungen auf diese Untersuchung.

Die **abhängige Variable**, die zur Beurteilung dieser Hypothese herangezogen wird, ist die *Anzahl der Verwendungen der Wiederherstellungsfunktionalität zur Korrektur inhaltliche verworfener Repräsentationen*. Höhere Werte deuten hier auf eine Annahme der Hypothese hin. Zusätzlich können qualitative Aussagen zur Nutzung dieser Funktionalität und deren *wahrgenommenen Nutzen* zur Beurteilung verwendet werden.

Nicht-Behinderung

Gegenstand dieses Abschnitts ist die Operationalisierung der Hypothese 6. Dabei wird überprüft, ob bei der Verwendung des Werkzeugs dieses in den Aufmerksamkeitsfokus der Benutzer tritt oder sich diese auf die eigentliche Modellierungsaufgabe konzentrieren können.

Unabhängige Variablen sind in diesem Fall das *Werkzeug* und die angewandte *Modellierungsmethode*. Die Modellierungsaufgabe hat keinen Einfluss auf die Überprüfung dieser Hypothese, lediglich etwaig vorhandene *Modellierungsvorkenntnisse* können als **Störvariable** wirken, da sie Einfluss auf die erwartete Funktionalität des Werkzeugs haben kann.

Zur Beurteilung, ob bzw. inwieweit das Werkzeug die Modellbildung behindert, werden qualitative beurteilbare **abhängige Variablen** herangezogen. Die Anzahl von *Fehlfunktionen des Werkzeugs* bzw. das *Auftreten von Systemabstürzen* als Indikator für eine behindernde Wirkung des Werkzeugs herangezogen werden. Das Auftreten von Missverständnissen und daraus resultierende Fehlbedienungen können ebenfalls eine Behinderung des Modellierungsvorgangs interpretiert werden. Zudem wird das artikulierte Empfinden der Benutzer als Maß für die wahrgenommene Be-

hinderung durch das Werkzeug herangezogen. Der Einfluss von Modellierungsvorkenntnissen kann in diesem Fall nicht mit rechnerischen Maßnahmen kompensiert werden. Etwaige Vorkenntnisse werden dementsprechend bei der Auswertung angeführt und müssen bei der Diskussion der Hypothese berücksichtigt werden.

Gewöhnung an das Werkzeug

Gegenstand dieses Abschnitts ist die Operationalisierung der Hypothese 7. Dabei wird überprüft, ob wiederholte Benutzung des Werkzeugs Auswirkung auf die Qualität der Interaktion hat. Eine Erhöhung der Qualität äußert sich in schnellerer Modellbindung und weniger Fehlbedienung.

Unabhängige Variablen sind in diesem Fall *die Verwendung des Modellierungswerkzeugs* und die angewandte *Methodik*. **Störvariablen**, die unerwünschten Einfluss auf die Messung haben, sind potentiell die *Modellierungsaufgabe* (die Einfluss auf die Dauer der Anwendung hat und außerdem die Verwendung unterschiedlicher Funktionalität bedingen kann) und eine etwaige *veränderte Funktionalität des Werkzeugs* zwischen den verglichenen Evaluierungsblöcken (die die Interaktion einerseits erleichtern kann, andererseits aber auch zu Fehlbedienung aufgrund von unbekannten Interaktionsmustern führen kann).

Die **abhängigen Variablen** zur Beurteilung der Qualität der Interaktion sind einerseits die *Anzahl der Fehlbedienungen* des Werkzeugs pro Zeiteinheit und andererseits die *Arbeitsdauer am Werkzeug*¹ in Abhängigkeit der Modellgröße. Die Normierung der abhängigen Variablen ist notwendig, um vergleichbare Werte für unterschiedliche Werkzeug-Anwendungen zu erhalten. Sinken beide Werte zwischen zwei Evaluierungsblöcken, die auf der gleichen Stichprobe aufbauen, signifikant, so kann die Hypothese angenommen werden. Um den Einfluss der Störvariablen zu reduzieren, ist es sinnvoll, in beiden Blöcken eine identische Modellierungsaufgabe zu stellen und die Funktionalität des Werkzeugs nicht zu verändern. Identische Modellierungsaufgaben können durch die wiederholte inhaltliche Beschäftigung mit der Aufgabe zu schnellerer Arbeit bzw. zu kompakteren Modellen führen. Diesem weiteren Störfaktor kann durch die Berücksichtigung der reinen Arbeitszeit am Werkzeug sowie der Normierung derselben in Abhängigkeit der Modellgröße entgegengewirkt werden.

Herstellung von Verb bindern

¹Die Arbeitsdauer am Werkzeug ist im Gegensatz zur gesamten Modellierungsdauer um jenen Zeitanteil reduziert, in dem die Teilnehmer interagieren, ohne am Werkzeug zu arbeiten.

Gegenstand dieses Abschnitts ist die Operationalisierung der Hypothese 8. Mit Hilfe dieser Hypothese soll überprüft werden, ob die Einführung der alternativen Möglichkeit zur Herstellung von Verbbindern deren Verwendung signifikant gesteigert hat.

Die für diese Hypothese relevanten **unabhängigen Variable** ist die *Verwendung des Werkzeugs*. **Störvariablen** sind die *Modellierungsaufgabe* (da sie die Anzahl der benötigten Verbinder beeinflussen kann), die *Methodik* (da sie die Verwendung von Verbbindern im Allgemeinen bedingen oder vermeiden kann) und *eventuell vorhandene Modellierungsvorkenntnisse* (da diese Einfluss auf die Struktur des Modells haben können). Um den Einfluss der Störvariablen zu reduzieren, wird die Messung zwischen zwei Evaluierungsblöcken vorgenommen, in denen die gleiche Stichprobe mit der gleichen Aufgabenstellung das Werkzeug mit der gleichen Methodik anwandte. Lediglich die Funktionalität des Werkzeugs wurde zwischen den beiden Anwendungen um den alternativen Weg zur Herstellung von Verbbindern erweitert.

Als **abhängige Variable** ist zur Beurteilung des Ausmaßes der Verwendung von Verbbindern in diesem Fall die *Connectedness* des Modells verwendbar. Die Connectedness ist das Verhältnis zwischen der Anzahl der im Modell verwendeten Verbinder und der Anzahl der verwendeten Knoten (Modellierungselemente). Hier ist zu prüfen, ob die Connectedness in jenem Evaluierungs-Block, in dem der alternative Weg zur Herstellung von Verbindungen verfügbar war, signifikant höher ist als in jenem Block, in dem sie nicht verfügbar war.

Löschtoken

Gegenstand dieses Abschnitts ist die Operationalisierung der Hypothese 9. Dabei wird überprüft, ob das Löschtoken intuitiv korrekt verwendet wird oder ob es zu Fehlinterpretationen kommt.

Die Verwendbarkeit des Löschtokens ist unabhängig von der Modellierungsaufgabe, der angewandten Methodik und auch von eventuell vorhandenen Modellierungsvorkenntnissen. Die einzige relevante **unabhängige Variable** ist die *Verwendung des Werkzeugs* selbst.

Zur Beurteilung der intuitiven Verwendbarkeit werden quantitative und qualitative Merkmale der Werkzeugverwendung herangezogen. Eine quantitativ beurteilbare **abhängige Variable** ist der Anteil der Fehlbedienungen des Löschtokens in Bezug auf alle Anwendungen des Werkzeugs, in denen es grundsätzlich verwendet wurde. Qualitativ wird die Art des Missverständnisses, das zu den jeweiligen Fehlbedienungen führt, beurteilt.

Zur Messung der quantitativen Variablen wird für jede Anwendung die Anzahl der Fehlbedienungen erhoben, die durch das Löschtoken verursacht wurden. Die-

ser Wert wird in Bezug zur Gesamtanzahl der Fehlbedienungen gesetzt, so dass der Anteil der durch das Löschtönen verursachten Fehlbedienungen berechnet werden kann. Bei "gleich guter" intuitiver Bedienbarkeit aller Werkzeuge wäre eine Gleichverteilung der Fehler zu erwarten. Ist der Anteil der durch das Löschtönen verursachten Fehlbedienungen höher als der Anteil, der bei Gleichverteilung zu erwarten wäre, so deutet dies auf eine Ablehnung der Hypothese hin.

Qualitativ werden Modellierungssituationen betrachtet, in denen das Löschtönen zum Einsatz kommt. Auf Basis von Transkripten der Interaktion zwischen den Benutzern und dem Werkzeug, bei denen es zu Fehlbedierungen kam, werden die aufgetretenen Missverständnisse explizit identifiziert.

II.2.2. Datenbasis

Als Grundlage der Überprüfung der Hypothesen werden hier die Evaluierungs-Blöcke 1 bis 5 verwendet. Dabei wurden für die quantitativ zu prüfenden Variablen die Blöcke 2 und 3 herangezogen. In die qualitative Auswertung der Ergebnisse wurden hingegen alle Blöcke (1-5) mit einbezogen.

II.2.3. Durchführung

In diesem Abschnitt werden die für diesen Evaluierungs-Teil relevanten deskriptiven Parameter der berücksichtigten Anwendungs-Blöcke angeführt.

Stichprobe

	Aushandlung (1. Durchgang)	Aushandlung (2. Durchgang)	Concept Mapping	Gesamt
$n_{Anwendungen}$	9	9	17	35
$n_{Teilnehmer}$	19	18	47	84

Dauer der Werkzeugverwendung

Die Dauer der Werkzeug-Verwendung wurde den Blöcken 2 („Aushandlung“) und 3 („Concept Mapping“) erhoben. Die Bearbeitungszeit ist wie folgt verteilt (siehe auch Abbildung II.1²):

²In allen Boxplots gilt folgende Notation:

- breite horizontale Linie: Bereich zwischen 25%- und 75%-Quantil

	Aushandlung (1. Durchgang)	Aushandlung (2. Durchgang)	Concept Mapping
t_{min}	11m 54s	2m 5s	14m 1s
\bar{t}	20m 53s	9m 49s	32m 32s
$s(t)$	4m 18s	5m 20s	10m 7s
t_{max}	27m 30s	19m 29s	45m 0s

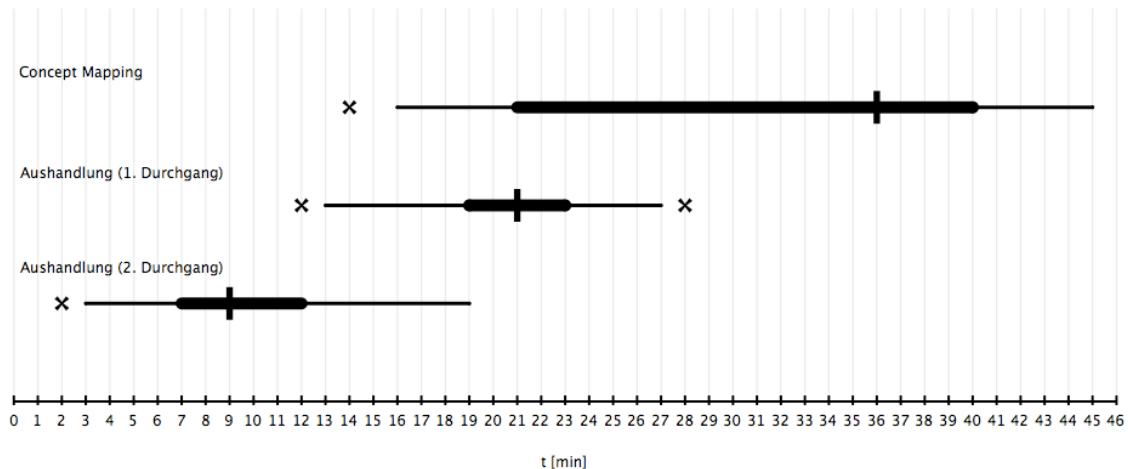


Abbildung II.1.: Dauer der Werkzeugverwendung – Überblick

Die erhobene Dauer der Werkzeug-Verwendung teilt sich in einen Anteil, an dem tatsächlich mit dem Werkzeug interagiert wird und einen Anteil, der anderen Tätigkeiten (wie inhaltlicher Diskussion, Bedeutungsaushandlung,...) gewidmet ist. Diese beiden Anteile sind in den einzelnen Blöcken wie folgt verteilt (siehe auch die Abbildungen II.2 und II.3):

II.3. Ergebnisse

II.3.1. Repräsentation diagrammatischer Modelle

Gegenstand dieses Abschnitts ist die Operationalisierung der Hypothese I.

- breite vertikale Linie: Median
- linke schmale Linie: Bereich zwischen 2,5%- und 25%-Quantil
- rechte schmale Linie: Bereich zwischen 75%- und 97,5%-Quantil
- Kreuze: Ausreißer (außerhalb 2,5%- und 97,5%-Quantil)

II. Evaluierung der Verwendbarkeit des Werkzeugs

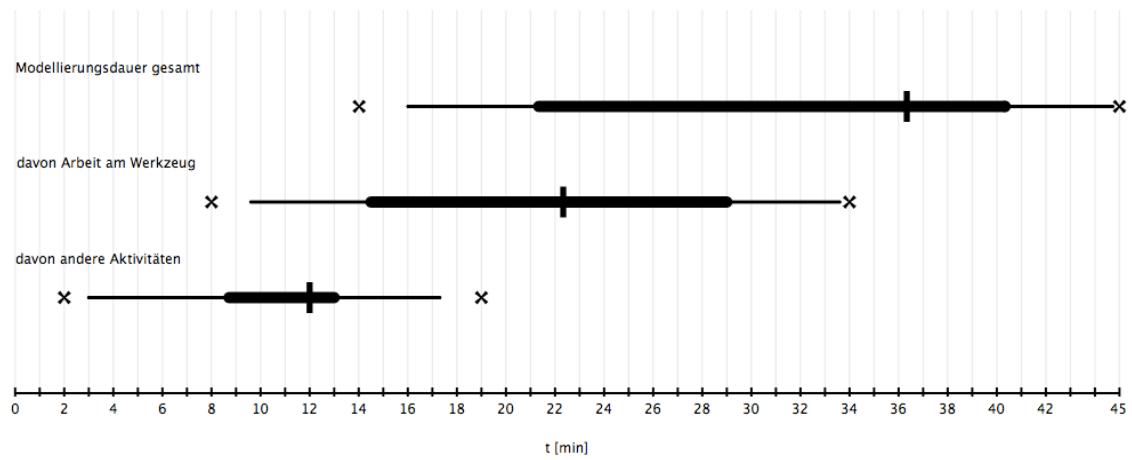


Abbildung II.2.: Dauer der Werkzeugverwendung – Concept Mapping

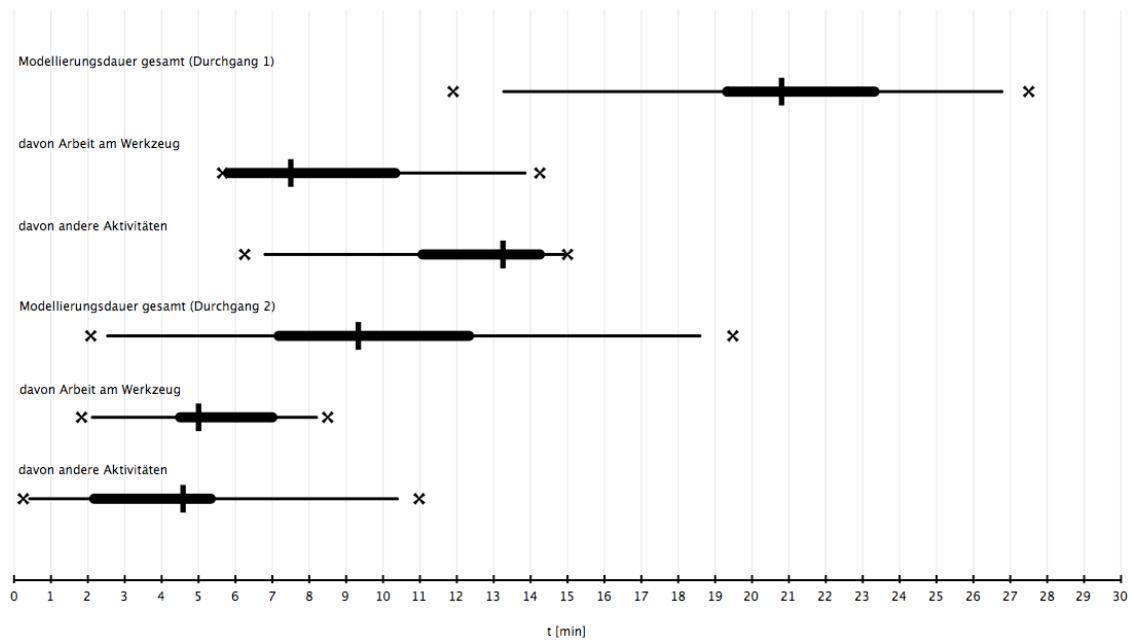


Abbildung II.3.: Dauer der Werkzeugverwendung – Aushandlung

Als Grundlage dieser Untersuchung dienen die Ergebnisse der ersten drei Evaluierungsblöcke, da die Ergebnisse von Block 4 und 5 zum Zeitpunkt der Auswertung noch nicht vorlagen.

Auswertung

Diskussion

Ergebnis

I I.3.2. Kollaboratives Arbeiten

Gegenstand der hier beschriebenen Untersuchung ist Hypothese 2 („Das Werkzeug ermöglicht kollaboratives Arbeiten an einer Aufgabe.“). Zur Untersuchung herangezogen wurden die Werkzeuganwendungen aus den Evaluierungsblöcken 2 ($n = 9$) und 3 ($n = 17$), wobei in Block 2 in Gruppen zu zwei Personen modelliert wurde (in einem Fall drei Personen), in Block 3 in Gruppen zu drei Personen (in drei Fällen nur zwei Personen).

Auswertung

Grundlage der ersten Auswertung ist die Verteilung der Modellierungsdauer zwischen den Teilnehmern. Um die unterschiedliche Gesamt-Modellierungsdauer in den einzelnen Anwendungen zu kompensieren, wurden die Berechnungen auf Basis der prozentuellen Zeitanteile der einzelnen Teilnehmer durchgeführt. Die einzelnen Datensätze wurden so sortiert, dass die anteilmäßige Modellierungsdauer von Teilnehmer A bis Teilnehmer C (bzw. B) abnimmt. In den einzelnen Evaluierungsblöcken ergeben sich die in Abbildung 11.4 dargestellten Verteilungen.

Zu prüfen ist hier, ob die Zeit-Anteile der einzelnen Teilnehmer signifikant unterschiedlich sind. Dazu wird für jeden Block die Signifikanz zwischen den Verteilung der einzelnen Teilnehmerklassen berechnet (eine Teilnehmerklasse setzt sich aus all jenen Teilnehmern zusammen, die am längsten, am zweitlängsten bzw. am dritt-längsten aktiv waren). Aufgrund der geringen Stichprobengröße kommt zur Prüfung der Signifikanz der t-Test nicht in Frage, es wird der *Wilcoxon-Test* herangezogen. Der t-Test setzt außerdem Normalverteilung der Prüfgrößen voraus, was zumindest bei einer der Verteilungen nicht der Fall ist (Shapiro-Wilk-Test für $conn_{B22}$: $p = 6.29e^{-5}$, damit ist von Nicht-Normalverteilung auszugehen).

II. Evaluierung der Verwendbarkeit des Werkzeugs

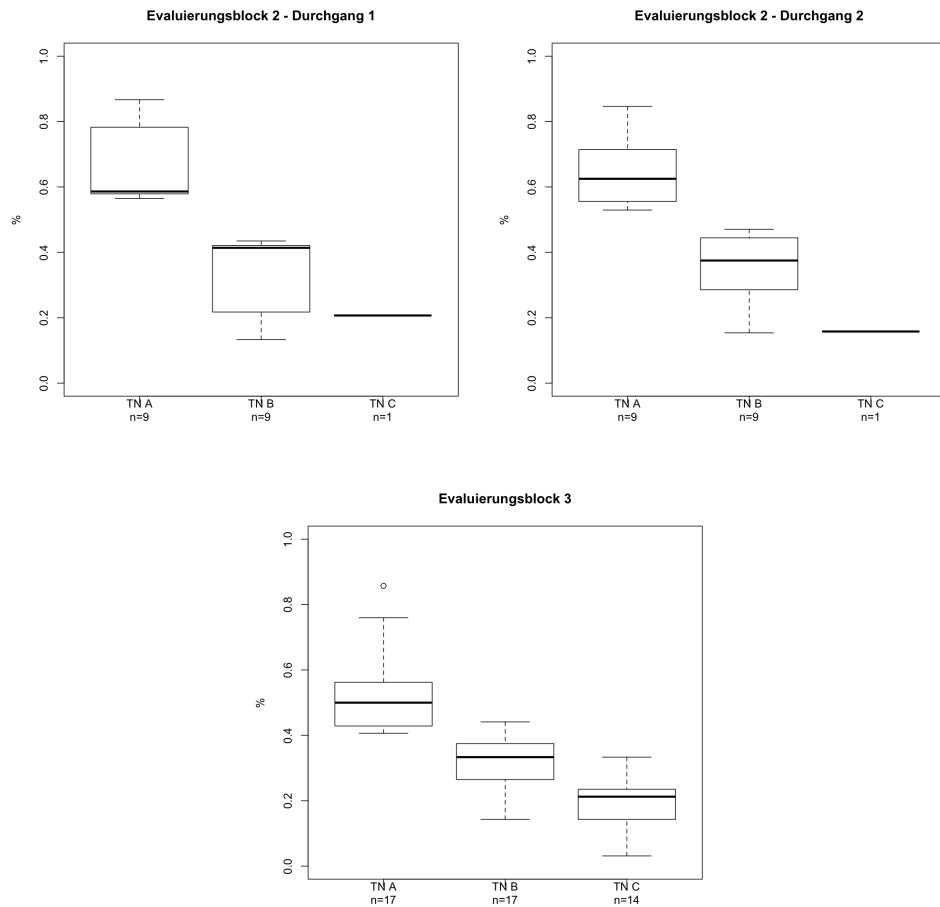


Abbildung II.4.: Zeitverteilung zwischen den Teilnehmern

Diskussion

Ergebnis

I I.3.3. Herstellung von Verbindern

Gegenstand der hier beschriebenen Untersuchung ist Hypothese 8 („Die Einführung der alternativen Möglichkeit zur Verbindungsherstellung erhöht die Nutzung von Verbindern bei der Modellerstellung.“). Zur Untersuchung herangezogen wurden die Werkzeuganwendungen aus Evaluierungsblock 2 ($n = 9$). Dieser wurde gewählt, da in diesem Block alle Teilnehmer das Werkzeug zweimal mit der gleichen Aufgabenstellung anwandten, wobei in der ersten Anwendungsrunde lediglich die ursprüngliche Funktionalität zur Herstellung von Verbindern verfügbar war, in der zweiten Runde aber bereits der alternative Funktionalität implementiert war. Zur weiteren Überprüfung der Ergebnisse werden außerdem die Ergebnisse aus Block 3 ($n = 17$) herangezogen, bei dessen Durchführung ebenfalls bereits die alternative Funktionalität verfügbar war.

Auswertung

Grundlage der Auswertung ist das Modellmerkmal „Connectedness“, worunter hier das Verhältnis zwischen der Anzahl der in einem Modell verwendeten Verbindern und den verwendeten Modellelementen verstanden wird. In den einzelnen Evaluierungsblöcken verteilt sich die Connectedness wie in den Abbildungen II.5, II.6 und II.7 dargestellt.

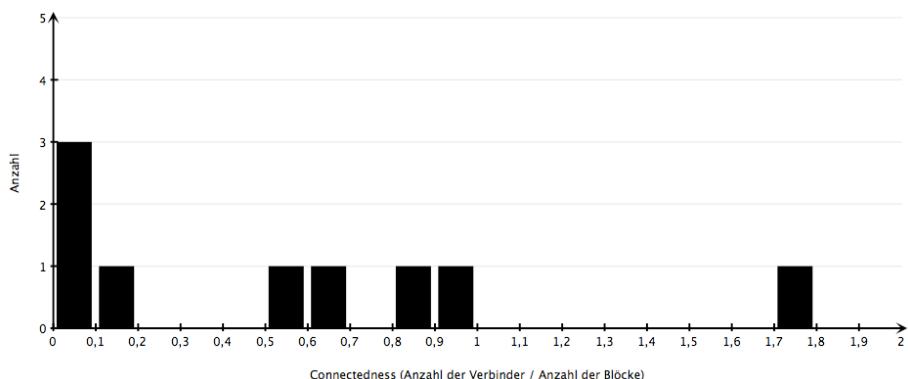


Abbildung II.5.: Connectedness in Evaluierungsblock 2 - Durchgang 1

Zu prüfen ist, ob die Connectedness in jenem Evaluierungs-Blöcken bzw.-Durchgängen, in denen die alternative Funktionalität zur Verbindungs-Herstellung verfügbar war,

II. Evaluierung der Verwendbarkeit des Werkzeugs

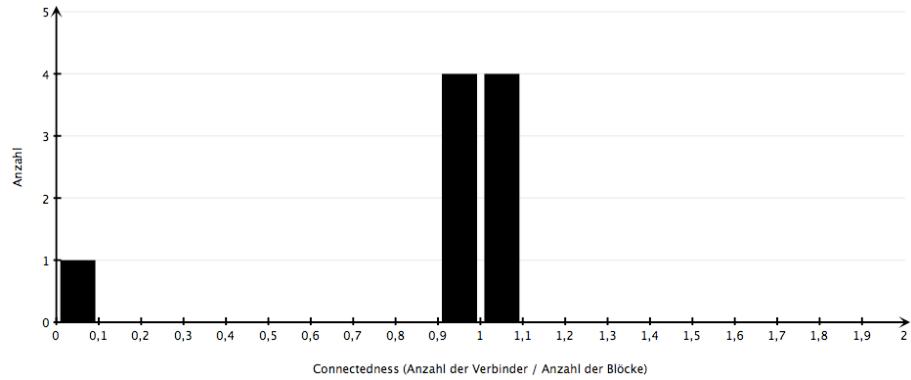


Abbildung II.6.: Connectedness in Evaluierungsblock 2 - Durchgang 2

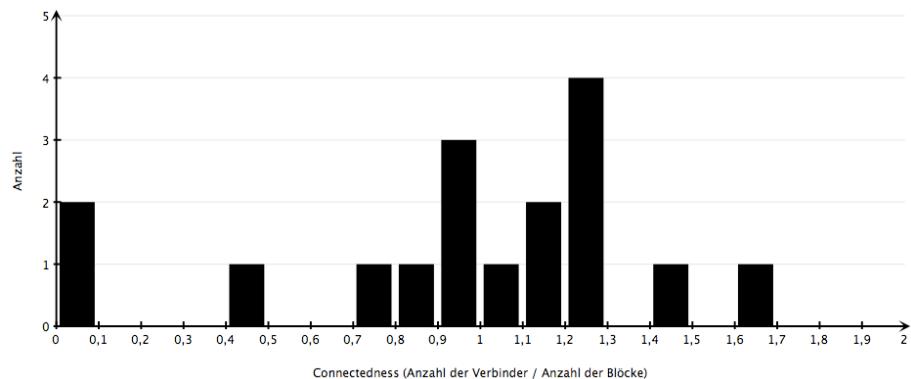


Abbildung II.7.: Connectedness in Evaluierungsblock 3

signifikant höher ist, als in jenen, in denen dies nicht der Fall war. Berechnet wird die Signifikanz zwischen den Ergebnissen der beiden Durchgänge von Block 2 ($conn_{B21}$ und $conn_{B22}$) sowie zwischen den Ergebnissen ersten Durchgang von Block 2 und den Ergebnissen von Block 3 ($conn_{B3}$). Im zweiten Fall ist zu beachten, dass die Aufgabenstellung nicht identisch war und somit eine potentielle Störvariable wirksam wird. Aufgrund der geringen Stichprobengröße kommt zur Prüfung der Signifikanz der t-Test nicht in Frage, es wird der *Wilcoxon-Test* herangezogen. Der t-Test setzt außerdem Normalverteilung der Prüfgrößen voraus, was zumindest bei einer der Verteilungen nicht der Fall ist (Shapiro-Wilk-Test für $conn_{B22}$: $p = 6.29e^{-5}$, damit ist von Nicht-Normalverteilung auszugehen).

Die Null-Hypothese des Wilcoxon-Tests ist, dass die beiden Verteilungen identisch verteilt sind. Entsprechend dem erwarteten Ergebnis (dass die mit der alternativen Funktionalität durchgeführten Anwendungen höhere Connectedness aufweisen) wurde die Alternativ-Hypothese so festgelegt, dass sie angenommen wird, wenn die Verteilung des zweiten Blocks gegenüber dem ersten Block nach rechts verschoben (also wertmäßig höher) ist.

Der Wilcoxon-Test für ungepaarte Stichproben ergibt für $conn_{B21}$ und $conn_{B22}$ und der eben beschriebenen Alternativ-Hypothese $p = 0.9854$ – die Alternativ-Hypothese ist damit anzunehmen, die zweite Verteilung (jene mit Einsatz der alternativen Funktionalität der Verbindungsherstellung) weist eine signifikant höhere Connectedness auf als die erste Verteilung (ohne diese Funktionalität).

Für $conn_{B21}$ und $conn_{B3}$ ergibt der Wilcoxon-Test für ungepaarte Stichproben mit der gleichen Alternativ-Hypothese $p = 0.98$ – auch hier ist die Alternativ-Hypothese anzunehmen.

Für $conn_{B22}$ und $conn_{B3}$ ergibt der Wilcoxon-Test für ungepaarte Stichproben mit der gleichen Alternativ-Hypothese $p = 0.7586$ – auch hier ist die Alternativ-Hypothese anzunehmen.

Diskussion

Aufgrund der Ergebnisse der berechneten Signifikanztests ist die Hypothese anzunehmen. Mit der Einführung der alternativen Möglichkeit zur Herstellung von Verbindungen war in den einzelnen Anwendungen des Werkzeugs eine Zunahme der Verwendung von Verbinder zu beobachten. Während die Benutzer bei der ursprünglichen Funktion zur Herstellung von Verbindungen zum Großteil auf diese verzichteten (auch bereits in Evaluierungsblock 1), wurden Verbinder unabhängig von der Aufgabenstellung mit der Einführung der alternativen Funktionalität verstärkt eingesetzt.

Die Connectedness eignet sich als Parameter zur vergleichenden Beurteilung des Ausmaßes der Verwendung von Verbindern, da durch die Einbeziehung der Größe des Modells (repräsentiert durch die Anzahl der verwendeten Modellelemente) in die Berechnung den Wert für unterschiedliche Modelle vergleichbar macht.

Einfluss auf die Höhe der Connectedness hat aber die Aufgabenstellung, die zur Bildung des Modells führt. Unterschiedliche Modellierungsaufgaben führen zu unterschiedlichen Modell-Topologien, die sich wiederum in der Anzahl der verwendeten Verbinder auswirkt. Dies zeigt sich am Ergebnis des Wilcoxon-Tests für $conn_{B22}$ und $conn_{B3}$ – in beiden Fällen stand die alternative Möglichkeit zur Verbindungsherstellung zur Verfügung $conn_{B3}$ ist trotzdem signifikant höher als $conn_{B22}$. Die kann darin begründet liegen, dass die Concept-Mapping-Aufgabe aus $conn_{B3}$ eher zu stärker verbundenen Modellen führt als eher zur ablauforientierten Modellen führende Arbeitsabstimmungs-Aufgabe aus $conn_{B22}$. Während bei Concept Mapping beliebige Konzepte in Beziehung stehen können, stehen Elemente bei ablauforientierten Modellen vor allem mit ihren kausalen Vorgängern und Nachfolgern in Beziehung, was die Anzahl der Verbinder einschränkt.

Aufgrund der großen Rolle der Aufgabenstellung ist bei der Überprüfung der Hypothese wichtig, diese Störvariable möglichst auszuschalten. Zur Beurteilung wird deswegen ausschließlich der Wilcoxon-Test zwischen $conn_{B21}$ und $conn_{B22}$ herangezogen, da in diese beiden Verteilungen mit der gleichen Aufgabenstellung und identischer Stichprobe (jedoch in zeitlichem Abstand von ca. einem Monat) Zustände gekommen sind (da die Messungen unabhängig voneinander entstanden, wird ein Wilcoxon-Test für ungepaarte Variablen verwendet). Das Resultat des Wilcoxon-Tests spricht stark für die Annahme der Alternativhypothese des Tests und damit für die Annahme von Hypothese 8. Zu berücksichtigen ist hier jedoch die geringe Stichprobengröße, die die Aussagekraft des Ergebnisses wieder in Frage stellt.

Ergebnis

Die Auswertung zeigt eine signifikant höhere Verwendung von Verbindern bei Verfügbarkeit der alternativen Funktionalität zur Verbindungs-Herstellung. Auch die Natur der Aufgabenstellung scheint hohen Einfluss auf die Verwendung von Verbindern zu haben (siehe dazu auch die Diskussion von Hypothese 14 in Abschnitt XY). **Hypothese 8 kann auf Basis der vorliegenden Daten angenommen werden.**

11.3.4. Verwendung des Löschtokens

In diesem Abschnitt werden die Ergebnisse der Überprüfung der Hypothese 9 („Das Löschtönen ermöglicht intuitives Löschen von Modellelementen.“) vorgestellt.

Auswertung

Die Teilnehmer möchten einen Block umbenennen.

A: Wie haben wir jetzt gesagt (*markiert den roten Baustein*) keine Modellierungsvorgabe (*gibt Bezeichnung ein*) System übernimmt die neue Beschriftung für den Baustein nicht.

A: Wo wurde das hingeschrieben? (*Pause*) Radiergummi? Glaubst du kann man das wegradieren?

B: Probiere es aus.

A legt Radiergummi zum Block mit der Absicht die Beschriftung zu löschen

B: Nein! Du löscht alles. Hör auf!

A: Ok wie war das zuerst? Lassen wir das mal weg. (*legt Baustein zur Seite*)

A legt den Block zur Seite.

Ein ähnliches Missverständnis zeigt sich auch in folgender Situation:

TLN A und B stellen jeweils ihren Marker zu den Blöcken, die verbunden werden sollen. Dabei wird eine gerichtete Verbindung erstellt.

C: Jetzt haben wir aber einen Pfeil gebastelt.

B: Ja stimmt. Interessant.

A: Wie war das mit dem Radiergummi. (*nimmt Radiergummi und legt ihn auf die Verbindung*)

B: Nein

C: Nein, mit dem Glas! Du löscht alles!

A: Nein nur die Verbindung. (**Macht Radierbewegungen auf der Verbindung**)

C: Ich glaube dass wir das Glas nehmen müssen.

A schiebt die Blöcke zwischen denen die Verbindung gelöscht werden soll zusammen.

A: Da es funktioniert. (*schiebt die Blöcke weiter auseinander und bemerkt dass die Verbindung nicht gelöscht wurde*) Nein.

B: Ich glaube der Radiergummi vernichtet alles.

A: Nein der Radiergummi vernichtet nur Verbindungen. Nur welche? (*schiebt beide Blöcke wieder zusammen – nimmt Radiergummi weg und schiebt Blöcke in die Ausgangsposition*)

11. Evaluierung der Verwendbarkeit des Werkzeugs

Es wird eine falsche Beschriftung eingefügt. Die Teilnehmer wollen diese löschen, verwenden den Radiergummi allerdings falsch.

B: Aber irgendwie steht jetzt Ereignisse nicht bei dem Ding (*zeigt auf gelben Block*) sondern dort (*zeigt auf beschriftete Verbindung*).

A verrückt den gelben Block ein wenig.

B: Normal ist das nicht oder?

C: Nein.

A nimmt den Radiergummi.

A: Ich glaube das. (*setzt den Radiergummi auf die Arbeitsfläche*)

C: Aber nicht alles!

A nimmt Radiergummi wieder weg. System erstellt eine Verbindung zwischen zwei roten Blöcken. Teilnehmer lachen. A legt Radiergummi auf die erstellte Verbindung, und nimmt ihn wieder weg. A nimmt die beiden verbundenen Blöcke und verschiebt sie.

A: Vielleicht so. (*führt die Blöcke zusammen*)

In der Szene erstellt das System einen ungewollten Verbinde, die Teilnehmer versuchen auf verschiedene Arten den Verbinde zu löschen.

B: Und wie kann ich die Verbindungen löschen?

B: Warte einmal, da gibt es irgendwo das mit dem Radiergummi.

A: murmelt zustimmend

B nimmt den Radiergummi und platziert ihn direkt auf dem Verbinde

Das System färbt den Tisch rot

A: Nein, warte. Da löscht du Alles!

B verschiebt den Radiergummi auf dem Tisch, hebt ihn an und platziert ihn direkt auf einem Block.

Sobald der Radiergummi von der Oberfläche auf den Block gelegt wurde, entfernt das System die rote Färbung.

A: Ich glaube da löscht du Alles.

B legt den Radiergummi an mehreren Stellen trotz der Warnung von TN A auf die Oberfläche

B: Nein, es will eh nicht.

C versucht die Benennung eines Verbinders mittels Radiergummi zu entfernen.

B: Aber irgendwie steht jetzt Ereignisse nicht bei dem

Ding (*deutet auf einen Block*) sondern dort (*deutet auf einen Verinder*). Das wollen wir nicht oder?

A: Nein.

C: Ich glaube das. (*nimmt den Radiergummi und legt ihn auf den Verinder den die Teilnehmer entfernen wollen.*)

A: Aber nicht alles.

C entfernt den Radiergummi wieder von der Modellierungs-oberfläche. In diesem Moment erstellt das System automatisch einen neuen Verinder. C versucht den neuen Verinder mittels Radiergummi zu entfernen.

A: Oh Gott.

C: Vielleicht so (*schiebt die beiden betroffenen Blöcke zusammen*), nein.

B: Nein.

A: Oh Gott oh Gott oh Gott.

B: Gehen wir einen Prozessschritt zurück.

C: Genau.

Teilnehmer versuchen mit dem Radiergummi und nur einem anderen Marker einen Verinder zu entfernen.

B: Können wir die nicht so auch einfach löschen?

C: Ja mit dem Radiergummi.

B: Muss ich den jetzt zuerst so (*Hält den Radiergummi zur Kamera*) hinhalten?

A: Nein, ich glaube, **den musst du einfach da (zeigt auf den Verinder) drauf legen.**

B legt den Radiergummi auf den vom System automatisch erstellten Verinder.

A: Und jetzt muss man (*legt ein Markierungstoken auf den Verinder*) Nein.

Der Verinder lässt sich auf diese Art nicht löschen und die Teilnehmer entscheiden sich den Fehler mittels der Wiederherstellungsfunktion zu beseitigen.

Diskussion

Ergebnis

I2. Evaluierung der erstellten Modelle

I2.1. Hypothesen

I2.1.1. Konzeptuell begründete Hypothesen

Hypothese 10 *Das Werkzeug schränkt Benutzer nicht bei der Externalisierung ihrer mentalen Modelle ein.*

Offenheit, Anzahl der Element-Arten

Hypothese 11 *Das Werkzeug ermöglicht die Repräsentation beliebig komplexer Modelle.*

Größe der Oberfläche, Einbettungen

Hypothese 12 *Das Werkzeug ermöglicht die Abstimmung individuelle Modelle.*

kollaborative Modellbildung

Hypothese 13 *Die Verwendung des Werkzeugs zur kollaborativen Modellierung führt zu besseren Modellen als bei der Verwendung von bildschirm-basierten Werkzeugen.*

I2.1.2. Explorativ gebildete Hypothesen

Hypothese 14 *Zur Abbildung von Zusammenhängen ist die Verwendung von Verbindern nicht notwendig.*

Connectedness

12.2. Untersuchungsdesign und Durchführung

12.2.1. Grundlagen

12.3. Ergebnisse

12.3.1. Connectedness

I3. Evaluierung der durchgeführten Articulation Work

I3.1. Hypothesen

I3.1.1. Konzeptuell begründete Hypothesen

Hypothese 15 *Das Werkzeug verbessert den Prozess der Abstimmung zwischen Personen.*

Hypothese 16 *Die Anwendung des Werkzeugs verbessert die Ergebnisse kollaborativer Arbeit.*

I3.1.2. Explorativ gebildete Hypothesen

I3.2. Untersuchungsdesign und Durchführung

I3.3. Ergebnisse

I4. Schlussbetrachtungen

I4.1. Anwendungsszenarien

I4.1.1. Problembeschreibung und Arbeitsabstimmung

Einzel- oder Gruppensessions.

Aufgabenstellung: meist aus Arbeitsabläufen der beteiligten Personen

Merkmal: Tisch ist Mittel zum Zweck, gelegtes Modell fungiert als Diskussionsgrundlage. Modell ist statisch, wird einmal gelegt und nicht mehr verändert. Eher kompakte Modelle, die den Kontext eines Problems beschreiben. Die eigentliche Problematik ist selten explizit im Modell dargestellt.

Anwendungsbeispiele: Block 1, Block 2, Block 4 (Session 1-3).

Vorteile:

I4.1.2. Concept Mapping

Einzel- oder Gruppensessions.

Aufgabenstellung: zur Erhebung bzw. Überprüfung von domänenspezifischen (Struktur-)Wissen

Merkmal: Tisch

Anwendungsbeispiele: Block 3, Block 5

I4.1.3. Strukturaufstellung und Manipulation

Anwendungsbeispiele: Block 4 (Session 4).

Aufgabenstellung: Erhebung und Reflexion der Strukturen in denen Arbeitsabläufe situiert sind (Abteilungen, Personen, Kommunikationskanäle).

Merkmal: Modelle sind nicht statisch - werden nach der Erstellung zwar meist nicht erweitert aber in ihrer Struktur verändert (räumliche Relation der Knoten zueinander).

Anhang

Anhang A.

Literatur zum Themengebiet Articulation Work

A.I. Literaturquellen

In der Literatursuche wurden Datenbanken aus den Bereichen Informatik, Psychologie, Soziologie, den Wirtschaftswissenschaften sowie der Organisationslehre durchsucht. Nach der initialen Suche wurde jeweils auch die in den gefundenen Arbeiten referenzierte Sekundärliteratur aufgearbeitet. Des Weiteren wurden mit Hilfe von rückwärts verlinkenden Datenbanken (wo vorhanden) Publikationen erfasst, die die bislang gefundenen Arbeiten referenzieren und diese hinsichtlich ihrer Relevanz überprüft.

Die in der Suche berücksichtigten Datenbanken bzw. Meta-Suchmaschinen sind:

Domänenspezifische Datenbanken • INSPEC¹ (Naturwissenschaften)

- Business Source Premier² (Wirtschaftswissenschaften)
- PsycINFO³ (Psychologie)
- PSYNDEXplus⁴ (Psychologie)
- SocINDEX⁵ (Soziologie)
- ERIC⁶ (Pädagogik)
- ACM Digital Library⁷ (Informatik)
- IEEE XPlore⁸ (Informatik)

¹via <http://ovidsp.ovid.com>

²via <http://search.ebscohost.com/>

³via <http://ovidsp.ovid.com>

⁴via <http://ovidsp.ovid.com>

⁵via <http://search.ebscohost.com/>

⁶<http://www.eric.ed.gov/>

⁷<http://portal.acm.org/dl.cfm>

Verlags-Datenbanken • ACM Digital Library⁹ (Informatik)

- IEEE Xplore¹⁰ (Informatik)
- SpringerLink¹¹ (fächerübergreifend)
- ScienceDirect¹² (fächerübergreifend)
- Emerald¹³ (Wirtschaftswissenschaften)
- Wiley Interscience¹⁴ (fächerübergreifend)

Meta-Suchmaschinen • Google Scholar¹⁵ (fächerübergreifend)

- CiteSeerX¹⁶ (Naturwissenschaften und Informatik)

A.2. Relevante Literatur

Die im Folgenden genannten Arbeiten beziehen sich auf „Articulation Work“, treffen jedoch keine Aussage hinsichtlich einer etwaigen Unterstützung derselben. Die hier behandelten Arbeiten wurden in vier Kategorien gruppiert:

- (I) Arbeiten, die sich mit der grundlegenden Konzeption von „Articulation Work“ beschäftigen (und in den ersten Abschnitten dieses Kapitels bereits behandelt wurden) und keine Aussage zu deren Unterstützung machen.
- (II) Arbeiten, in denen „Articulation Work“ als erklärendes Rahmenwerk für beobachtete Phänomene verwendet wird, und in der Folge das Hauptaugenmerk auf diese Phänomene gelegt wird, ohne nochmals näher auf „Articulation Work“ einzugehen.
- (III) Arbeiten, die auf die Unterstützung von Articulation Work eingehen.
- (IV) Arbeiten, in denen „Articulation Work“ lediglich erwähnt wird, allerdings nicht näher darauf Bezug genommen wird.

In chronologischer Reihenfolge des Erscheinens sind die folgenden Arbeiten einer oder mehreren der genannten Kategorien zuzuordnen (Kategorie jeweils in Klammer angeführt):

⁸<http://ieeexplore.ieee.org>

⁹<http://portal.acm.org/dl.cfm>

¹⁰<http://ieeexplore.ieee.org>

¹¹<http://www.springerlink.de>

¹²<http://www.sciencedirect.com>

¹³www.emeraldinsight.com

¹⁴www3.interscience.wiley.com

¹⁵<http://scholar.google.com/>

¹⁶<http://citeseerx.ist.psu.edu/>

Strauss (1985) (I) prägt in dieser Arbeit den Begriff „Articulation Work“ und beschreibt dieses auf konzeptioneller Ebene ohne eine unmittelbaren Praxis- bzw. Umsetzungsbezug herzustellen.

Gasser (1986) (I) beschreibt die Integration von Computerunterstützung in alltägliche Arbeitsabläufe und die Anpassungsleistung der arbeitenden Individuen, wenn die aktuelle Arbeitssituation nicht mehr mit dem der Computerunterstützung zugrunde liegenden Modell übereinstimmt. Er identifiziert dabei spezifische Aktivitäten, die im Rahmen der ablaufenden „Articulation Work“ auftreten können.

Gerson und Star (1986) (I, II) zeigen die konkrete Manifestation von Articulation Work in einer Fallstudie aus einem Versicherungskonzern und identifizieren daraus die organisationalen Rahmenbedingungen, die zu jenen Problemen führen, die „Articulation Work“ notwendig machen.

Bendifallah und Scacchi (1987) (II) untersuchen bezugnehmend auf Gasser (1986) „Articulation Work“ im Kontext von IT-Support-Arbeit in Unternehmen anhand von zwei Fallstudien und identifizieren dabei zwei unterschiedliche Strategien bei der Durchführung derselben. Im Detail gehen sie jedoch nicht auf die konkret zu setzenden Maßnahmen ein.

Fujimura (1987) (I) leitet die grundlegende Unterscheidung zwischen „Production Work“ und „Articulation Work“ anhand einer Fallstudie aus dem wissenschaftlich-medizinischen Forschungsbetrieb ab. Sie bleibt dabei auf konzeptueller Ebene und beschreibt die auftretenden Phänomene, geht jedoch nicht auf unterstützende Maßnahmen ein.

Strauss (1988) (I) detailliert und erweitert seine Konzepte und setzt diese in den Kontext organisationaler Projektarbeit (im dort beschrieben Verständnis im Wesentlichen identisch mit „non-routine“, „collective activity“). Anhand einer Fallstudie aus dem Krankenhaus-Organisations-Bereich zeigt er das Auftreten in der Praxis, beschäftigt sich jedoch nicht mit möglicherweise unterstützenden Interventionen.

Schmidt (1990) (I) beschreibt ein Framework für die Analyse kooperativer Arbeit und erwähnt dabei „Articulation Work“ als ein zu berücksichtigendes Konzept. Diese Arbeit bildet die Grundlage für die im Hinblick auf die Unterstützung von „Articulation Work“ relevantere Arbeit von Schmidt und Bannon (1992).

Mi und Scacchi (1991) (III) betrachten „Articulation Work“ im Kontext der Softwareentwicklung und argumentieren für deren explizite Berücksichtigung in Software Engineering Prozessen. Sie schlagen einen formalisierten Prozess zur Durchführung von „Articulation Work“ vor und führen einen Satz von regel-

basierten Heuristiken zur konkreten Durchführung ein. Sie sind damit die ersten, die sich explizit mit der Unterstützung von „Articulation Work“ beschäftigen.

Schmidt und Bannon (1992) (I, III) begründen mit dieser Arbeit eine Entwicklungsrichtung der CSCW, die neben der Unterstützung der eigentlichen produktiven Arbeit auch auf die Unterstützung von „Articulation Work“ fokussiert. Sie beschreiben damit erstmals Anforderungen an die und Möglichkeiten der technische Unterstützung von „Articulation Work“.

Bannon und Schmidt (1993) (II) zeigen in diesem Sammelwerk die ersten Ergebnisse des COMIC-Projektes (Rodden, 1995) und erwähnen dabei in einzelnen Beiträgen „Articulation Work“ als ein im Bereich der CSCW zu berücksichtigendes Konzept.

Corbin und Strauss (1993) (I) beschäftigen sich mit der Festlegung von Interaktionsmodalitäten in kooperativer Arbeit durch „Articulation Work“ und detaillieren dabei das Verständnis von expliziter „Articulation Work“, indem sie mögliche Zeitpunkte des Auftretens sowie Schritte bei deren Durchführung nennen.

Strauss (1993) (I) fasst im Rahmen einer umfassenderen Arbeit zur Entwicklung einer „Theory of Action“ seine Überlegungen zur Rolle und Ausgestaltung von „Articulation Work“ zusammen und würdigt diese kritisch. Konkrete Maßnahmen zur Unterstützung oder Ermöglichung von Articulation Work sind aber auch hier nicht vorhanden.

(Bowers, 1994) (III) ist Editor eines COMIC-Deliverables, in dem zum ersten Mal auf die in (Schmidt und Simone, 1996) ausformulierten Anforderungen zur technischen Unterstützung von „Articulation Work“ eingegangen wird.

Lenoir (1994) (IV) erwähnt „Articulation Work“ (konkret die Arbeit von Fujimura (1987)) als Beispiel der Verknüpfung unterschiedlicher wissenschaftlicher Arbeitskontexte, geht aber dann nicht näher auf „Articulation Work“ ein.

Schmidt (1994) (III) rephrasiert im Wesentlichen (Schmidt, 1990) mit Fokus auf den Aspekt der kooperativen Arbeit (und nicht der Computerunterstützung derselben). Er detailliert darin die Artikulationsnotwendigkeiten bei kooperativer Arbeit, führt jedoch hinsichtlich der Unterstützung von Articulation Work keine zusätzlichen Anforderungen ein.

Schmidt et al. (1995) (III) basiert wie (Schmidt, 1994) auf (Schmidt, 1990), leitet jedoch inhaltlich bereits zu der oben im Detail behandelten Arbeit von Schmidt und Simone (1996) über.

Grinter (1995) (II) beschreibt die Verwendung von Konfigurations-Management-Systemen zur Koordination von Softwareentwicklungs-Prozessen. Sie bezieht

sich dabei am Rand auf „Articulation Work“ (via (Schmidt und Bannon, 1992)), führt diesen Aspekt aber nicht näher aus. Diese Arbeit bildet jedoch die Grundlage für die hinsichtlich der Unterstützung von „Articulation Work“ relevantere Arbeit derselben Autorin (Grinter, 1996).

Simone et al. (1995) (III) konkretisieren die in Schmidt und Simone (1996) beschriebene Notation zur Spezifikation von Koordinationsmechanismen in CSCW-Systemen und bereiten damit den Weg zur technischen Unterstützung von „coordinating predefined work“, die in (Divitini und Simone, 2000) umfassend beschrieben ist.

Grinter (1996) (III) betrachtet die Rolle von „Articulation Work“ im Kontext der Softwareentwicklung und zeigt anhand zweier qualitativer empirischer Studien die Auswirkungen eines computerbasierten Configuration Management Systems bei der kooperativen Erstellung von Software.

Schmidt und Simone (1996) (I, III) entwickeln in ihrer Arbeit ein generisches Vorgehen zur Konzeption von technischer Unterstützung von „Articulation Work“. Aufbauend auf früheren Arbeiten der Autoren (z.B. (Schmidt, 1990) und (Schmidt und Bannon, 1992)) formulieren die Autoren eine Spezifikationsnotation für CSCW-Systeme, die auf der Unterstützung von „Articulation Work“ aufbauen.

Bannon und Bødker (1997) (II) beschreibt die Verwendung von „Common Information Spaces“ im Kontext von CSCW und identifiziert die Artikulations-Bedürfnisse, die im Rahmen der Verwendung derselben auftreten können. Die Autoren gehen nicht näher auf die Umsetzung oder Unterstützung dieser konkreten Ausprägungen von „Articulation Work“ ein.

Fjuk et al. (1997) (I, III) versuchen die Konzepte von „Articulation Work“ durch eine Abbildung auf die Konzepte der „Activity Theory“ zu konkretisieren. Die Autoren geben dabei neben der Erweiterung der konzeptionellen Grundlagen auch mögliche Ansatzpunkte für die Unterstützung durch rechnerbasierte Werkzeuge an.

Fjuk und Dirckinck-Holmfeld (1997) (II) verwendet die Ansätze von Strauss (1993), um die Interaktion in computerbasierten kooperativen Lernumgebungen (also in CSCL¹⁷-Systemen) zu betrachten. Sie verwendet dabei „Articulation Work“ als Analysedimension (als jener Teil des Arbeitsablaufs, in dem Interaktion zwischen den Lernenden vorrangig auftritt), geht jedoch nicht näher auf deren Unterstützung ein.

(Simone und Bandini, 1997) (IV) beschreiben ein System zur Generierung von Awareness in kooperativen Anwendungen und erwähnen dabei am Rande „Ar-

¹⁷Computer Supported Cooperative Learning

ticulation Work“, als einen Aspekt, bei dessen Unterstützung das System interessant sein könnte.

Simone und Divitini (1997) (III) berichten über den aktuellen Stand der Entwicklung bei der technischen Unterstützung von Koordinationsmechanismen in CSCW-Systemen. Sämtliche hier enthaltenen Ergebnisse werden umfassender in (Divitini und Simone, 2000) dargestellt.

Kling und Star (1998) (II) beschreiben „Articulation Work“ als einen Aspekt, dessen Unterstützung bei der Gestaltung von „human centered (computer) systems“ zu berücksichtigen ist. Sie gehen jedoch nicht unmittelbar auf die mögliche Form der Unterstützung ein.

Carstensen und Schmidt (1999) (III) führen Aspekte von (Schmidt und Simone, 1996) genauer oder aus einem anderen Betrachtungswinkel aus, fügen aber dem Verständnis von „Articulation Work“ bzw. deren Unterstützung keine neuen Aspekte hinzu.

Schmidt und Simone (1999) (III) betonen basierend auf (Schmidt und Simone, 1996) den dynamischen Charakter von „Articulation Work“, die in einem Arbeitsablauf je nach Kontext unterschiedliche Ausprägungen annehmen kann. Sie fordern eine Berücksichtigung dieser Dynamik in technischen Werkzeugen zur Unterstützung von „Articulation Work“, fügen aber den Ausführungen von (Schmidt und Simone, 1996) keine fundamental neuen Anforderungen hinzu. Die Autoren leiten mit dieser Arbeit über zu der erstmals in Simone et al. (1999) vorgestellten technischen Implementierung des in den vorgegangenen Publikationen konzipierten Werkzeugs.

Simone et al. (1999) (III) stellen als Umsetzung der in (Schmidt und Simone, 1996) aufgestellten Forderungen zur Unterstützung von „Articulation Work“ durch CSCW-Systeme den „Reconciler“ vor, ein auf Java und CORBA¹⁸ basierendes Software-Modul, dass den globalen Kontext und Zustand eines (digitalen) Arbeitsobjektes bei dessen Bearbeitung durch ein Individuum offenlegt und dadurch die Entwicklung einer gemeinsamen Sicht auf geteilt benutzte Objekte ermöglicht und die Vermeidung von Konflikten unterstützt. Der „Reconciler“ ist damit ein technisches Werkzeug zur Unterstützung von „situated Articulation Work“, die Arbeit detailliert jedoch lediglich die in (Schmidt und Simone, 1996) genannten Unterstützungsaspekte um diese technisch implementierbar zu machen.

Suchman (1999) (II) beschäftigt sich mit „invisible work“ in denen Arbeitsar tefakte an die tatsächlichen Erfordernisse des jeweiligen Arbeitskontext angepasst werden („design-for-use“). Sie argumentiert für die Anerkennung (al-

¹⁸Common Object Request Broker Architecture

so Sichtbarmachung) dieser Arbeit durch die Entwicklung expliziter Design-Praktiken, geht aber nicht näher auf deren Ausgestaltung ein.

Star und Strauss (1999) (III) verfassen die einzige Arbeit, in der Strauss selbst Stellung zur Unterstützung von „Articulation Work“ im Generellen und der Unterstützung durch Computersysteme im Speziellen Stellung nimmt. Die Autoren würdigen die Argumente und Forderungen aus (Schmidt und Simone, 1996) kritisch und argumentieren gegen „Sichtbarkeit von Arbeit um jeden Preis“. „Articulation Work“ bedingt nicht notwendigerweise die vollständige Offenlegung aller Arbeitsaspekte sondern geht immer nur soweit wie für eine Wiederaufnahme bzw. Aufrechterhaltung der produktiven Arbeit notwendig. Als Konsequenz fordern sie CSCW-Systeme, die – zusätzlich zu den Anforderungen aus (Schmidt und Simone, 1996) – die Kontrolle über die Sichtbarkeit der eigenen Arbeit bei den arbeitenden Individuen belassen (und stärken damit die Anforderung, die bereits in (Schmidt und Bannon, 1992) aufgestellt wurde, in (Schmidt und Simone, 1996) jedoch nicht explizit berücksichtigt wurde).

Berg und Timmermans (2000) (II) beschreiben „Articulation Work“ im Kontext von „order and disorder“ in kooperativen Arbeitssituationen (konkret im medizinischen Sektor). „Articulation Work“ ist dabei eine Ausprägung von „disordered work“, im Sinne, dass sie nicht vorgegebenen Regeln gehorcht bzw. zur Anwendung kommt, wenn spezifizierte, routinierte Arbeit („ordered work“) nicht mehr funktioniert. Die Autoren gehen jedoch nicht auf eine mögliche Unterstützung von „Articulation Work“ oder „ordered work“ ein.

Davitini und Simone (2000) (III) stellen ein System zur Unterstützung von etablierter kooperativer Arbeit in Form eines adaptiven Workflow-Systems vor, dessen Verhalten durch die Durchführung von „Articulation Work“ beeinflusst werden kann bzw. die Durchführung derselben unterstützt.

Schmidt und Simone (2000) (III) führen im Kontext von CSCW die bereits in (Schmidt und Simone, 1996) entwickelten Konzepte nochmals weiter und zeigen dass bei Articulation Work die Grenze zwischen der Herstellung von „mutual awareness“ (als Bezeichnung einer ad-hoc durchgeführten Abstimmung) und der Verwendung „coordinative artifacts and protocols“ (als Ausprägung einer Koordination von etablierten Arbeitsprozessen) fließend ist. So fordern als Folge, dass eine technische Unterstützung beide Arten von „Articulation Work“ unterstützen muss, detaillieren oder verändern aber die konkreten Anforderungen aus (Schmidt und Simone, 1996) nicht weiter.

Simone (2000) (II) beschreibt die Rolle von „classification schemes“ für CSCW, die der Klassifikation von Domänenkonzepten zugrunde liegen. Anhand mehrerer Fallstudien beschreibt die Autorin die lokale, informelle und emergen-

te Bildung von Klassifikations-Schemata in Gruppen. Sie argumentiert letztlich dafür, dass diese Schema-Bildung Teil von „Articulation Work“ ist und unterstützt werden muss, um eventuell auftretende Inkonsistenzen zwischen den Schemata einzelner Gruppen oder Individuen zu vermeiden. Letztlich beschreibt die Autorin, dass das in (Simone et al., 1999) vorgestellte System diese Anforderung erfüllen kann.

Christensen (2001) (II) beschäftigt sich mit „Articulation Work“ in Arbeitssituationen, in die mobil arbeitende Individuen involviert sind und konzentriert sich auf jene Arbeits-Aspekte, die spezifisch für derartige Situationen zusätzlich zu artikulieren sind. Er identifiziert diese Aspekte im Rahmen einer Studie und beschreibt ausschließlich den Status quo ohne konkrete Unterstützungsmaßnahmen anzuführen. Weiterführende Arbeiten zu diesem Ansatz sind nicht publiziert.

Fuchs et al. (2001) beschreiben die technische Unterstützung von „Articulation Work“ in (verteilten) Gruppen mittels CSCW-Technologie. Die Autoren präsentieren ein konkret umgesetztes System, das eine Reihe von Werkzeugen zur Unterstützung von „Articulation Work“ bietet.

Raposo et al. (2001) (III) stellen ein konzeptionelles Framework vor, das die Koordination von voneinander abhängigen Aufgaben in Gruppen erlauben soll und damit „Articulation Work“ mit dem Ziel „coordination of predefined work“ unterstützen soll. Dabei schlagen die Autoren eine Struktur vor, die es erlaubt, für eine Abhängigkeit zwischen Aufgaben unterschiedliche Koordinationsstrategien festzulegen, die dann kontextabhängig ausgewählt werden können. Das Framework wird in (Raposo und Hugo, 2002) weiter konkretisiert. In (Raposo und Hugo, 2002) wird dessen Umsetzung in einem technischen System beschrieben.

Simone und Sarini (2001) (II, III) entwickeln die Ansätze hinsichtlich der Unterstützung der Bildung von „classification schemes“ aus (Simone, 2000) weiter und konzentrieren sich dabei auf deren Adaptierung an konkrete Arbeitssituationen. Sie führen dabei aber keine neuen Aspekte hinsichtlich der Unterstützung von „Articulation Work“ ein.

Bosse (2002) (II) baut auf der Arbeit von (Bannon und Bødker, 1997) zu „Common Information Spaces“ auf und identifiziert im Rahmen einer Fallstudie im medizinischen Bereich Gestaltungsparameter, in deren Rahmen auch „Articulation Work“ als in unterschiedlichen Ausprägungen zu unterstützendes Phänomen genannt wird, ohne näher auf die Implikationen dieser Forderung einzugehen.

Davenport (2002) (II, III) beschreibt „Articulation Work“ als eine Form von „alltäglichem Wissensmanagement“, mit Hilfe dessen beteiligte Individuen im

Arbeitsprozess lernen und ihre Kompetenzen erweitern („situated learning“). Anhand einer Fallstudie zeigt sie, dass das Konzept der „Communities of Practice“ (Wenger, 1998) und deren Methoden geeignet sind, diese Form von „Articulation Work“ zu unterstützen. Die Autorin deutet die Möglichkeit einer Unterstützung durch rechnerbasierte Werkzeuge an, führt diese Idee aber nur am Rande aus.

Herrmann et al. (2002) (II, III) beschäftigen sich mit Modellen von soziotechnischen Arbeitsprozessen und zeigen auf, dass zu deren (kooperativen Erstellung) „Articulation Work“ notwendig ist.

Mark et al. (2002b) (II) stellen eine Kurzfassung des in (Mark et al., 2002a) ausführlich beschriebenen Tests des „Reconciler“-Systems vor.

Mark et al. (2002a) (II) beschreiben einen ersten Test des „Reconciler“-Systems und zeigen, dass das Werkzeug tatsächlich bei der Entwicklung eines gemeinsamen Sichtweise über die Arbeitsdomäne betragen kann.

Raposo und Hugo (2002) beschäftigen sich aufbauend auf (Raposo et al., 2001) mit der Konkretisierung des Frameworks zur Unterstützung der Koordination von Aufgaben, die in gegenseitiger Abhängigkeit stehen. Die Autoren bereiten damit das Feld für die technische Umsetzung des Frameworks, die in (Raposo et al., 2004) beschrieben wird.

Sarini und Simone (2002a) (III) beschäftigen sich mit „recursive Articulation Work“, also jener Form, deren Gegenstand selbst wiederum „Articulation Work“ ist. Die Autoren leiten Anforderungen an die Unterstützung dieser Form von „Articulation Work“ ab und zeigen die konkrete Umsetzung als Teil des „Reconciler“-Systems.

Sarini und Simone (2002b) beschreiben in Form einer Kurzfassung die wesentlichen Konzepte und Implementierungsansätze des „Reconciler“-Systems.

Schmidt (2002) (IV) beschäftigt sich konzeptionell mit der Unterstützung von Awareness in CSCW-Systemen und erwähnt dabei am Rande, dass Awareness oft ein wichtiger Aspekt von „Articulation Work“ ist.

Simone (2002) (IV) beschreibt die im Rahmen des „Reconciler“-Projektes durchgeführte Arbeit im Kontext von Wissensmanagement und „Organizational Memories“¹⁹. Sie zeigt, in welchen Aspekten Berührungspunkte zwischen Wissensmanagement und CSCW bestehen und weist auf mögliche Unterstützungsleistungen hin. Auf „Articulation Work“ wird nur im Zusammenhang mit dem im Wissensmanagement relevanten Abgleich von Ontologien verwiesen, der als „Articulation Work“ gesehen werden kann.

¹⁹für einen Überblick zu diesem Themengebiet siehe (Maier, 2008)

Eschenfelder (2003) (II) beschreibt eine qualitative Studie über das Management von content-zentrierten Websites und zieht „Articulation Work“ (wie in (Corbin und Strauss, 1993) beschrieben) als das der Analyse zugrundeliegende Framework heran. Die Autorin zeigt im zweiten Teil der Arbeit auf, wie Content Management Systeme den Verwaltungsprozess unterstützen können, geht aber nicht weiter auf „Articulation Work“ ein.

Olesen und Markusen (2003) (IV) beschreiben die Veränderung des Arbeitsablaufs der Rezeptausstellung in einem Krankenhaus durch Einführung eines technischen Systems, dass die elektronische Verschreibung von Medikamenten erlaubt. Die Autoren verfolgen dabei einen kulturwissenschaftlichen Ansatz und weisen lediglich in der Einleitung auf „Articulation Work“ als eine bei der Umstellung des Arbeitsablaufs notwendige Tätigkeit hin.

Sarini (2003) (III) fasst die konzeptuellen Grundlagen, die Implementierung und den Test des „Reconciler“-Systems in Form seiner Dissertation zusammen. Er führt dabei jedoch keine nicht bereits in früheren Publikationen veröffentlichten Argumente oder Anforderungen ein.

Gerson (2004) (III) beschreibt die Verwendung von „Reconciliation Mechanisms“ zur Auflösung von Problemen in der Zusammenarbeit bei räumlich verteilt durchgeföhrter Arbeit. Diese „Reconciliation Mechanisms“ sind vorrangig organisationale oder soziale Maßnahmen, die die Zusammenarbeit verbessern bzw. wieder möglich machen. Ein expliziter Bezug zu „Articulation Work“ wird nicht hergestellt, ausgehend von der Beschreibung scheinen „Reconciliation Mechanisms“ aber ein Mittel zur Durchführung von „Articulation Work“ zu sein. Gerson gibt exemplarisch vier dieser Mechanismen an (z.B. „shared resource pools“ oder „participant review“), ohne jedoch deren detaillierte Ausgestaltung einzugehen.

Jørgensen (2004) (III) beschreibt die Verwendung von „interaktiven“ Prozessmodellen in organisationalen Arbeitsprozessen und die Veränderung dieser Prozesse durch Modellierungsvorgänge. Dabei bezeichnet er den Modellierungsvorgang als „Articulation Work“. „Interaktive“ Prozesse sind dabei solche, die wissensintensiv sind, im Vorhinein spezifiziert werden können und deren konkreter Ablauf erst zum Zeitpunkt der Ausführung festgelegt wird (was jenen Arbeitsabläufen entspricht, die als „problematic“ oder „non-routine“ bezeichnet werden). Der Autor entwickelt im Rahmen der Arbeit eine Methodik zur Modellierung derartiger Prozesse und ein technisches Werkzeug, dass die Erstellung und Instanzierung dieser Modelle ermöglicht unterstützt bzw. ermöglicht.

Raposo et al. (2004) decken in ihrer Arbeit zur (technischen) Unterstützung kooperativer Arbeit explizit alle Zeitpunkte, in denen „Articulation Work“ auf-

treten kann, ab („pre-articulation“, „coordination“, „post-articulation“). Sie schlagen zur Koordination formalisiert festgeschriebene „Commitments“ vor, die in der „pre-articulation“-Phase definiert werden und während der „post-articulation“ evaluiert werden. Damit decken die Autoren auch „recursive Articulation Work“ Sarini und Simone (2002a) ab. In der Arbeit wird im wesentlichen der vorgeschlagene Formalismus und dessen konzeptionelle Anwendung dargestellt.

Færgemann et al. (2005) (I, II) beschreiben „Articulation Work“ in Arbeitsprozessen, die unterschiedliche große Personenkreise umfassen, die verschieden stark miteinander vertraut sind. Die Autoren leiten auf ihren empirischen Beobachtungen vier unterschiedliche Arten von „Articulation Work“ ab, die sich jeweils in der Größe ihres Durchführungskontexts (d.h. des Teilnehmerkreises) unterscheiden. Sie beschreiben die Charakteristika dieser Arten von „Articulation Work“, gehen aber nicht auf deren Unterstützung ein (wobei sie andeuten, dass eine technische Unterstützung jeweils unterschiedlich ausfallen muss, bezeichnen dies jedoch als eine offene Forschungsfrage).

Hasu (2005) (II, IV) beschreibt die Einbindung neuer (computer-basierter) Werkzeuge in Arbeitsabläufe durch technische Laien (konkret die Verwendung eines neuen, komplexen medizinischen Gerätes durch Neurologen). Sie klassifiziert die im Zuge dessen anfallenden Aktivitäten als „invisible Articulation Work“. Sie geht im Übrigen darauf ein, wie derartige Prozesse durch ethnographische Forschung erfasst werden können, die Unterstützung von „Articulation Work“ selbst wird aber nicht weiter thematisiert.

Hampson und Junor (2005) (II) beschreiben die Arbeit im interaktiven (d.h. hier telemediengestützten) Kundenservice als „Articulation Work“. Die Autoren führen eine Klassifikation von unterschiedlichen Arten von Arbeitsabläufen ein, um ihren Fokus abzugrenzen. In der Folge beschreiben sie die im Rahmen des „interactive customer service“ auftretende Phänomene, deren konkrete Ausprägungen und die Reaktionen der Kundenbetreuer. Sie legen dar, welche Rolle „Articulation Work“ in diesem Kontext spielt, gehen dabei aber nicht darauf ein, wie diese Abläufe unterstützt werden können.

Cabitza et al. (2006) (III) entwickeln den „Reconciler“-Ansatz weiter und wenden ihn unter Bezugnahme auf Færgemann et al. (2005) auf „globale Articulation Work“ an. Sie entwickeln dabei ein konzeptionelles Framework, das (wie im Falle des „Reconciler“-Ansatzes) auf Artefakten als Artikulations-Objekten beruht und geben eine Methodik an, wie derartige Artefakte entwickelt werden können (im Sinne der „recursive Articulation Work“). Die vollständige Umsetzung sowie eine Evaluierung des Konzepts stand zum Zeitpunkt der Publikation der Arbeit noch aus.

Crabtree et al. (2006) (II, III) zeigen die Relevanz von „Articulation Work“ in Situationen, in denen Personen einander Hilfestellungen geben. Die Autoren beschreiben dabei Situationen, in denen die Hilfestellung „remote“ (d.h. aus der Entfernung) erfolgt. Sie zeigen, welche Artikulationsprozesse dabei regelmäßig auftreten und leiten Anforderungen an eine mögliche Unterstützung für derartige Arbeitsprozesse ab. Obwohl grundsätzlich relevant für die Unterstützung von „Articulation Work“, bleibt diese Arbeit hinsichtlich der Anforderungen jedoch relativ abstrakt und unspezifisch.

Kaghan und Lounsbury (2006) (II, III) (bzw. der ebenfalls vorliegende ausführlichere Preprint Kaghan und Lounsbury (2004)) zeigen mit kulturwissenschaftlichem Hintergrund, wie organisationale Artefakte (also Ergebnisse bzw. Gegenstände von Arbeit) kooperativ erstellt, verwendet und angepasst werden und in der Folge die mit ihnen verbundene Arbeit wiederspiegeln. Die Autoren bedienen sich dabei einer Fallstudie aus dem Bereich des Technologietransfers zwischen Universitäten und Wirtschaft, wo Verträge als Artefakte bzw. die Vertragsverhandlung als relevanter Arbeitsablauf im Detail betrachtet werden. „Articulation Work“ kommt dabei im Rahmen der Vertragsanbahnung („arranging deals“) zum Einsatz. Allgemein hat „Articulation Work“ hier das Ziel ein erreichtes gemeinsames Verständnis so in einem Artefakt abzubilden, dass dieses von den Beteiligten als Repräsentant der vereinbarten Zusammenarbeit akzeptiert wird. Die Autoren nehmen wie Davenport (2002) Bezug auf „Communities of Practice“ als wesentliches Konzept bei der Entwicklung dieser Artefakte.

Baker und Millerand (2007) (I, II) beschreiben, wie „Articulation Work“ im Rahmen des Designs von „information infrastructure“ (als Bezeichnung von Systemen, die die Verwaltung und strukturierte Manipulation von Information erlauben) zur Anwendung kommt. Die Autoren beziehen sich auf eine von ihnen durchgeführte empirische Studie und fassen aufgrund ihrer Erkenntnisse den Begriff „Articulation Work“ so breit, dass er nicht nur die Abstimmung der eigentlichen Arbeitsabläufe umfasst sondern etwa auch die Aushandlung eines gemeinsamen Verständnisses über den Aufbau der Arbeitsdomäne.

Cabitza und Sarini (2007) (IV) beschreiben die Verwendung von „dokumentatischen Artefakten“ und deren Computer-Unterstützung im medizinischen Bereich. Die Autoren gehen dabei nur in einem Nebensatz explizit auf „Articulation Work“ ein, die Arbeit dient aber gemeinsam mit Cabitza et al. (2006) als Grundlage der weiteren Entwicklungen zur Unterstützung von „Articulation Work“, die in (Cabitza und Simone, 2009a) beschrieben wird.

Convertino et al. (2008) (IV) beschreiben, wie in kooperativen Arbeitsprozessen ein gemeinsames Verständnis der Arbeitsdomäne („common ground“) ent-

wickelt werden kann und in weiterer Folge eine einfachere Zusammenarbeit zwischen den beteiligten Individuen ermöglicht. Im Rahmen der in der Arbeit beschriebenen empirischen Studie beziehen sich die Autoren aber nur am Rande auf „Articulation Work“.

Larsen und Bardram (2008) (II) beschreiben, wie in kooperativen Arbeitsprozessen Information über die Kompetenzen und Verantwortlichkeiten der beteiligten Individuen ausgetauscht wird. Aus der vorgestellten empirischen Studie leiten die Autoren ab, dass – trotz fortgeschritten er technischer Möglichkeiten – die Abstimmung von Kompetenzen und Verantwortlichkeiten in kooperativer Arbeit in synchronen zu einem besseren Ergebnis führt als in asynchronen Settings.

Cabitza et al. (2008) (IV) betrachtet die Ausführungen aus Cabitza und Sarini (2007) aus Perspektive des Wissensmanagement und bildet damit ebenso die Grundlage für die in (Cabitza und Simone, 2009a) vorgestellte technische Lösung vor. „Articulation Work“ als Konzept wird hier nicht explizit angesprochen.

Cabitza und Simone (2009a) (III) schlagen „active artifacts“ als Mittel zur Unterstützung von „Articulation Work“ zwischen „Communities“ im Arbeitsablauf (im Sinne von „coordinating predefined work“) vor. „Active artifacts“ können dabei nicht nur Information tragen, sondern auch auf ihren aktuellen Kontext reagieren und selbstständig aktiv Information vermitteln. Dabei führen die Autoren auch ein Konzept an, wie das Verhalten derartiger „active artifacts“ spezifiziert werden können. Hinsichtlich der Unterstützung von „Articulation Work“ ist die Arbeit als technische Detaillierung und Verfeinerung der in Cabitza et al. (2006) bereits vorgestellten Konzepte zu sehen.

Cabitza und Simone (2009b) (III) stellen die in Cabitza et al. (2006) vorgeschlagene und in (Cabitza und Simone, 2009a) eingesetzte Sprache zur Spezifikation von Koordinations-Artefakten in „global Articulation Work“ im Detail vor. Diese basiert im Wesentlichen auf der Formulierung von ECA-Regeln, die im operativen Betrieb die Grundlage der Koordinations-Unterstützung bilden.

Cabitza et al. (2009) (II) stellen eine empirische Studie zur Motivation des in (Cabitza und Simone, 2009a) vorgestellten Systems vor und zeigen dessen unterstützende Wirkung bei der Durchführung von „Articulation Work“.

Betrachtet man diese Arbeiten in ihrer Gesamtheit, so zeigt sich die historische Entwicklung der Forschung zum Thema „Articulation Work“ oder unter Verwendung derselben. Vor allem wird ein starker Bezug zur Konzeption von CSCW-Systemen sichtbar, in deren Kontext ein Großteil der verfügbaren Arbeiten verfasst wurden. Zudem sind auch Gruppen von Publikationen zu erkennen, die im gleichen Kontext

publiziert wurden sich nur in Einzelaspekten unterscheiden. Abbildung A.1 auf Seite XVII zeigt diese Zusammenhänge.

Beginnend mit den Arbeiten von Strauss in der linken oberen Ecke ist vertikal die zeitliche Dimension der Publikation von Arbeiten zu Artikulation Work aufgetragen. Die Seitenbreite wird zur thematischen Gruppierung der Publikationen verwendet. Die Pfeile zwischen Publikationen bzw. Publikationsgruppen stellen einen inhaltlichen Bezug dar. Die Publikationen am Endpunkt des Pfeils nehmen dabei Bezug auf jene, die sich am Ausgangspunkt des Pfeils befinden.

Am linken Rand der Darstellung sind die Arbeiten zu finden, die im soziologischen Kontext verfasst wurden. Die meisten der dort angesiedelten Publikationen sind Grundlagenarbeiten, die den Begriff „Articulation Work“ und dessen konzeptionellen Kontext erörtern oder anhand von Fallstudien das Auftreten von „Articulation Work“ zeigen.

Im Zentrum der Darstellung steht die größte Gruppe von Arbeiten, die im Kontext von CSCW verfasst wurde. Die Arbeiten, die sich auf CSCW beziehen, haben dabei zum Teil die Ableitung für Anforderungen an eine technische Unterstützung von „Articulation Work“ zum Ziel, der Rest der Arbeiten beschäftigt sich eher mit der technischen Umsetzung der Unterstützung. Jene Publikationen, die eher ersterer Gruppe zuzuordnen sind, sind eher links angeordnet, die technisch orientierten Publikationen befinden sich eher rechts. Die Entfernung zur Mittelachse hat dabei keine Aussagekraft sondern ist nur einer übersichtlichen Anordnung geschuldet. Innerhalb der CSCW-Gruppe gibt es zwei bedeutende Sub-Gruppen, die untereinander in Beziehung stehen. Einerseits ist die Gruppe von Publikationen zu nennen, die im Rahmen des COMIC-Projektes 1992-1995 entstanden sind (Rodden, 1995). In diesem Projekt wurde die Grundlage der Berücksichtigung von „Articulation Work“ als Thema von CSCW gelegt. Bereits im Rahmen des COMIC-Projektes beginnend, publiziert die Gruppe um Simone Arbeiten zur konkreten technischen Umsetzung der Unterstützung durch computerbasierte Werkzeuge. Die Implementierungen, auf die dabei immer wieder Bezug genommen wird, werden als „Ariadne“ (für den Koordinierungsaspekt von „Articulation Work“) „Reconciler“ (für den Awareness-Aspekt von „Articulation Work“) bezeichnet, was auch als Namensgeber dieser Gruppe von Arbeiten herangezogen wurde.

Weiter rechts am oberen Rand der Abbildung befinden sich Arbeiten, die Articulation Work im Kontext der Software-Entwicklung betrachten. Dies sind die ersten Arbeiten, die eine konkrete Anwendung der Konzepte um „Articulation Work“ außerhalb der Soziologie bzw. der Community um Strauss zeigen. Die Unterstützung von Articulation Work ist hier nur teilweise Gegenstand der Betrachtung, wo sie aber angesprochen wird ist sie entsprechend der Anwendungsdomäne eher technisch orientiert.

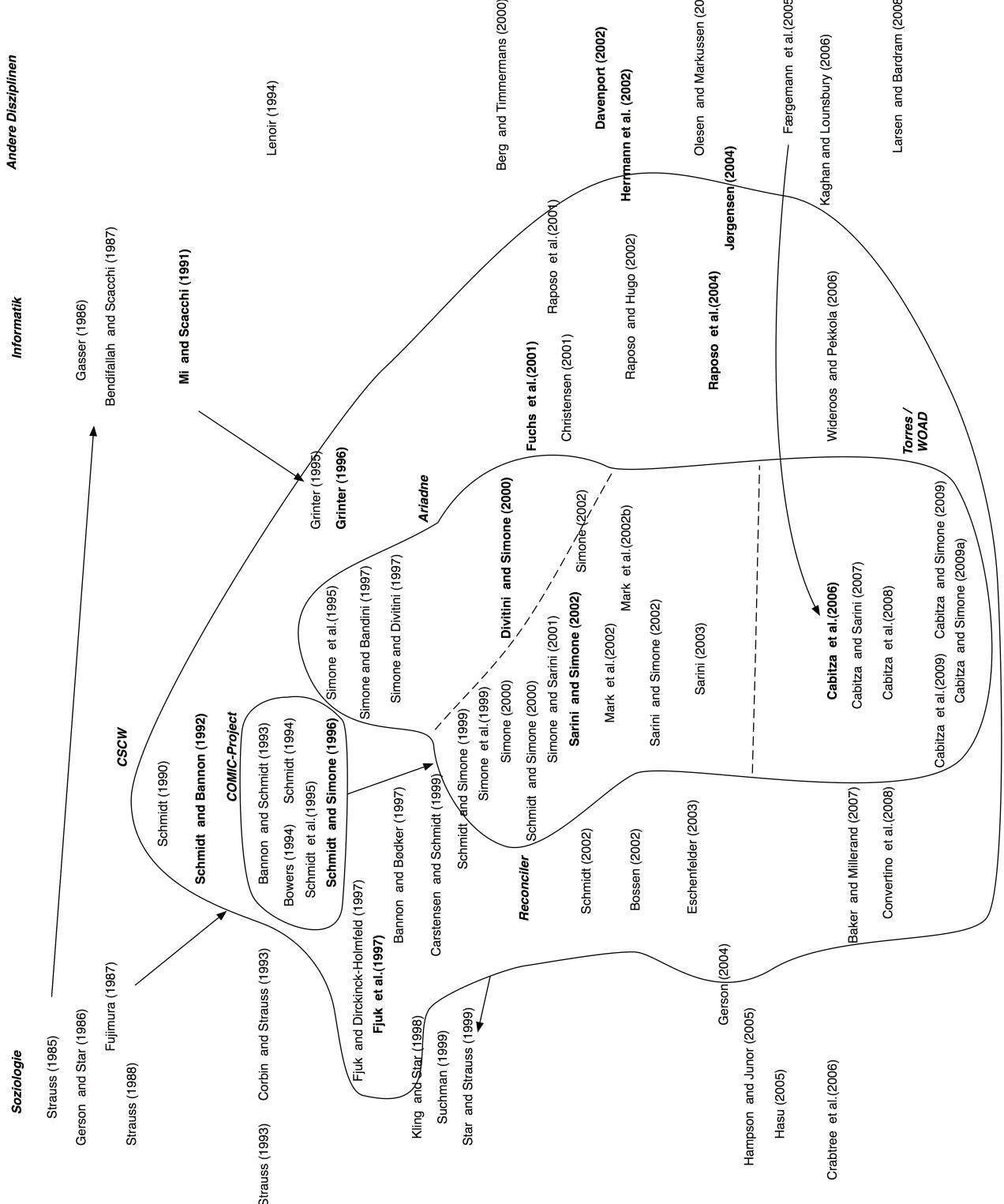


Abbildung A.1.: Literatur zu Articulation Work im Kontext

Ganz rechts sind jene Arbeiten zu finden, die auf „Articulation Work“ Bezug nehmen, jedoch nicht einer der bisher beschriebenen Gruppen zuzuordnen sind. Hier finden sich Publikationen die vor philosophischem, organisationswissenschaftlichem Hintergrund oder mit Bezug zum Wissensmanagement verfasst wurden. Herauszugreifen ist hier die Arbeit von Jørgensen (2004), der die Rolle von Modellen (konkret konzeptionellen Modellen von Arbeit) bei der Durchführung von „Articulation Work“ betrachtet und damit erstmals einen konkreten Unterstützungsbezug zwischen „Artikulation Work“ und der Domäne der Organisationswissenschaften herstellt (was wiederum für die Betrachtung von „Articulation Work“ im organisationalen Kontext von Interesse ist).

Insgesamt ist in der Abbildung ein starker Schwerpunkt auf die technische Unterstützung von Arbeit, konkret „Articulation Work“, zu erkennen. Dieser Schwerpunkt wurde sowohl konzeptionell als auch technisch ab Beginn der 90er-Jahre des 20. Jahrhunderts bis etwa 2005 ausführlich bearbeitet. In den letzten Jahren treten verstärkt Fallstudien auf, die einen Bezug zu „Articulation Work“ herstellen, jedoch nur bedingt auf deren Unterstützung eignen.

Verzeichnisse

Abbildungsverzeichnis

2.1.	Struktur von Arbeitsabläufen	11
2.2.	Konzeptualisierung von „Arbeit“ nach (Strauss, 1985) und (Fujimura, 1987)	12
2.3.	Articulation Work im Durchführungskontext	26
2.4.	Abzustimmende Arbeitsaspekte	30
2.5.	Zusammenhänge zwischen Arbeitsaspekten	31
2.6.	Artikulations-Prozess	34
2.7.	Mentale Modelle im Kontext der Arbeitsmodellierung	51
3.1.	Schemata und mentale Modelle	66
3.2.	Externalisierung mentaler Modelle	71
3.3.	Struktur einer Concept Map	81
3.4.	Externalisierung mentaler Modelle mittels Strukturlegetechniken und Concept Mapping	85
5.1.	Bedeutung von Objekten in TUIs	105
5.2.	Interaktionsmodelle für GUI und TUI	107
5.3.	Arten von Tangible User Interfaces	109
5.4.	Überblick über das MCRit-Modell	120
6.1.	Architektur des TUIpist-Framework	143
6.2.	Zusammenspiel der Komponenten in TUIpist	144
6.3.	AR Toolkit Marker	145
6.4.	Visual Codes – Aufbau und Features	146
6.5.	ReacTIVision Code	147
6.6.	Überblick über den Aufbau des Werkzeugs – Eingabekomponenten .	155
6.7.	An Tokens angebrachte ReacTIVision-Codes zur Identifikation . .	157
6.8.	Arten von Modellierungstokens	157
6.9.	Rückwand von Container Tokens	158
6.10.	Geöffnetes Container Token	159
6.11.	Modellelemente – Taxonomie	160
6.12.	Einbettbare Tokens	161
6.13.	Markierung-Token	162

6.14. Lösch-Token	163
6.15. Registrierungstoken	164
6.16. Snapshot-Token	165
6.17. History-Token	165
6.18. Softwarearchitektur zur Erkennung von Benutzerinteraktion	178
7.1. Überblick über den Aufbau des Werkzeugs – Ausgabekomponenten	206
7.2. Softwarearchitektur zur Verwaltung der Ausgabekanäle	213
7.3. Darstellung von Modellementen	214
7.4. Darstellung von Verbindern	215
7.5. Darstellung gerichteter Verbinder	216
7.6. Darstellung von Containern und eingebetteten Elementen	217
7.7. Markierung von Modellementen	218
7.8. Darstellung der Modellierungshistorie	220
7.9. Unterstützung der Wiederherstellung von Modellzuständen	222
7.10. Zusammenhänge der Klassen zur Ausgabebehandlung	224
8.1. Grundlegende Elemente einer Topic Map	232
8.2. Umfassende Darstellung der Elemente einer Topic Map	234
8.3. Abgrenzung zwischen Subject und Occurrence in Topic Maps	235
8.4. Benennung von Topics	236
8.5. Beziehungen in der Metamodellbildung in Topic Maps	239
8.6. Abbildung von Gültigkeitsbereichen durch Scopes	242
8.7. Abbildung von Modellinformation in Topic Maps	244
8.8. Definition des Meta-Models (ohne Kardinalitäten)	247
8.9. Einbindung des Meta-Meta-Modells	249
8.10. Ausschnitt einer mittels GraphViz visualisierten Topic Map	253
8.11. Modellierungshistorie als exportierte Grafik	255
8.12. Modell-Hierarchie als exportierte Grafik	257
II.1. Dauer der Werkzeugverwendung – Überblick	317
II.2. Dauer der Werkzeugverwendung – Concept Mapping	318
II.3. Dauer der Werkzeugverwendung – Aushandlung	318
II.4. Zeitverteilung zwischen den Teilnehmern	320
II.5. Connectedness in Evaluierungsblock 2 - Durchgang 1	321
II.6. Connectedness in Evaluierungsblock 2 - Durchgang 2	322
II.7. Connctedness in Evaluierungsblock 3	322
A.1. Literatur zu Articulation Work im Kontext	XVII

Tabellenverzeichnis

5.1.	Kategorien von konzeptionellen Arbeiten im Gebiet Tangible User Interfaces	124
5.2.	Gegenüberstellung der Nomenklatur zur Beschreibung der Elemente eines TUI	126
6.1.	Gegenüberstellung der Frameworks für video-basierten Input	151
6.2.	Gegenüberstellung der generischen Frameworks	153
9.1.	Beurteilung des Werkzeugs hinsichtlich des Degree of Coherence . .	276
9.2.	Spezifikation des Werkzeug mittels TAC-Schema	278
9.3.	Einordnung des Systems in die Taxonomie nach Fishkin	282
10.1.	Ursprüngliches globales Untersuchungsdesign	303
10.2.	Einfluss der Untersuchungen auf die zu evaluierenden Aspekte	303

Abkürzungsverzeichnis

API	Application Interface
AWT	Abstract Window Toolkit
CORBA	Common Object Request Broker Architecture
CSCL	Computer Supported Cooperative Learning
CSCW	Computer Supported Cooperative Work
EAN	European Article Number
ECA	Event-Condition-Action
EMF	Eclipse Modeling Framework
EPK	Ereignisgesteuerte Prozesskette
GEF	Graphical Editing Framework (Eclipse-Komponente)
GIF	Graphics Interchange Format
GMF	Graphical Modeling Framework (Eclipse-Komponente)
GPS	Global Positioning System
GUI	Graphical User Interface
HSLT	Heidelberger Strukturlegetechnik
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
JPEG	Joint Photographic Experts Group (File Interchange Format)
JRE	Java Runtime Environment
LCD	Liquid Crystal Display (Flüssigkristallanzeige)
LED	Light Emitting Diode (Leuchtdiode)
MaNET	Mannheimer Netzwerk-Elaborations-Technik

Acronyms

MCRit	Model-Control-Representation (intangible and tangible)
MCRpd	Model-Control-Representation (physical and digital)
MVC	Model-View-Controller
OCR	Optical Character Recognition
OLED	Organic Light Emitting Diode
PNG	Portable Network Graphics
RDBMS	Relationales Datenbank Management System
RFID	Radio Frequency Identification
RMI	Remote Methode Invocation
TAC	Token and Constraint
TCP	Transport Control Protocol
TUI	Tangible User Interface
UML	Unified Modelling Language
WfMS	Workflow-Management-System
XML	Extensible Markup Language
XTM	XML Topic Map

Abbildungsquellen

Dieser Anhang enthält Quellenangaben für alle in dieser Arbeit verwendeten Abbildungen, sofern sie nicht vom Autor erstellt wurden. Sofern nicht anders angegeben sind alle Abbildungen und Fotos Werke des Autors und dürfen nicht ohne ausdrückliche Zustimmung verwendet werden.

Abbildung 5.2 Bild des MVC- und MCRpd-Modells entnommen aus (Ullmer und Ishii, 2000)

Abbildung 6.3 Bild des AR Toolkit Markers entnommen von www.hitl.washington.edu/artoolkit/ (Website der Entwickler)

Abbildung 6.4 Bild des Visual Code Markers entnommen aus (Rohs und Gfeller, 2004)

Abbildung 8.3 Foto der Tasse lizenfrei unter <http://www.oldskoolman.de/bilder/freigestellte-bilder/essen-trinken/kaffee-tasse-freigestellt/>, übrige Abbildung eigene Darstellung

Grafiken aus ImplementierungInput fehlen noch

Publikationen im Kontext dieser Arbeit

Oppl und Stary (2005) description

Oppl (2006) description

Oppl et al. (2006) description

Oppl (2007b) description

Oppl (2007a) description

Oppl und Peherstorfer (2007) description

Furtmüller und Oppl (2007) description

Oppl (2008b) description

Oppl (2008a) description

Oppl und Stary (2009) description

Oppl (2009d) description

Oppl (2009c) description

Oppl (2009a) description

Oppl (2009b) description

Literaturverzeichnis

- Abecker, A., Bernardi, A., Hinkelmann, K., Kühn, O., und Sintek, M. (1998). Toward a Technology for Organizational Memories. *IEEE Intelligent Systems*, 13(3):40–48. Referenziert auf S. 69
- Allied Vision Technologies GmbH (2008). AVT Guppy. Technical Manual V6.2.0, Allied Vision Technologies GmbH, Stadtroda, Germany. Referenziert auf S. 166
- Argyris, C. und Schön, D. (1978). *Organizational Learning: A Theory Of Action Perspective*. Addison-Wesley. Referenziert auf S. 1, 2, 24, 65
- Arnold, K., Scheifler, R., Waldo, J., O'Sullivan, B., und Wollrath, A. (1999). *Jini Specification*. Addison-Wesley Longman Publishing, Boston, MA, USA. Referenziert auf S. 144, 152
- Azuma, R. (1997). A survey of augmented reality. *Presence-Teleoperators and Virtual Environments*, 6(4):355–385. Referenziert auf S.
- Baker, K. und Millerand, F. (2007). Articulation work supporting information infrastructure design: Coordination, categorization, and assessment in practice. *Proceedings of HICSS 2007*, Seite 242a. Referenziert auf S. 3, 14, XIV
- Bannon, L. und Bødker, S. (1997). Constructing common information spaces. In *Proceedings of the fifth conference on European Conference on Computer-Supported Cooperative Work*, Seiten 81–96. Kluwer Academic Publishers Norwell, MA, USA. Referenziert auf S. VII, X
- Bannon, L. und Schmidt, K. (1993). Issues of Supporting Organizational Context in CSCW Systems. Deliverable D1.1, The COMIC Project (Esprit Basic Research Action 6225). Referenziert auf S. VI
- Becker, J., Rosemann, M., und von Uthmann, C. (2000). Guidelines of business process modeling. In van der Aalst, W., Sedel, J., und Oberweis, A., editors, *Business Process Management: Models, Techniques, and Empirical Studies*, number 1806 in LNCS, Seiten 241–262. Springer. Referenziert auf S. 289

- Bederson, B., Grosjean, J., und Meyer, J. (2004). Toolkit design for interactive structured graphics. *IEEE Transactions on Software Engineering*, 30(8):535–546. Referenziert auf S.
- Beer, W., Christian, V., Ferscha, A., und Mehrmann, L. (2003). Modeling Context-aware Behavior by Interpreted ECA Rules. In *Proceedings of the International Conference on Parallel and Distributed Computing (EUROPAR '03)*, volume 2790 of LNCS, Seiten 1064–1073. Springer. Referenziert auf S. 141
- Bellotti, V., Back, M., Edwards, W., Grinter, R., Henderson, A., und Lopes, C. (2002). Making sense of sensing systems: five questions for designers and researchers. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves*, Seiten 415–422. ACM New York, NY, USA. Referenziert auf S. 110, 274, 275
- Bendifallah, S. und Scacchi, W. (1987). Understanding software maintenance work. *IEEE Transactions on Software Engineering*, 13(3):311–323. Referenziert auf S. 14, 38, V
- Berg, M. und Timmermans, S. (2000). Order and their others: On the constitution of universalities in medical work. *Configurations*, 8(1):31–61. Referenziert auf S. IX
- Blackwell, A., Morrison, C., und Edge, D. (2007). A solid diagram metaphor for tangible interaction. In *CHI '07 extended abstracts on Human factors in computing systems*, New York, NY, USA. ACM. Referenziert auf S. 129
- Bloks, R. H. J. (1996). The IEEE-1394 high speed serial bus. *Philips Journal of Research*, 50(1-2):209–216. Referenziert auf S. 166
- Bluetooth SIG (2007). Bluetooth Specification Version 2.1 + EDR. Specification, Bluetooth SIG. Referenziert auf S. 138
- Bossen, C. (2002). The parameters of common information spaces: the heterogeneity of cooperative work at a hospital ward. In *Proceedings of the 2002 ACM conference on Computer supported cooperative work*, Seiten 176–185. ACM Press New York, NY, USA. Referenziert auf S. X
- Bowers, J. (1994). A Conceptual Framework for Describing Organizations. Deliverable DI.2, The COMIC Project (Esprit Basic Research Action 6225). Referenziert auf S. VI
- Brant, J. M. (1995). HotDraw. Master's thesis, University of Illinois at Urbana Champaign. Referenziert auf S. 208
- Budinsky, F., Brodsky, S., und Merks, E. (2003). *Eclipse modeling framework*. Pearson Education. Referenziert auf S. 207

- Cabitza, F. und Sarini, M. (2007). On the pathway towards ICT-support for a better and sustainable healthcare. In *Proceedings of the second European Conference on eHealth (ECEH07)*, Seiten 89–100. Springer. Referenziert auf S. XIV, XV
- Cabitza, F., Sarini, M., Simone, C., und Telaro, M. (2006). Torres, a Conceptual Framework for Articulation Work across Boundaries. In Hassanaly, P., Herrmann, T., Kunau, G., und Zacklad, M., editors, *Cooperative Systems Design: Seamless Integration of Artifacts and Conversations: Enhanced Concepts of Infrastructure for Communication*, volume 137 of *Frontiers in Artificial Intelligence and Applications*, Seiten 102–119. IOS Press. Referenziert auf S. 58, XIII, XIV, XV
- Cabitza, F. und Simone, C. (2009a). Active artifacts as bridges between context and community knowledge sources. In *C&T'09: Proceedings of the fourth international conference on Communities and technologies*, Seiten 115–124, New York, NY, USA. ACM. Referenziert auf S. XIV, XV
- Cabitza, F. und Simone, C. (2009b). LWOAD: A Specification Language to Enable the End-User Development of Coordinative Functionalities. In *Proceedings of End User Development: 2nd International Symposium, IS-EUD 2009, Siegen, Germany, March 2-4, 2009. Proceedings*, Seite 146. Springer. Referenziert auf S. XV
- Cabitza, F., Simone, C., und Sarini, M. (2008). Knowledge Artifacts as Bridges between Theory and Practice: The Clinical Pathway Case. In *Proceedings of IFIP 20th World Computer Congress, Conference on Knowledge Management in Action*, Seite 37. Springer. Referenziert auf S. XV
- Cabitza, F., Simone, C., und Sarini, M. (2009). Leveraging Coordinative Conventions to Promote Collaboration Awareness. *Computer Supported Cooperative Work (CSCW)*, 18(4):301–330. Referenziert auf S. 59, XV
- Cañas, A., Hill, G., Carff, R., Suri, N., Lott, J., Eskridge, T., Gómez, G., Arroyo, M., und Carvajal, R. (2004). Cmaptools: A knowledge modeling and sharing environment. In *Concept Maps: Theory, Methodology, Technology, Proceedings of the 1st International Conference on Concept Mapping. Pamplona, Spain: Universidad Pública de Navarra*. Referenziert auf S. 82, 296, 300, 301
- Carriero, N. und Gelernter, D. (1989). Linda in context. *Communications of the ACM*, 32(4):444–458. Referenziert auf S. 142
- Carstensen, P. und Schmidt, K. (1999). Computer supported cooperative work: new challenges to systems design. preprint, to appear in: Itoh, k. (ed.): *Handbook of human factors*, Center for Tele-Information, Danmarks Tekniske Universitet. Referenziert auf S. 27, VIII

- Christensen, U. (2001). Conventions and articulation work in a mobile workplace. *ACM SIGGROUP Bulletin*, 22(3):16–21. Referenziert auf S. X
- Comiskey, B., Albert, J., Yoshizawa, H., Jacobson, J., by Michaels, C., et al. (1998). An electrophoretic ink for all-printed reflective electronic displays. *Nature*, 394:253–255. Referenziert auf S. 201
- Convertino, G., Mentis, H. M., Rosson, M. B., Carroll, J. M., Slavkovic, A., und Ganoe, C. H. (2008). Articulating common ground in cooperative work: content and process. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, Seiten 1637–1646, New York, NY, USA. ACM. Referenziert auf S. XIV
- Corbin, J. und Strauss, A. (1993). The Articulation of Work through Interaction. *The Sociological Quarterly*, 34(1):71–83. Referenziert auf S. 14, 15, 22, 26, 27, 34, VI, XII
- Crabtree, A., O'Neill, J., Tolmie, P., Castellani, S., Colombino, T., und Grasso, A. (2006). The practical indispensability of articulation work to immediate and remote help-giving. In *CSCW '06: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, Seiten 219–228, New York, NY, USA. ACM. Referenziert auf S. XIV
- Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., und Weerawarana, S. (2002). Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI. *IEEE Internet Computing*, 6(2):86–93. Referenziert auf S. 152
- Dahme, C. und Raeithel, A. (1997). Ein tätigkeitstheoretischer Ansatz zur Entwicklung von brauchbarer Software. *Informatik-Spektrum*, 20:5–12. Referenziert auf S. 16
- Dann, H.-D. (1992). Variation von Lege-Strukturen zur Wissensrepräsentation. In Scheele, B., editor, *Struktur-Lege-Verfahren als Dialog-Konsens-Methodik. Ein Zwischenfazit zur Forschungsentwicklung bei der rekonstruktiven Erhebung subjektiver Theorien*, volume 25 of *Arbeiten zur sozialwissenschaftlichen Psychologie*, Seiten 2–41. Aschendorff. Referenziert auf S. 76, 78
- Davenport, E. (2002). Mundane knowledge management and microlevel organizational learning: An ethological approach. *Journal of the American Society for Information Science and Technology*, 53(12):1038–1046. Referenziert auf S. 48, 60, 62, X, XIV
- de Kleer, J. und Brown, J. (1981). Mental models of physical mechanisms and their acquisition. In Anderson, J., editor, *Cognitive skills and their acquisition*, Seiten 285–309. Erlbaum. Referenziert auf S. 64

Dey, A. K., Salber, D., und Abowd, G. D. (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction (HCI) Journal*, 16(2-4):97–166. Referenziert auf S. 140, 142

Diefenbruch, M., Goesmann, T., Herrmann, T., und Hoffmann, M. (2002). KontextNavigator und ExperKnowledge - Zwei Wege zur Unterstützung des Prozesswissens in Unternehmen. In Abecker, A., Hinkelmann, K., Maus, H., und Müller, H., editors, *Geschäftsprozessorientiertes Wissensmanagement*, Seiten 275–292. Springer. Referenziert auf S. 69

Davitini, M. und Simone, C. (2000). Supporting different dimensions of adaptability in workflow modeling. *Computer Supported Cooperative Work (CSCW)*, 9(3):365–397. Referenziert auf S. 31, 42, 46, 47, 52, 53, 60, 61, 70, VII, VIII, IX

Do-Lenh, S., Kaplan, F., Sharma, A., und Dillenbourg, P. (2009). Multi-finger interactions with papers on augmented tabletops. In *TEI '09: Proceedings of the 3rd International Conference on Tangible and Embedded Interaction*, Seiten 267–274, New York, NY, USA. ACM. Referenziert auf S. 79, 130

Downing, T. (1998). *Java RMI: remote method invocation*. IDG Books Worldwide, Inc., Foster City, CA, USA. Referenziert auf S. 152

Eckert, A. (1998). *Kognition und Wissensdiagnose. Die Entwicklung und empirische Überprüfung des computerunterstützten wissensdiagnostischen Instrumentariums Netzwerk-Elaborierungs-Technik (NET)*. Pabst Science Publishers. Referenziert auf S. 78

Eclipse Foundation (2009). GMF Documentation. online reference manual (http://wiki.eclipse.org/gmf_documentation), EclipseFoundation. Referenziert auf S.

Ellson, J., Gansner, E., Koutsofios, L., North, S., und Woodhull, G. (2002). Graphviz—open source graph drawing tools. In *Graph Drawing*, Lecture Notes in Computer Science, Seiten 483–484. Springer. Referenziert auf S. 252

Emery, F. und Trist, E. (1960). Socio-technical systems. *Management science, models and techniques*, 2:83–97. Referenziert auf S. 16

Engeström, Y. (1987). *Learning by expanding*. Orienta-konsultit, Helsinki. Referenziert auf S. 16, 17

Engeström, Y. (2000). Activity theory as a framework for analyzing and redesigning work. *Ergonomics*, 43(7):940–974. Referenziert auf S. 43

- Eschenfelder, K. R. (2003). The importance of articulation work to agency content management: Balancing publication and control. In *Proceedings of the Hawaii International Conference on System Sciences*, volume 5, Seite 135b, Los Alamitos, CA, USA. IEEE Computer Society. Referenziert auf S. XII
- Færgemann, L., Schilder-Knudsen, T., und Carstensen, P. H. (2005). The duality of articulation work in large heterogenous settings - a study in health care. In Schmidt, K., Gellersen, H., Mackay, W., und Beaudouin-Lafon, M., editors, *Proceedings of the 9th European Conference on Computer-Supported Cooperative Work*, Seiten 163–183. Springer. Referenziert auf S. 15, 21, 22, 28, 58, 60, XIII
- Feiner, T. (2008). Modelleditor auf Basis dynamischer Metamodelle zur Unterstützung partizipativer Modellerfassung und -reflexion. Master's thesis, University of Linz. Referenziert auf S. 207, 209, 229
- Ferscha, A., Vogl, S., Emsenhuber, B., und Wally, B. (2008). Physical shortcuts for media remote controls. In *Proceedings of the 2nd international conference on INtelligent TEchnologies for interactive enterTAINment table of contents*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering) ICST, Brussels, Belgium, Belgium. Referenziert auf S. 138
- Firestone, J. und McElroy, M. (2003). *Key Issues in the new Knowledge Management*. Butterworth-Heinemann. Referenziert auf S. 2
- Fishkin, K. P. (2004). A taxonomy for and analysis of tangible interfaces. *Personal and Ubiquitous Computing*, 8(5):347–358. Referenziert auf S. 97, 113, 116, 117, 118, 119, 125, 126, 197, 198, 199, 202, 205, 229, 277, 281, 284
- Fitzmaurice, G. (1996). *Graspable User Interfaces*. Phd-thesis, University of Toronto. Referenziert auf S. 94, 97, 99, 100, 125, 266, 285
- Fitzmaurice, G., Ishii, H., und Buxton, W. (1995). Bricks: laying the foundations for graspable user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, Seiten 442–449. ACM Press/Addison-Wesley Publishing Co. New York, NY, USA. Referenziert auf S. 94, 97, 99, 125, 263, 265, 285
- Fjeld, M. (2001). *Designing for tangible interaction*. PhD thesis, Swiss Federal Institute of Technology. Referenziert auf S. 129
- Fjeld, M., Bichsel, M., und Rauterberg, M. (1997). BUILD-IT: An Intuitive Design Tool Based on Direct Object Manipulation. In *Proceedings of the International Gesture Workshop on Gesture and Sign Language in Human-Computer Interaction*, Seiten 297–308. Springer-Verlag London, UK. Referenziert auf S. 130

- Fjuk, A. und Dirckinck-Holmfeld, L. (1997). Articulation of Actions in Distributed Collaborative Learning. *Scandinavian Journal of Information Systems*, 9(2):3–24. Referenziert auf S. VII
- Fjuk, A., Nurminen, M., und Smørdal, O. (1997). Taking Articulation Work Seriously: An Activity Theoretical Approach. Technical Report TUCS TR 120, Turku Centre for Computer Science. Referenziert auf S. 13, 15, 16, 17, 18, 22, 23, 28, 43, 44, 60, 289, VII
- Fuchs, L., Poltrock, S., und Wetzel, I. (2001). TeamSpace: an environment for team articulation work and virtual meetings. In *Proceedings of the 12th International Workshop on Database and Expert Systems Applications*, Seiten 527–531. IEEE Press. Referenziert auf S. 45, 60, 69, X
- Fujimura, J. (1987). Constructing 'Do-Able' Problems in Cancer Research: Articulating Alignment. *Social Studies of Science*, 17(2):257–293. Referenziert auf S. 2, 9, 11, 12, 24, 290, V, VI, XXI
- Furtmüller, F. (2007). Implementierung eines Frameworks für berührbare Benutzungsschnittstellen. Master's thesis, University of Linz. Referenziert auf S. 142
- Furtmüller, F. und Oppl, S. (2007). A Tuple-Space based Middleware for Collaborative Tangible User Interfaces. In *Proceedings of WETICE '07*. IEEE Press. Referenziert auf S. 142, 143, 144, 196, XXIX
- Gamma, E. und Eggenschwiler, T. (1996). The JHotDraw-Framework. online <http://www.jhotdraw.org/>. Referenziert auf S. 207, 208, 229
- Gamma, E., Helm, R., Johnson, R., und Vlissides, J. (1995). *Design patterns: elements of reusable object-oriented software*. Addison-wesley Reading, MA. Referenziert auf S. 179, 191, 195
- Gasser, L. (1986). The integration of computing and routine work. *ACM Transactions on Office Information Systems*, 4(3):205–225. Referenziert auf S. 14, 15, V
- Gellersen, H., Kortuem, G., Schmidt, A., und Beigl, M. (2004). Physical prototyping with smart-its. *IEEE Pervasive Computing*, 3(3):74–82. Referenziert auf S. 137
- Gerson, E. (2004). The organization of reconciliation in distributed work. position paper for workshop "Distributed Collective Practices" at CSCW 04. Referenziert auf S. XII
- Gerson, E. und Star, S. (1986). Analyzing due process in the workplace. *ACM Transactions on Information Systems (TOIS)*, 4(3):257–270. Referenziert auf S. 2, 9, 11, 12, 13, 15, 22, 23, 29, V

- Goguen, J. (1993). On Notation. Revised version of a paper in TOOLS 10: Technology of Object-Oriented Languages and Systems, edited by Boris Magnusson, Bertrand Meyer and Jean-Francois Perrot (Prentice-Hall, 1993), Department of Computer Science and Engineering, University of California at San Diego. Referenziert auf S. 79
- Grinter, R. (1995). Using a configuration management tool to coordinate software development. In *Proceedings of conference on Organizational computing systems*, Seiten 168–177. ACM Press. Referenziert auf S. VI
- Grinter, R. (1996). Supporting articulation work using software configuration management systems. *Computer Supported Cooperative Work (CSCW)*, 5(4):447–465. Referenziert auf S. 23, 38, 40, 60, 69, VII
- Gross, T. (2003). Ambient Interfaces: Design Challenges and Recommendations. *Human Factors and Ergonomics*, Seite 68. Referenziert auf S.
- GSI (2008). Introduction to GS1 DataMatrix. Guideline, GS1. Referenziert auf S. 146
- Hampson, I. und Junor, A. (2005). Invisible work, invisible skills: interactive customer service as articulation work. *New Technology, Work & Employment*, 20(2):166 – 181. Referenziert auf S. 13, 15, 19, 22, XIII
- Hanke, U. (2006). *Externe Modellbildung als Hilfe bei der Informationsverarbeitung und beim Lernen*. PhD thesis, University of Freiburg. Referenziert auf S. 62, 65, 70, 72
- Hasu, M. (2005). In search of sensitive ethnography of change: Tracing the invisible handoffs from technology developers to users. *Mind, Culture, and Activity*, 12(2):90 – 112. Referenziert auf S. XIII
- Herrmann, T., Hoffmann, M., Kunau, G., und Loser, K. (2002). Modelling cooperative work: Chances and risks of structuring. In *Cooperative Systems Design, A Challenge of the Mobility Age. Proceedings of COOP 2002*, Seiten 53–70. IOS press. Referenziert auf S. 50, 51, 60, 61, 62, 70, 83, XI
- Herrmann, T., Hoffmann, M., Kunau, G., und Loser, K. (2004a). A modelling method for the development of groupware applications as socio-technical systems. *Behaviour & Information Technology*, 23(2):119–135. Referenziert auf S. 50, 296
- Herrmann, T., Hoffmann, M., Loser, K., und Moysich, K. (2000). Semistructured models are surprisingly useful for user-centered design. In Dieng, R., Giboin, A., Karsenty, L., und De Michelis, G., editors, *Designing Cooperative Systems. Proceedings of COOP 2000*, Seiten 159–174, Amsterdam. IOS press. Referenziert auf S. 52

- Herrmann, T., Kunau, G., Loser, K., und Menold, N. (2004b). Socio-technical walk-through: designing technology along work processes. In *Artful integration: interweaving media, materials and practices. Proceedings of the eighth Conference on Participatory design.*, Seiten 132–141. ACM Press New York, NY, USA. Referenziert auf S. 50
- Holmquist, L., Redström, J., und Ljungstrand, P. (1999). Token-Based Acces to Digital Information. In *Proceedings of the 1st international Symposium on Handheld and Ubiquitous Computing*, Seiten 234–245. Springer-Verlag London, UK. Referenziert auf S. 97, 102, 119, 125, 268, 269
- Holzner, S. (2004). *Eclipse*. O'Reilly Germany. Referenziert auf S.
- Hornecker, E. (2004). *Tangible User Interfaces als kooperationsunterstützendes Medium*. Phd-Thesis, University of Bremen. Dept. of Computing. Referenziert auf S. 129, 130
- Hughes, F. (1971). *The Sociological Eye*. Aldine de Gruyter. Referenziert auf S. 10, 13
- Huss, J. (2003). Diagnose und Unterstützung mentaler Wissensrepräsentationen in Projektteams - Eine Fallstudie. Master's thesis, Technical University of Berlin. Referenziert auf S. 73, 76, 78
- Ifenthaler, D. (2006). *Diagnose lernabhängiger Veränderung mentaler Modelle - Entwicklung der SMD-Technologie als methodologisches Verfahren zur relationalen, strukturellen und semantischen Analyse individueller Modellkonstruktionen*. PhD thesis, University of Freiburg. Referenziert auf S. 65, 66, 68, 69, 70, 71, 72, 76, 77, 78, 80, 83, 290
- Ishii, H. (2008). Tangible bits: beyond pixels. In *Proceedings of the 2nd international conference on Tangible and embedded interaction*. ACM New York, NY, USA. Referenziert auf S. 97, 119, 120, 121, 122, 123, 126, 128, 284, 285
- Ishii, H. und Ullmer, B. (1997). Tangible bits: towards seamless interfaces between people, bits and atoms. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, Seiten 234–241. ACM Press New York, NY, USA. Referenziert auf S. 94, 97, 100, 125, 267, 268
- ISO (2000). Iso 9001:2000 - quality management systems - requirements. International standard, ISO. Referenziert auf S. 1
- ISO (2005). Iso 9000:2005 - quality management systems - fundamentals and vocabulary. International standard, ISO. Referenziert auf S. 1
- ISO JTC1/SC31 (2006). Information technology – automatic identification and data capture techniques – qr code 2005 bar code symbology specification. International Standard 18004:2006, ISO/IEC. Referenziert auf S. 133, 146

- ISO JTC1/SC34 (2008). Topic Maps Constraint Language. draft standard, ISO/IEC. Referenziert auf S. 240
- ISO JTC1/SC34/WG3 (2006). Information Technology - Topic Maps - Part 3: XML Syntax. International standard, ISO. Referenziert auf S. 252
- ISO JTC1/SC34/WG3 (2008). Information Technology - Topic Maps - Part 2: Data Model. International Standard 13250-2, ISO/IEC. Referenziert auf S. 232, 236, 251
- James, M. (1997). *Microcontroller Cookbook - PIC & 8051*. Butterworth-Heinemann. Referenziert auf S. 137
- Johnson-Laird, P. N. (1981). Mental models in cognitive science. *Cognitive Science*, 4(1):71–115. Referenziert auf S. 64
- Jørgensen, H. (2004). *Interactive Process Models*. PhD thesis, Department of Computer and Information Sciences, Norwegian University of Science and Technology Trondheim. Referenziert auf S. 3, 56, 57, 60, 61, 70, 83, 289, XII, XVIII
- Kaghan, W. und Lounsbury, M. (2004). Articulation Work and the Institutional Elements of Organizational Artifacts: The Case of Contracts and Contracting. pre-print, to appear in: Rafaeli, a. & pratt, m.: *Artifacts and organizations*, Cornell University. Referenziert auf S. XIV
- Kaghan, W. N. und Lounsbury, M. (2006). Artifacts, articulation work, and institutional residue. In Rafaeli, A. und Pratt, M. G., editors, *Artifacts and organizations: Beyond mere symbolism*, Seiten 259 – 275. Lawrence Erlbaum Associates Publishers. Referenziert auf S. XIV
- Kaltenbrunner, M. und Bencina, R. (2007). reacTIVision: a computer-vision framework for table-based tangible interaction. In *TEI '07: Proceedings of the 1st international conference on Tangible and embedded interaction*, Seiten 69–74, New York, NY, USA. ACM Press. Referenziert auf S. 146, 196
- Kaltenbrunner, M., Jorda, S., Alonso, M., und Geiger, G. (2006). The reactable*: A collaborative musical instrument. In *Proceedings of WETICE '06*. IEEE Press. Referenziert auf S. 130, 202, 204
- Kato, H., Billinghurst, M., Poupyrev, I., Imamoto, K., und Tachibana, K. (2000). Virtual object manipulation on a table-top AR environment. In *IEEE and ACM International Symposium on Augmented Reality, 2000.(ISAR 2000). Proceedings*, Seiten 111–119. Referenziert auf S. 133, 145

- Kim, D. (1993). *A Framework and Methodology for Linked Individual and Organisational Learning: Applications in TQM and Product Development*. PhD thesis, Sloan School of Management, Massachusetts Institute of Technology. Referenziert auf S. 2, 71
- Klemmer, S., Li, J., Lin, J., und Landay, J. (2004). Papier-mâché: Toolkit support for tangible input. *CHI Letters, Human Factors in Computing Systems: CHI2004.*, 6(1). Referenziert auf S. 97, 116, 141, 281
- Klemmer, S., Thomsen, M., Phelps-Goodman, E., Lee, R., und Landay, J. (2002). Where do web sites come from? capturing and interacting with design history. chi 2002. *Human Factors in Computing Systems, CHI Letters*, 4(1). Referenziert auf S. 307
- Kling, R. und Star, S. (1998). Human centered systems in the perspective of organizational and social informatics. *ACM SIGCAS Computers and Society*, 28(1):22–29. Referenziert auf S. VIII
- Kluwe, R. H. (1990). Wissen. In Sarges, W., editor, *Management-Diagnostik*, Seiten 174–181. Hogrefe, Göttingen. Referenziert auf S. 76
- Koleva, B., Benford, S., Ng, K., und Rodden, T. (2003). A Framework for Tangible User Interfaces. In *Workshop-Proceedings on Real World User Interfaces, Mobile HCI Conference 03*, Seiten 257–264. Referenziert auf S. 97, 111, 126, 275, 276, 277
- Krasner, G. und Pope, S. (1988). A cookbook for using the model-view controller user interface paradigm in Smalltalk-80. *Journal of Object-oriented programming*, 1(3):26–49. Referenziert auf S.
- Larkin, J. und Simon, H. (1987). Why a diagram is (sometimes) worth ten thousand words. *Cognitive science*, 11(1):65–100. Referenziert auf S. 311
- Larsen, S. B. und Bardram, J. E. (2008). Competence articulation: alignment of competences and responsibilities in synchronous telemedical collaboration. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, Seiten 553–562, New York, NY, USA. ACM. Referenziert auf S. XV
- Lave, J. und Wenger, E. (1991). *Situated learning: Legitimate peripheral participation*. Cambridge University Press. Referenziert auf S. 48
- Lenoir, T. (1994). Was the last turn the right turn? the semiotic turn and a. j. greimas. *Configurations*, 2(1):119–136. Referenziert auf S. VI
- Leont'ev, A. (1972). The Problem of Activity in Psychology. *Voprosy filosofii (english translation)*, (9):95–108. Referenziert auf S. 16, 43

Literaturverzeichnis

- Leont'ev, A. (1978). *Activity, Consciousness, and Personality*. Prentice-Hall. Referenziert auf S. 15, 73
- Maier, M. (2008). Organizational Memories - Konzepte und Realisierungen. Master's thesis, University of Linz. Referenziert auf S. 69, XI
- Mandl, H. und Fischer, F. (2000). Mapping-Techniken und Begriffsnetze in Lern-und Kooperationsprozessen. In Mandl, H. und Fischer, F., editors, *Wissen sichtbar machen*. Hogrefe. Referenziert auf S. 78
- Mark, G., Gonzalez, V. M., Sarini, M., und Simone, C. (2002a). Reconciling Different Perspectives: An Experiment on Technology Support for Articulation. In Blay-Fornarino, M., Pinna-Dery, A., Schmidt, K., und Zarate, P., editors, *Proceedings of COOP 2002: Cooperative Systems Design: A Challenge of the Mobility Age*, Fontiers in Artificial Intelligence and Applications, Seiten 23–37. IOS Press. Referenziert auf S. 53, XI
- Mark, G., Gonzalez, V. M., Sarini, M., und Simone, C. (2002b). Supporting articulation with the reconciler. In *CHI Extended Abstracts 2002*, Seiten 814–815. Referenziert auf S. XI
- McAffer, J. und Lemieux, J. (2005). *Eclipse Rich Client Platform: Designing, Coding, and Packaging Java (TM) Applications*. Addison-Wesley Professional. Referenziert auf S. 209
- Mi, P. und Scacchi, W. (1991). Modeling Articulation Work in Software Engineering Processes. In *Proceedings of the First International Conference on the Software Process*, Seiten 188–201. IEEE Press. Referenziert auf S. 33, 34, V
- Montessori, M. (2005). *The Montessori Method*. Kessinger Publishing. Referenziert auf S.
- Moore, B., Dean, D., Gerber, A., Wagenknecht, G., und Vanderheyden, P. (2004). Eclipse Development using the Graphical Editing Framework and the Eclipse Modeling Framework. Ibm redbooks, IBM. Referenziert auf S.
- Nardi, B. und Kapteinin, V. (2006). *Acting with Technology - Activity Theory and Interaction Design*. MIT Press. Referenziert auf S. 16
- Neubauer, M. (2008). Abbildung generischer Modelle auf Topic Maps. Master's thesis, University of Linz. Referenziert auf S. 250, 251
- Nonaka, I. und Takeuchi, H. (1995). *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*. Oxford University Press. Referenziert auf S. 2, 13, 81

- Norman, D. (1983a). Some observations on mental models. In Gentner, D. und Stevens, A., editors, *Mental models*, Seiten 7–14. Lawrence Erlbaum Associates. Referenziert auf S. 67, 211
- Norman, D. A. (1983b). Design principles for human-computer interfaces. In *CHI '83: Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, Seiten 1–10, New York, NY, USA. ACM. Referenziert auf S. 64
- Novak, J. und Cañas, A. J. (2006). The theory underlying concept maps and how to construct them. Technical Report IHMC CmapTools 2006-01, Florida Institute for Human and Machine Cognition. Referenziert auf S. 78, 80, 81, 82, 290
- Olesen, F. und Markussen, R. (2003). Reconfigured medication: Writing medicine in a sociotechnical practice. *Configurations*, 11(3):351–381. Referenziert auf S. 2, XII
- Oppl, S. (2004). Context-aware Group-Interaction. Master's thesis, University of Linz, Department for Pervasive Computing. Referenziert auf S. 141
- Oppl, S. (2006). Towards Intuitive Work Modeling with a Tangible Collaboration Interface Approach. In *Proceedings of WETICE '06*. IEEE Press. Referenziert auf S. XXIX
- Oppl, S. (2007a). Flexibility of Content for Organisational Learning - A Topic Map Approach. Master's thesis, University of Linz. Referenziert auf S. 251, XXIX
- Oppl, S. (2007b). Spielen Sie noch? - Bausteine im Unternehmenskontext. In Paul-Stueve, T., editor, *Workshop-Proceedings der 7. fachübergreifenden Konferenz Mensch und Computer 2007*. Verlag der Bauhaus-Universität Weimar. Referenziert auf S. XXIX
- Oppl, S. (2008a). Begreifbare Modellierung von Arbeit. In *Workshop-Proceedings der 8. fachübergreifenden Konferenz Mensch und Computer 2008*. logos Verlag. Referenziert auf S. XXIX
- Oppl, S. (2008b). Graspable work modeling. In *Proceedings of Mensch und Computer 2008*. Oldenbourg Verlag. Referenziert auf S. XXIX
- Oppl, S. (2009a). A Tabletop Interface to support Concept Mapping. In *Proceedings of EduMedia 2009*. Salzburg Research. Referenziert auf S. XXIX
- Oppl, S. (2009b). Konsistente Verwendung von Metaphern als Erfolgskriterium für komplexe Tangible User Interfaces. In *Workshop-Proceedings der 9. fachübergreifenden Konferenz Mensch und Computer 2009*. Referenziert auf S. XXIX

- Oppl, S. (2009c). Unterstützung expliziter Articulation Work – Statement of Interest for Participation in the Session on disruptive or seamless HCI in eLearning. In *Proceedings of LATEL (Interdisciplinary approaches to technology-enhanced learning)*. TU Darmstadt. Referenziert auf S. XXIX
- Oppl, S. (2009d). Unterstützung expliziter Articulation Work durch Externalisierung von Arbeitswissen. In Peschl, M. und Risku, H., editors, *Kognitive und technologische Konzepte für kooperatives Lernen*. Vienna University Press. Referenziert auf S. XXIX
- Oppl, S. und Peherstorfer, P. (2007). Human Intervention in cross-organizational Process Development. In *Proceedings of the 4th International Conference on Knowledge Management (ICKM 2007)*. Referenziert auf S. XXIX
- Oppl, S. und Stary, C. (2005). Towards Human-Centered Design of Diagrammatic Representation Schemes. In Dix, A. und Dittmar, A., editors, *Proceedings of the 4th International Workshop on Task Models and Diagrams for User Interface Design (TAMODIA 2005)*, Seiten 55–62. ACM Press New York, NY, USA. Referenziert auf S. 243, XXIX
- Oppl, S. und Stary, C. (2009). Tabletop concept mapping. In *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction (TEI '09)*. ACM Press. Referenziert auf S. XXIX
- Oppl, S., Stary, C., und Auinger, A. (2006). Towards Tangible Work Modeling. In *Proceedings of Mensch und Computer 2006*, Seiten 400–405. Oldenburg Wissenschaftsverlag. Referenziert auf S. XXIX
- Patten, J. und Ishii, H. (2007). Mechanical constraints as computational constraints in tabletop tangible interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '07)*, Seiten 809–818, New York, NY, USA. ACM. Referenziert auf S. 95
- Patten, J., Ishii, H., Hines, J., und Pangaro, G. (2001). Sensetable: a wireless object tracking platform for tangible interfaces. In *Proceedings of the ACM Conference on Human Factors in Computing Systems 2001 (CHI '01)*. Referenziert auf S. 129, 130
- Pedersen, E. W. und Hornb, K. (2009). mixitui: a tangible sequencer for electronic live performances. In *TEI '09: Proceedings of the 3rd International Conference on Tangible and Embedded Interaction*, Seiten 223–230, New York, NY, USA. ACM. Referenziert auf S. 202, 204
- Pepper, S. (2000). The tao of topic maps. In *Proceedings of XML Europe*. Referenziert auf S. 232

- Piaget, J. (1976). *Die Äquilibrium der kognitiven Strukturen*. Klett-Cotta. Referenziert auf S. 65, 95
- Pirnay-Dummer, P. N. (2006). *Expertise und Modellbildung - MITOCAR*. PhD thesis, University of Freiburg. Referenziert auf S. 65, 70
- Raposo, A., Gerosa, M., und Fuks, H. (2004). Combining Communication and Coordination Toward Articulation of Collaborative Activities. In *Proceedings of Groupware: Design, Implementation, and Use: 10th International Workshop, CRIWG 2004*. Springer. Referenziert auf S. 12, 54, 60, 83, XI, XII
- Raposo, A. und Hugo, F. (2002). Defining task interdependencies and coordination mechanisms for collaborative systems. In *Proceedings of COOP 2002*, Seiten 88–113. IOS Press. Referenziert auf S. X, XI
- Raposo, A., Magalhães, L., Ricarte, I., und Fuks, H. (2001). Coordination of collaborative activities: A framework for the definition of tasks interdependencies. In *Proceeding of the 7th International Workshop on Groupware-CRIWG*. Referenziert auf S. 60, X, XI
- Rath, H. (2003). *The Topic Maps Handbook*. empolis GmbH. Referenziert auf S. 232
- Red Hat Middleware (2007). Hibernate Reference Documentation. Reference documentation, Red Hat Middleware. Referenziert auf S. 252
- Resnick, M., Martin, F., Berg, R., Borovoy, R., Colella, V., Kramer, K., und Silverman, B. (1998). Digital manipulatives: new toys to think with. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, Seiten 281–287, New York, NY, USA. ACM Press. Referenziert auf S. 95
- Rodden, T. (1995). The COMIC Project - Computer-based Mechanisms of Interaction in Cooperative Work. online: <http://www.comp.lancs.ac.uk/computing/research/softeng/comic/>, ESPRIT Basic research Project 622S
- Rohs, M. (2005). Visual code widgets for marker-based interaction. In *25th IEEE International Conference on Distributed Computing Systems Workshops, 2005*, Seiten 506–513. IEEE Press. Referenziert auf S. 145
- Rohs, M. und Gfeller, B. (2004). Using camera-equipped mobile phones for interacting with real-world objects. In *Advances in Pervasive Computing*, Seiten 265–271. Austrian Computer Society (OCG). Referenziert auf S. 146, XXVII
- Rumbaugh, J., Jacobson, I., und Booch, G. (2004). *Unified Modeling Language Reference Manual, The*. Pearson Higher Education. Referenziert auf S. 244, 248, 250

- Rumelhart, D. und Norman, D. (1978). Accretion, tuning, and restructuring: Three modes of learning. In Cotton, J. und Klatzky, R., editors, *Semantic factors in cognition*, Seiten 37–53. Erlbaum, Hillsdale, N.J. Referenziert auf S. 62
- Sachs, P. (1995). Transforming work: collaboration, learning, and design. *Communications of the ACM*, 38(9):36–44. Referenziert auf S. 1, 24
- Sarini, M. (2003). *Alignment of meanings and of protocols as a form of articulation work in cooperation*. PhD thesis, University of Torino. Referenziert auf S. 42, XII
- Sarini, M. und Simone, C. (2002a). Recursive articulation work in ariadne: The alignment of meanings. In *Proceedings of COOP 2002*, Seiten 191–206. Referenziert auf S. 14, 22, 52, 53, 58, 59, 60, 61, 70, 83, XI, XIII
- Sarini, M. und Simone, C. (2002b). The Reconciler: supporting actors in meaning negotiation. In *Proceedings of the Workshop on Meaning Negotiation (MeaN-02) at AAAI-02*. Referenziert auf S. 60, XI
- Scheele, B. und Groeben, N. (1988). *Dialog-Konsens-Methoden zur Rekonstruktion Subjektiver Theorien Die Heidelberger Struktur-Lege-Technik (SLT), konsensuale Ziel-Mittel-Argumentation und kommunikative Flussdiagramm-Beschreibung von Handlungen*. Francke, Tuebingen. Referenziert auf S. 76, 77, 78, 84
- Scheer, A. und Nuettgens, M. (2000). Aris architecture and reference models for business process management. *Business Process Management: Models, Techniques, and Empirical Studies*, Seiten 376–389. Referenziert auf S. 296
- Scheer, A.-W. (2003). *ARIS – Business Process Modeling*. Springer, 3 edition. Referenziert auf S. 250
- Schilit, B., Adams, N., und Want, R. (1994). Context-aware computing applications. In *Proceedings of the Workshop on Mobile Computing Systems and Applications*, Seiten 85–90. Referenziert auf S. 140
- Schmidt, K. (1990). Analysis of Cooperative Work. A Conceptual Framework. Technical Report Risø-M-2890, Risø National Laboratory. Referenziert auf S. 14, 40, V, VI, VII
- Schmidt, K. (1994). Cooperative Work and its Articulation. *Travail Humain*, 57(4):345–366. Referenziert auf S. 10, 11, 60, VI
- Schmidt, K. (2002). The problem with ‘awareness’. *Computer Supported Cooperative Work*, 11(3):285–298. Referenziert auf S. XI

- Schmidt, K. und Bannon, L. (1992). Taking CSCW seriously: Supporting Articulation Work. *Computer Supported Cooperative Work (CSCW)*, 1(1):7–40. Referenziert auf S. 1, 3, 23, 35, 36, 37, 38, 40, 46, 50, 52, V, VI, VII, IX
- Schmidt, K. und Simone, C. (1996). Coordination mechanisms: Towards a conceptual foundation of CSCW systems design. *Computer Supported Cooperative Work*, 5(2/3):155–200. Referenziert auf S. 29, 30, 40, 41, 42, 44, 46, 47, VI, VII, VIII, IX
- Schmidt, K. und Simone, C. (1999). Cooperative work is seamless: Integrating the support of the many modalities of articulation work. Working paper 52, Center for Tele-Information. Referenziert auf S. VIII
- Schmidt, K. und Simone, C. (2000). Mind the Gap!: Towards a unified view of CSCW. In *Proceedings of COOP2000: The Fourth International Conference on the Design of Cooperative Systems*, Sophia Antipolis, France. INRIA. Referenziert auf S. 17, 22, 28, 60, 79, IX
- Schmidt, K., Simone, C., Divitini, M., Carstensen, P., und Sørensen, C. (1995). A ‘contrat sociale’ for cscw systems. Working paper, Roskilde University. Referenziert auf S. VI
- Seel, N. (2003). *Psychologie des Lernens*. Ernst Reinhardt Verlag, München Basel, 2nd edition. Referenziert auf S. 65
- Seel, N. M. (1991). *Weltwissen und mentale Modelle*. Hogrefe, Göttingen u.a. Referenziert auf S. 62, 64, 65, 66, 67, 70, 72, 81
- Seiringer, G. (2008). Entwicklung eines Editors für die Erstellung und Visualisierung von Topic Maps. Master’s thesis, University of Linz. Referenziert auf S.
- Semmer, N. und Udris, I. (2004). Bedeutung und Wirkung von Arbeit. In Schuler, H., editor, *Lehrbuch Organisationspsychologie*, Seiten 157–195. Huber, Bern, 3rd edition. Referenziert auf S. 9
- Senge, P. (1990). *The Fifth Discipline: The Art and Practice of the Learning Organization*. Doubleday/Currency. Referenziert auf S. 71, 75
- Senge, P., Kleiner, A., Roberts, C., und Smith, B. (1994). *The fifth discipline fieldbook: Strategies and tools for building a learning organization*. Broadway Business. Referenziert auf S. 75
- Shaer, O., Leland, N., Calvillo-Gamez, E., und Jacob, R. (2004). The TAC paradigm: specifying tangible user interfaces. *Personal and Ubiquitous Computing*, 8(5):359–369. Referenziert auf S. 97, 114, 124, 126, 128, 277

- Shapiro, S. und Wilk, M. (1965). An analysis of variance test for normality. *Biometrika*, 52(3):591–599. Referenziert auf S. 302
- Shinar, J. (2004). *Organic light-emitting devices: a survey*. Springer. Referenziert auf S. 201
- Shipman, F. und Hsieh, H. (2000). Navigable history: a reader's view of writer's time. *New review of hypermedia and multimedia*, 6(1):147–167. Referenziert auf S. 92, 307
- Simone, C. (2000). Making classification schemes a first class notion in cscw. In *Proceedings of the 1st CISCPH workshop on Cooperative Organization of Common Information Spaces*. Referenziert auf S. IX, X, XVI
- Simone, C. (2002). Unifying or reconciling when constructing organizational memory? some open issues. In *Knowledge management and organizational memories*, Seiten 137–143. Kluwer Academic Publishers. Referenziert auf S. XI
- Simone, C. und Bandini, S. (1997). Compositional features for promoting awareness within and across cooperative applications. In *Proceedings of GROUP '97*, Seiten 358–367, New York, NY, USA. ACM. Referenziert auf S. VII
- Simone, C. und Divitini, M. (1997). Ariadne: Supporting Coordination through a Flexible Use of the Knowledge on Work Processes. *Journal of Universal Computer Science*, 3(8):865–898. Referenziert auf S. VIII
- Simone, C., Divitini, M., und Schmidt, K. (1995). A notation for malleable and interoperable coordination mechanisms for CSCW systems. In *Proceedings of conference on Organizational computing systems*, Seiten 44–54. ACM New York, NY, USA. Referenziert auf S. VII
- Simone, C., Mark, G., und Giubbilei, D. (1999). Interoperability as a means of articulation work. *ACM SIGSOFT Software Engineering Notes*, 24(2):39–48. Referenziert auf S. 3, VIII, X
- Simone, C. und Sarini, M. (2001). Adaptability of classification schemes in cooperation: what does it mean? In *Proceedings of the 2nd CISCPH workshop on Cooperative Organization of Common Information Spaces*. Referenziert auf S. X
- Stachowiak, H. (1973). *Allgemeine Modelltheorie*. Springer Wien. Referenziert auf S. 71, 77, 233
- Star, S. L. und Strauss, A. (1999). Layers of silence, arenas of voice: The ecology of visible and invisible work. *Computer Supported Cooperative Work: The Journal of Collaborative Computing*, 8(1/2):9 – 30. Referenziert auf S. 12, 14, 19, 22, IX

- Stary, C. (1994). *Interaktive Systeme: Softwareentwicklung und Softwareergonomie*. Vieweg. Referenziert auf S. 144
- Strauss, A. (1985). Work and the Division of Labor. *The Sociological Quarterly*, 26(1):1–19. Referenziert auf S. 2, 9, 12, 21, 23, 63, V, XXI
- Strauss, A. (1988). The Articulation of Project Work: An Organizational Process. *The Sociological Quarterly*, 29(2):163–178. Referenziert auf S. 2, 13, 14, 21, 63, V
- Strauss, A. (1993). *Continual Permutations of Action*. Aldine de Gruyter, New York. Referenziert auf S. 11, 13, 17, 19, 21, 22, 24, 61, 62, 63, 64, 290, 291, VI, VII
- Suchman, L. (1987). *Plans and Situated Actions: The Problem of Human-Machine Communication*. Cambridge University Press. Referenziert auf S. 41
- Suchman, L. (1995). Making work visible. *Communications of the ACM*, 38(9):56–64. Referenziert auf S. 1, 22
- Suchman, L. (1999). Supporting articulation work: Aspects of a feminist practice of office technology production. In Kling, R., editor, *Computerization and Controversy - value conflicts and social choices*, Seiten 407–423. Academic Press, Inc. Orlando, FL, USA. Referenziert auf S. 22, VIII
- Suzuki, H. und Kato, H. (1995). Interaction-level support for collaborative learning: AlgoBlock—an open programming language. In *Proceedings of the first international conference on Computer support for collaborative learning table of contents*, Seiten 349–355, Hillsdale, NJ, USA. L. Erlbaum Associates Inc. Referenziert auf S. 94
- Tanenbaum, K. und Antle, A. N. (2009). A tangible approach to concept mapping. In Ao, S.-I., editor, *IAENG TRANSACTIONS ON ENGINEERING TECHNOLOGIES VOLUME 2: Special Edition of the World Congress on Engineering and Computer Science*, volume 1127, Seiten 121–132. AIP. Referenziert auf S. 79, 130
- The LEGO Group (2002). Die Wissenschaft von LEGO SERIOUS PLAY. brochure, The LEGO Group. Referenziert auf S.
- Ullmer, B. (2002). *Tangible interfaces for manipulating aggregates of digital information*. PhD thesis, Massachusetts Institute of Technology. Referenziert auf S. 97, 109, 114, 124, 126
- Ullmer, B. und Ishii, H. (1997). The metaDESK: models and prototypes for tangible user interfaces. In *Proceedings of the 10th annual ACM symposium on User interface software and technology*, Seiten 223–232, New York. ACM Press. Referenziert auf S. 101, 108, 267, 268

Literaturverzeichnis

- Ullmer, B. und Ishii, H. (2000). Emerging frameworks for tangible user interfaces. *IBM Systems Journal*, 39(3):915–931. Referenziert auf S. 97, 106, 107, 111, 119, 126, 197, 271, XXVII
- Ullmer, B., Ishii, H., und Jacob, R. (2005). Token + constraint systems for tangible interaction with digital information. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 12(1):81–118. Referenziert auf S. 109, 110, 119, 121, 273, 274, 275, 285
- Underkoffler, J. und Ishii, H. (1999). Urp: A luminous-tangible workbench for urban planning and design. In *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, Seiten 386–393. ACM New York, NY, USA. Referenziert auf S. 97, 105, 119, 126, 270
- Van Laerhoven, K., Villar, N., Schmidt, A., Gellersen, H., Hakansson, M., und Holmquist, L. (2003). Pin&Play: the surface as network medium. *IEEE Communications Magazine*, 41(4):90–95. Referenziert auf S. 137
- Van Someren, M., Barnard, Y., und Sandberg, J. (1994). *The think aloud method: A practical guide to modelling cognitive processes*. Academic Press. Referenziert auf S. 73, 74, 75
- Vatant, B. (2004). Ontology-driven Topic Maps. In *Proceedings of XML Europe 2004*, Amsterdam. Referenziert auf S. 232
- von Krogh, G., Nonaka, I., und Ichijō, K. (2000). *Enabling Knowledge Creation: How to Unlock the Mystery of Tacit Knowledge and Release the Power of Innovation*. Oxford University Press US. Referenziert auf S. 2
- Wagner, D. und Schmalstieg, D. (2003). ARToolKit on the PocketPC platform. In *IEEE International Augmented Reality Toolkit Workshop, 2003*, Seiten 14–15. IEEE Press. Referenziert auf S. 149
- Weiser, M. (1991). The computer for the 21st century. *Scientific American*, 265. Referenziert auf S. 95, 101, 140
- Wellner, P. (1993). Interacting with paper on the DigitalDesk. *Communications of the ACM*, 36:87–96. Referenziert auf S. 94
- Wenger, E. (1998). Communities of practice - learning as a social system. *The Systems Thinker*, 9(5). Referenziert auf S. XI
- Wenger, E. (1999). *Communities of Practice: Learning, Meaning, and Identity*. Cambridge University Press. Referenziert auf S. 48, 49, 60, 69

ZigBee Alliance (2007). Zigbee Specification. Specification r17, ZigBee Alliance. Referenziert auf S. 138

Zuckerman, O., Arida, S., und Resnick, M. (2005). Extending tangible interfaces for education: digital montessori-inspired manipulatives. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI)*, Seiten 859–868. ACM Press New York, NY, USA. Referenziert auf S. 95, 201