

3 SYSTEME AUF BASIS VISUELLER SPRACHEN

In diesem Kapitel werden Systeme vorgestellt, die visuelle Sprachen verwenden und für kooperative Situationen ausgelegt sind. In diesem Zusammenhang werden zu Beginn die Basis-Systeme erläutert, auf denen die Entwicklung der Diskussionsunterstützung mit dem DiscBoard aufsetzt. Anschließend werden weitere Systeme erläutert, die visuelle Sprachen als Kommunikationsmedien verwenden. In diesem Kontext werden auch Systeme zur Kreativitätsunterstützung mit angesprochen, die unter Umständen nur für die Benutzung durch eine einzelne Person entwickelt wurden. Das Kapitel schließt mit einer Synopse der erläuterten Systeme. Das System DiscBoard selbst sowie der FreeStyler werden dann später im zweiten Teil dieser Arbeit vorgestellt.

3.1 Architektur der Basis-Systeme für die Diskussionsunterstützung

Das System zur Diskussionsunterstützung DiscBoard setzt auf drei verschiedenen Teilsystemen auf: MatchMaker (Hoppe & Tewissen, 1994; Tewissen, 1996; Mühlenbrock, Tewissen & Hoppe, 1997; Gaßner, Tewissen et al., 1998), Dalis (Mühlenbrock, Tewissen & Hoppe, 1997; Mühlenbrock, M. & Loesch, A., 1998; Gaßner, Tewissen et al., 1998) und CardBoard (Tewissen, 1996, Mühlenbrock, Tewissen & Hoppe, 1997; Gaßner, Tewissen et al., 1998; Hoppe, Gaßner et al., 2000), die in den folgenden Unterkapiteln erläutert werden. Während es sich bei MatchMaker und CardBoard um eine C++-Entwicklung handelt, ist das Dalis-System in Prolog implementiert. Die Kommunikation zwischen den drei beteiligten Systemen basiert in allen Fällen auf TCP/IP. MatchMaker und CardBoard sind für Windows-Plattformen entwickelt worden. Dalis ist an Quintus-Prolog gebunden, das hier unter UNIX eingesetzt wird⁷. Die Kombination aller drei Systeme zu einem Gesamtsystem bewirkt eine deutliche Zunahme der Komplexität, die für den alltäglichen Einsatz nicht unbedingt erwünscht ist. Insgesamt steht deshalb bei dieser Entwicklung nicht die Architektur im Vordergrund, sondern die Möglichkeiten, auf visuellen Sprachen zu operieren.

⁷ Zum Zeitpunkt der Dalis-Entwicklung war Quintus-Prolog unter UNIX die geeignetste Prolog-Implementierung unter der Voraussetzung, dass TCP/IP verwendet werden sollte.

Die Grundfunktionalität der drei Basis-Systeme war kein Entwicklungsgegenstand dieser Arbeit, soll im Weiteren aber trotzdem kurz erläutert werden. MatchMaker ist mittlerweile reimplementiert worden in einer Java-Version (Jansen, Pinkwart & Tewissen, 2001; Jansen, 2002), die weitgehend andere Synchronisationskonzepte verfolgt. Diese neue Version wird aber erst im zweiten Fallbeispiel dieser Arbeit, dem FreeStyler, verwendet.

3.1.1 MatchMaker

MatchMaker (Hoppe & Tewissen, 1994; Tewissen, 1996; Mühlenbrock, Tewissen & Hoppe, 1997; Gaßner, Tewissen et al., 1998) organisiert die Synchronisation von Userinterface-Objekten mehrerer verteilter Applikationen. Diese Synchronisation wird im Weiteren auch als Kopplung bezeichnet.

Mit MatchMaker wird eine replizierte Architektur realisiert, in der eigenständige Applikationen temporär synchronisiert werden. Die Synchronisation beruht auf der Verteilung von Userinterface-Events, die dann jede Applikation für sich ausführt. Dafür stellt MatchMaker eine Funktionsbibliothek für Userinterface-Objekte zur Verfügung, die die Koppelungsfunktion anbieten. Prinzipiell ist es durch diese Art der Kopplung über Events möglich, verschiedenartige Objekte zu synchronisieren. Für das jeweilige Objekt muss jeweils definiert sein, wie der Event interpretiert werden soll.

Eine Replikation besteht hier in der Nachbildung der Inhaltsdaten, die in der Kooperation erarbeitet werden. Die replizierte Architektur, die durch MatchMaker aufgebaut wird, hat den Vorteil, dass die Daten, die die Arbeitsinhalte bilden, wirklich für jede Applikation vorhanden sind. So kann jede der beteiligten Applikationen beendet werden, ohne dass es bei den anderen zu einem Datenverlust kommt. Außerdem bewirkt sie eine gewisse Sicherheit im Fall von Systemsabstürzen. Die Replikation ist in einer Client/Server-Architektur realisiert. MatchMaker organisiert darin die Kommunikation und speichert dafür Daten darüber, wo und welche Objekte gekoppelt sind. Die konkreten Daten der Objekte liegen aber nur in den Applikationen, den Clients vor.

Anhand der MatchMaker-Architektur, soll noch eine weitere Eigenschaft erläutert werden und zwar die partielle Kopplung. Das bedeutet, dass nur Teile einer Applikation gekoppelt werden. So ist es beispielsweise möglich, private und öffentliche Arbeitsbereiche zu erstellen. Abbildung 5 zeigt ein Beispiel für eine partielle Kopplung. Nur die gekoppelten Bereiche sind für andere sichtbar. Bei der Diskussionsunterstützung werden gesamte Workspaces synchronisiert. Mit MatchMaker ist es möglich, mehrere Workspaces mehrerer Applikationen zu koppeln. Dies ist in Abbildung 5 durch das Paar roter und blauer Pfeile angedeutet.

Über MatchMaker verläuft außerdem die Kommunikation der Applikationen mit einem weiteren Basis-System, dem Dalis-System. Dort wird für die Diskussionsunterstützung das Diskussionsmodell verwaltet.

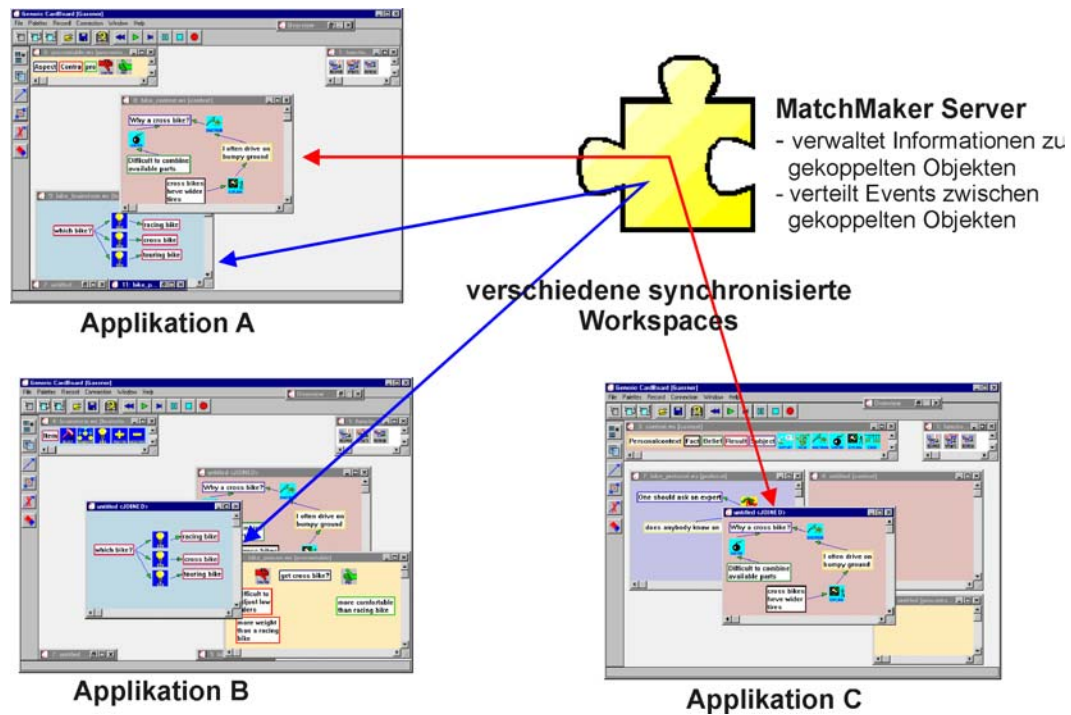


Abbildung 5

Der MatchMaker Server synchronisiert Objekte unterschiedlicher Applikationen. Im dargestellten Beispiel wird jeweils ein rosafarbener Workspace der Applikation A und C sowie ein blauer Workspace der Applikation A und B synchronisiert.

3.1.2 Dalis

Dalis ist das zweite eingebundene Teilsystem (Mühlenbrock, Tewissen & Hoppe, 1997; Mühlenbrock & Hoppe, 1999; Mühlenbrock, M. & Loesch, A., 1998; Gaßner, Tewissen et al., 1998). Es führt immer ein Monitoring der über MatchMaker angebotenen Applikationen durch. In dieser Grundfunktion hält Dalis den Ist-Zustand der Applikationen vor. Das beinhaltet die Zustände von Karten, Links und Workspaces. Jede Änderung des Zustandes eines Workspaces oder eines Objektes einer visuellen Sprache wird an Dalis übermittelt.

Zwischen den drei Teilsystemen ist eine Schnittstelle implementiert, in der Daten über die vorgegebenen Kommunikationsprimitive create, modify und delete ausgetauscht werden können. Diese Primitive beziehen sich auf die Objekte, die während der inhaltlichen Arbeit entstehen, also auf Objektinstanzen der visuellen Sprachen, sowie auf Workspaces. Beispielsweise bewirkt das Erstellen eines Objektes einer visuellen Sprache, eine create-

Mitteilung an MatchMaker. Von dort wird diese Mitteilung an alle gekoppelten Applikationen sowie an Dalis weitergeleitet.

Auf Grundlage des Ist-Zustandes, der in der Datenbasis von Dalis vorliegt, und den aktuellen create-, modify- und delete-Messages können in Dalis Reaktionen auf Zustände bzw. deren Änderung implementiert werden. Beispielsweise können Operationen in den Applikationen aus einer domänenspezifischen Modellierung in Dalis angestoßen werden. Dies bildet auch den Hintergrund, warum Dalis später in die Diskussionsunterstützung eingebunden wird, denn so soll das Diskussionsmodell zur Verfügung gestellt werden. D.h. die eigentliche Modellierung, die auch zur Unterstützung des Diskussionsprozesses verwendet wird, findet in den Dalis-Komponenten statt. Das Diskussionsmodell und die damit verbundene operationale Semantik ist wie Dalis in Prolog implementiert.

Abbildung 6 erweitert Abbildung 5 um die Darstellung der Kommunikation mit Dalis. Während die Synchronisation der Applikationen alleine von MatchMaker organisiert wird, werden create-, modify- und delete-Informationen auch zwischen MatchMaker und Dalis ausgetauscht.

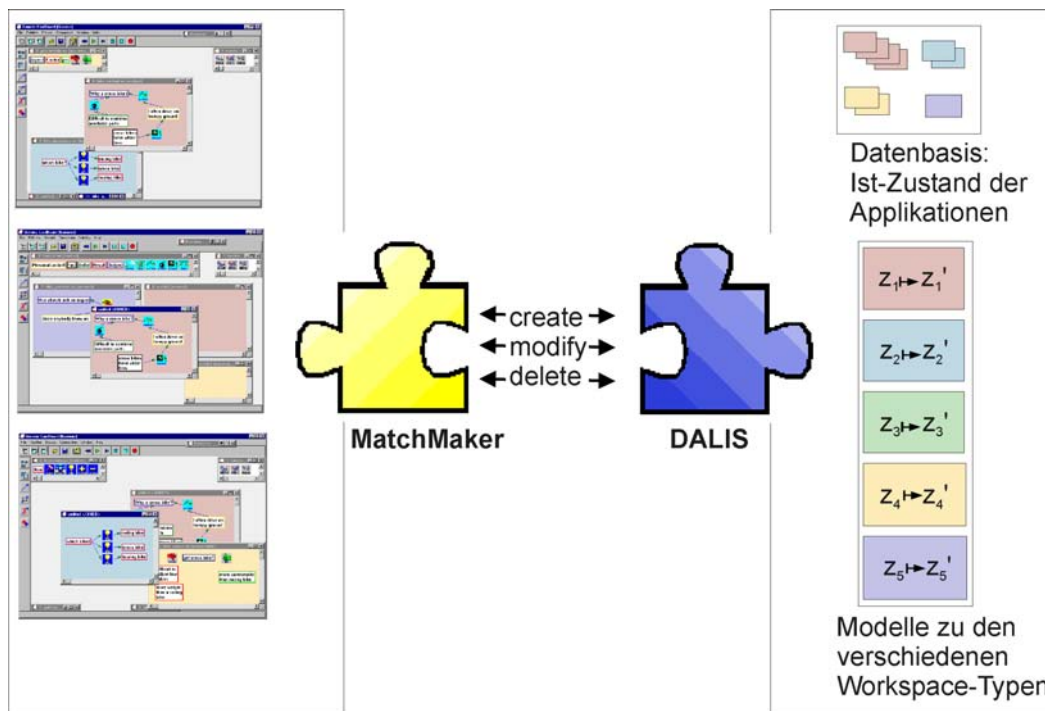


Abbildung 6

Dalis und MatchMaker kommunizieren über die standardisierten Primitiven create, modify und delete. In der Dalis-Datenbasis wird der Ist-Zustand der visualisierten Objekte abgelegt.

Die in Abbildung 6 angedeutete Dalis-seitige Modellierung ist bereits eine Vorausschau auf das Beispiel der Diskussionsunterstützung. Darin sollen die bunten Rechtecke oben rechts jeweils Gruppen von Datensätzen symbolisieren, die einen Workspace

beschreiben. So sind vier rosafarbene Rechtecke eingeblendet für die vier entsprechenden Workspaces der drei Applikationen auf der linken Seite der Abbildung. Die blauen, gelben und lilafarbenen Rechtecke symbolisieren die andere Workspaces der Applikationen. Die mit $z \rightarrow z'$ beschrifteten Rechtecke auf der rechten Seite symbolisieren Reaktionen auf den jeweiligen Zustand eines Workspaces.

3.1.3 CardBoard

Das System CardBoard wird durch seine Entwicklungsstadien hindurch in Tewissen (1996), Mühlenbrock, Tewissen und Hoppe (1997), Gaßner, Tewissen et al. (1998) und Hoppe, Gaßner et al. (2000) vorgestellt. Es folgt deshalb nur eine kurze Erläuterung.

Die Bezeichnung CardBoard bezieht sich auf das Rahmensystem, ohne eine bestimmte Einschränkung auf eine visuelle Sprache und damit auf eine Anwendung. Die spätere Spezialisierung auf die Diskussionsunterstützung wird mit der Bezeichnung DiscBoard abgegrenzt.

Das Rahmensystem CardBoard

Das System CardBoard (Abbildung 7) bildet den Rahmen für die Arbeit mit visuellen Sprachen, die man sich hier erst einmal als beschriftbare Karten und Verbindungen vorstellen kann. Die visuellen Sprachen liefern Objekte, die in Workspaces gelegt, strukturiert und bearbeitet werden.

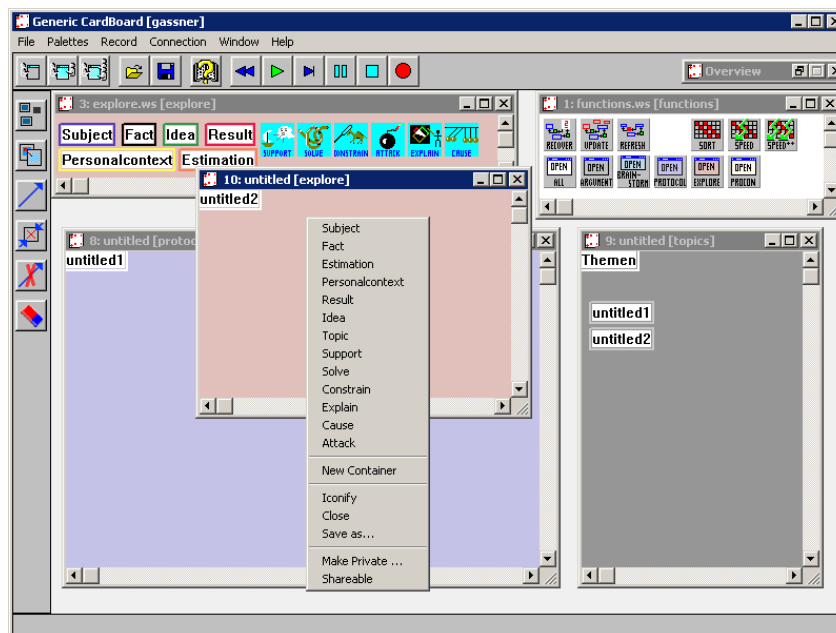


Abbildung 7

Das Rahmensystem CardBoard mit mehreren Workspaces, dem Bearbeitungs-Menü und zwei Paletten.

Die CardBoard-Oberfläche bietet zwei Standard-Paletten an und zentral einen freien Raum in dem Workspaces angelegt werden können (Abbildung 7). Über die obere, waagerechte Palette können neue Workspaces geöffnet werden (Tabelle 3). Außerdem können die Aktionen, die in den Workspaces ausgeführt werden, aufgezeichnet werden. Das CardBoard bietet dafür eine „Record&Replay“-Funktion an (Tabelle 3).



	Buttons zum Öffnen von Workspaces: „private“ (links), „shared“ (mittig), „joint“ (rechts)
	Das CardBoard bietet die Möglichkeit, die Ereignisse eines bestimmaren Workspaces aufzuzeichnen.

Tabelle 3

Funktionen des CardBoards zum Öffnen von Workspaces und zur Steuerung des „Record&Replay“.

Das CardBoard stellt drei Arten von Workspaces zur Verfügung:

- „private“ Workspaces, auf die keine weitere Person Zugriffsrechte hat,
- „shared“ Workspaces als Kommunikationskanäle zu einer konkreten Person bzw. Applikation,
- „joint“ Workspaces als Kommunikationskanäle zwischen einer Gruppe von Personen.

Während der Arbeit können im Prinzip beliebig viele „private“ oder „shared“ Workspaces geöffnet werden. Arbeiten mehrere Personen zusammen in einem Workspace, so können sie parallel Karten und Verbindungen anlegen, verändern oder löschen. Diese Eingaben sind sofort für alle Beteiligten sichtbar. Von diesen wird deshalb ein Mindestmaß an Koordination verlangt, so dass nicht eine Person Beträge anderer verhindert.

Die linke, senkrechte Palette bietet allgemeine Funktionen zur Bearbeitung der Objekte in den Workspaces an, also zur Bearbeitung der Elemente der visuellen Sprachen. Diese Funktionen sind unabhängig von der konkreten Sprache. Sie werden in Tabelle 4 näher erläutert.







	Einzelne Karten können einer Auswahl hinzugefügt werden.		Vorgesehen: Generieren ganzer Relationen.
	Es kann definiert werden, ob Karten zwischen Workspaces gleichen Typs verschoben oder kopiert werden sollen.		Vorgesehen: Löschen von Links.
	Einzelne Links können menügesteuert generiert werden.		Löschen von Karten.

Tabelle 4

Funktionen des CardBoards zur Bearbeitung der Elemente der visuellen Sprachen.

Parametrisierung durch visuelle Sprachen

Die visuellen Sprachen sind Parameter des Rahmensystems CardBoard, das die allgemeinen Funktionen zur Nutzung der visuellen Sprachen zur Verfügung stellt. Die Sprachobjekte sollen schnell und flexibel definierbar sein. Deshalb werden die Objekte der Sprache nicht systemintern definiert, sondern extern spezifiziert. So können auch Anwender Sprachen definieren, ohne das System selbst erweitern zu müssen. Das bezieht sich allerdings nur auf die rein syntaktische Definition der Sprachen.

In einer Initialisierungsdatei wird festgelegt, welche Sprachen geladen werden sollen. Abbildung 8a zeigt den Inhalt einer solchen Datei. Jede erwähnte Sprache referenziert die Datei, in der sie definiert ist. Jeder Sprache wird außerdem ein Name zugewiesen, der später verwendet wird, um Menüs zu generieren. Abbildung 8b zeigt ein Menü, das geöffnet wird, wenn ein neuer Workspace angefordert wird. Darin werden die Namen der Sprachen aufgenommen, die in der Initialisierungsdatei aufgeführt sind.

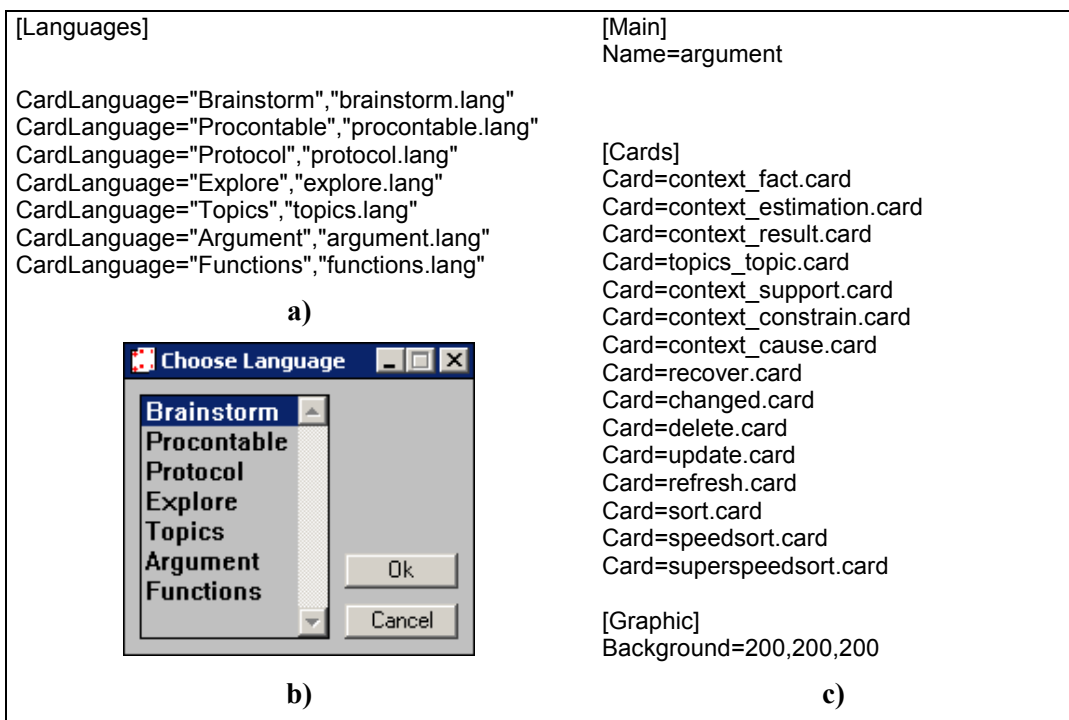


Abbildung 8

a) Das CardBoard wird durch extern definierte Sprachen parametrisiert. b) Über ein Menü wird ausgewählt, mit welcher visuellen Sprache gearbeitet werden soll. c) Die Definition einer visuellen Sprache in einer Datei.

Abbildung 8c zeigt die Definition einer visuellen Sprache. Diese Datei verweist wiederum auf die Definitionen der einzelnen verwendeten Kartentypen, die wieder in separaten Dateien definiert werden. Die Einträge des Menüs zum Erstellen von Karten in

einem Workspace werden auf Basis dieser externen Spezifikation generiert. So ein Menü ist in Abbildung 7 zu sehen.

3.1.4 Die visuellen Sprachen im CardBoard

Definition

Es wurde argumentiert, dass Netzstrukturen eine kognitionstheoretisch wie technisch geeignete Grundlage sind, um visuelle Sprachen für kognitive Tools einzusetzen. Im Rahmensystem CardBoard werden deshalb visuelle Sprachen zur Verfügung gestellt, mit denen Netzstrukturen erstellt werden können. Formal handelt es sich dabei um Graphen. Für die Nutzerinnen und Nutzer stellen die Schaubilder jedoch einfach Diagramme dar. Durch die Art und Weise ihrer Verwendung in kooperativen Arbeitsszenarien und deren flexible Bearbeitung wirken diese Diagramme als Medien in einem Kommunikationsprozess.

Die visuellen Sprachen bieten jeweils Mengen von Objekten als graphische Hilfsmittel an, um Diagramme zu erstellen. Die Primitive der visuellen Sprachen, die Objekte, werden auch als Karten bezeichnet, resultierend aus einer ursprünglich psychologischen Anwendung (Plötzner, Hoppe, Fehse, Nolte & Tewissen, 1996). Die Kartenmetapher erläutert aber gleichzeitig die Art des Einsatzes, nämlich kurze Notizen wie auf Karteikarten festzuhalten. Es existieren zwei Arten von Karten:

- **Inhaltskarten**, die als Container für freie Eingaben dienen und
- **Konnektorkarten**, mit denen die Inhaltskarten verbunden werden können.

Die angebotenen Karten werden auch als Kartentypen bezeichnet, deren Unterschied typischerweise auch in der graphischen Darstellung sichtbar ist.

Die Konnektorkarten sind Bestandteil einer Kante in einem Graphen, bzw. Bestandteil einer Relation zwischen Inhaltskarten. Um eine Kante zu vervollständigen, müssen neben einer Konnektorkarte noch die Verbindungen zu den Inhaltskarten bestimmt werden. Dafür werden in der Applikationsoberfläche zwischen einer Konnektorkarte und den in Relation stehenden Inhaltskarten Pfeile eingefügt: die Links. Durch die Stelligkeit der Relationen inklusive der Richtung der Links wird die Struktur der Graphen und damit die Syntax der Sprache bestimmt, wobei prinzipiell n-stellige Relationen dargestellt werden können (entsprechend dem Konzept eines Hypergraphen). Die Interaktion zur Erstellung von Links wird über die Konnektorkarten gesteuert. Diese definieren Anzahl, Typ und Richtung der Links für einen Kartentyp.

Über den Typ einer Karte oder auch den eines Links können den jeweiligen Objekten bestimmte operationale Semantiken zugeordnet werden.

Sowohl die Inhaltskarten als auch die Konnektorkarten sind so definierbar, dass sie während der Kommunikationsprozesse editierbar sind.

Die Spezifikationen der Kartentypen befinden sich in externen Files, die beim Start des Rahmensystems CardBoard geladen werden. Im vorangegangenen Abschnitt wurde bereits erläutert, wie die Zugehörigkeit der Sprachprimitive zu einer visuellen Sprache festgelegt wird. Abbildung 9a zeigt nun ein Beispiel wie eine Karte, in diesem Fall eine Konnektorkarte, definiert wird.

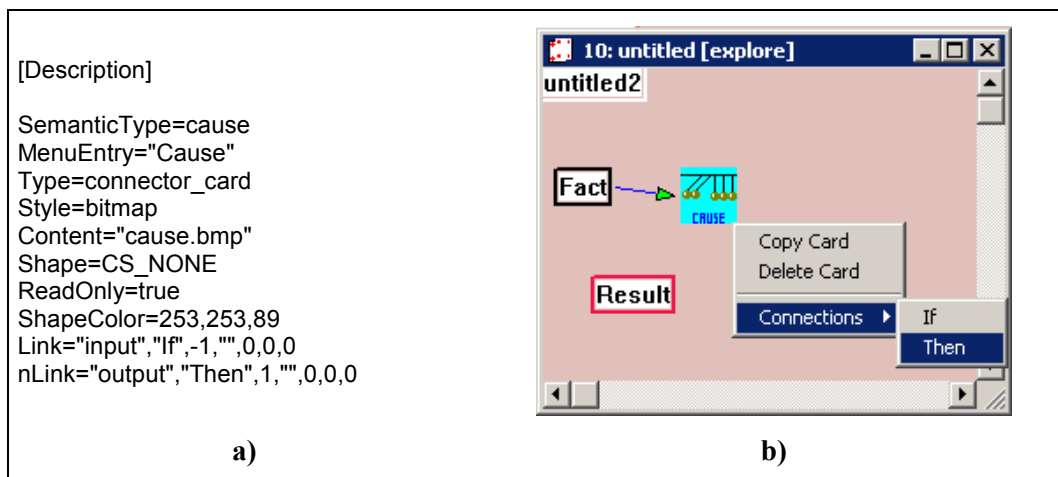


Abbildung 9

- a) Die Spezifikation einer Karte wie sie in einer Datei durchgeführt wird.
b) Interaktion mit einer Konnektorkarte zur Erstellung von Links.

Über „Tags“ werden dort Typ, Aussehen und Bestandteile der Karte festgelegt und es findet sich dort auch die Unterscheidung der Konnektor- und Inhaltskarten wieder: Mit Hilfe des „Type-Tags“ wird zwischen „connector_card“ und „content_card“ unterschieden. Mit dem „Style-Tag“ wird definiert, wie der Inhalt des „Content-Tags“ zu interpretieren ist. Mit dem „Content-Tag“ kann sowohl auf eine Datei verwiesen werden, aus der der Inhalt geladen werden soll, als auch direkt ein Inhalt bestimmt werden. Über den Verweis auf eine Datei ist es möglich, Graphiken einzubinden, und zur Gestaltung von Symbolen zu verwenden. Das „Content-Tag“ könnte auch auf dynamische Komponenten verweisen, die eine spezifische Interaktion anbieten. Die Spezifikation einer Konnektorkarte beinhaltet auch die Definitionen für die Links („Link-Tag“), bestehend aus dem Namen des Links, seinem Eintrag im kontextsensitiven Popup-Menü der Konnektorkarte, seiner Richtung in Bezug auf die Konnektorkarte, dem Label des Links und seiner Farbe. Im gezeigten Beispiel sind keine Label angegeben. „Shape“ und „ShapeColor“ bestimmen das Aussehen der Karten. Insbesondere bei Inhaltskarten bilden sie die wesentlichen Merkmale an denen man den Kartentyp in der graphischen Darstellung erkennen kann.

Das „ReadOnly-Tag“ und das „Movable-Tag“ bestimmen Interaktionsmöglichkeiten auf Ebene des Interfaces. Das „SemantikType-Tag“ wird für die Zuordnung der operationalen Semantik verwendet, die von Dalis gesteuert wird.

Abbildung 9b zeigt die Interaktion mit einer Konnektorkarte, um Verbindungen zu Inhaltskarten herzustellen. Dafür bietet eine Konnektorkarte, hier die „Cause“-Karte, ein Menü an, unter dem der Eintrag „Connections“ zu den möglichen Verbindungstypen überleitet. In dem dargestellten Beispiel werden die Verbindungen „If“ und „Then“ angeboten. Mit der Wahl des Verbindungstypen wird automatisch auch die Richtung des Links zu der betroffenen Inhaltskarte festgelegt. In der Abbildung sind „Fact“ und „Result“ die zu verbindenden Inhaltskarten, deren Inhalt hier noch nicht weiter spezifiziert wurde. In der Abbildung sind beide Inhaltskarten in der Form dargestellt, in der sie generiert werden. Der „default“-Text stellt eine zusätzliche Hilfe dar, um den Typ der Karte zu erkennen. Im vorliegenden Ansatz werden mit visuellen Sprachen immer semi-formale Repräsentationen erstellt. Ein wesentliches Kriterium semi-formaler Repräsentationen besteht in der Mixtur von Computer-interpretierbaren und nicht – interpretierbaren Bestandteilen eines Ausdrucks ([dynagloss1], [dynagloss2]). Ein typisches Beispiel dafür sind Hypertexte, bei denen die Links formal sind und interpretiert werden, die Inhalte aber nicht. Auch E-Mails sind ein Beispiel für semi-formale Repräsentationen. Normalerweise haben Nutzerinnen und Nutzer keinen Einfluss auf die Interpretation der formalen Bestandteile einer Darstellung, wohl aber auf die Inhalte, also die nicht-formalen Bestandteile. Der Vorteil von semi-formalen Darstellungen besteht gerade darin, individuelle nicht formalisierte Inhaltseingaben mit formalen interpretierbaren Elementen zu verbinden.

Über die Darstellungen mit visuellen Sprachen kann auf diverse strukturorientierte und kartentyporientierte Merkmale, wie Art von Verbindungen und Muster sowie auf Ort, Nähe und Anzahl und von Karten und Verbindungen zugegriffen werden. Die so gewonnenen Daten bilden den computerinterpretierbaren bzw. formalen Bestandteil der Repräsentation. Diese Daten können genauso für Interpretationen ausgenutzt werden wie sie Auslöser für Operationen sein können. Gleichzeitig können die unformalisierten Inhalte frei mit dem System bearbeitet werden.

Die Kartenhierarchie im CardBoard

Eine Entscheidung beim CardBoard-System bestand darin, nicht nur die Inhaltskarten, sondern auch die Konnektorkarten als Karten zu implementieren. Dies sollte zu einer flexiblen Möglichkeit bei deren Gestaltung führen. Außerdem waren so Hyperlinks

graphisch gut umsetzbar. In der CardBoard-Implementierung werden entsprechend sowohl die Inhaltskarten als auch die Konnektorkarten von der gleichen Klasse abgeleitet (Abbildung 10). Die CardBoard-Implementierung stellt als übergeordnete Klasse die „BaseCard“-Klasse zu Verfügung. Davon abgeleitet wird sowohl die „ContentCard“-Klasse als auch die „ConnectorCard“-Klasse. Weiter verfeinerte Ausprägungen werden in abgeleiteten Klassen definiert.

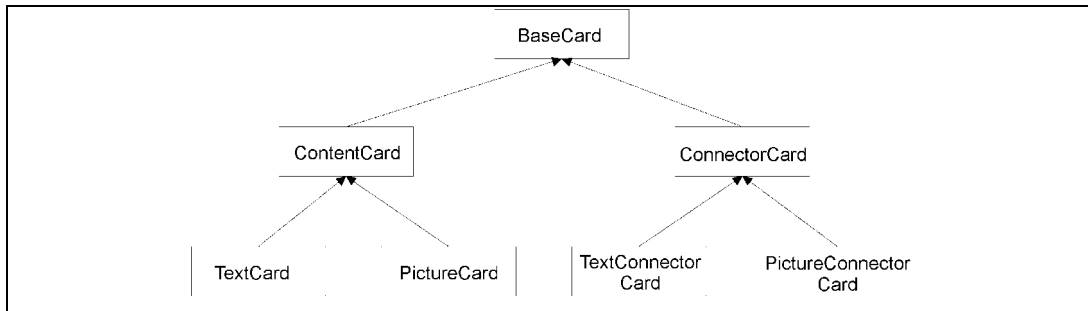


Abbildung 10

Die Klassenhierarchie der Karten im CardBoard.

Die „BaseCard“-Klasse stellt allgemeine Funktionen bereit, die die Interpretation der externen Kartenspezifikationen leisten und z.B. Drag&Drop und delete-Aktionen steuern. Die Funktion der „ContentCard“-Klasse und der „ConnectorCard“-Klasse besteht darin, auf Basis der in der Spezifikation referenzierten oder definierten Inhalte das jeweilige Oberflächenobjekt aufzubauen. Dazu gehört auch, die notwendigen Popup-Menüs für die Karten zu generieren und auf die spätere Interaktion zu reagieren. Das gilt insbesondere für die Konnektorkarte, für die die „ConnectorCard“-Klasse den Umgang mit den Links implementiert.

3.2 Systeme im Vergleich

Obwohl die „face-to-face“-Kooperation nach wie vor die üblichste Art der Zusammenarbeit ist (Lernen in Klassen, Meetings, Debatten, etc.), existieren heute wenige Systeme, die explizit für eine Diskussionsunterstützung in einem „face-to-face“-Szenario implementiert sind, die also versuchen Computer in dieses Szenario zu integrieren.

Anders als in verteilten Szenarios wird in diesem Kontext direkt deutlich, dass die Art der Hilfestellung durch Software auf die Externalisierung, Repräsentation, (Re-)Strukturierung und Erstellung von Artefakten abzielen muss und nicht darauf, die Kommunikation selbst zu ermöglichen. Das bedeutet, dass die Darstellungsmittel und Arbeitsmethoden eine entscheidende Rolle bei der Diskussionsunterstützung tragen.

Eine andere These dieser Arbeit besteht darin, dass die nachhaltige Nutzung von Diskussionsinhalten nicht ausreichend in die Systemgestaltungen mit einfließt. Dies ist in sofern fatal, als dass aus dem Kontext des Wissensmanagements abgeleitet wird, dass die nachhaltige Dokumentation einen wesentlichen Mehrwert bei der Nutzung einer Diskussionsunterstützung darstellt. Durch die nachhaltige Nutzung soll Wissen re-integriert werden, „neues“ Wissen entstehen und „communities“ aufgebaut werden.

Insgesamt werden dem gemäß drei Kriterien für den folgenden Systemvergleich ausgewählt:

- Nutzbarkeit im „**face-to-face**“-**Setting**.
- Verwendung von **visuellen Sprachen**, durch die Diskussionen durch kreative oder vielfältige Ausdrucks- und Darstellungsmöglichkeiten angereichert werden oder prozessorientierte Unterstützung bieten.
- **Nachhaltige Nutzung**, Wissensbildung.

Für den Vergleich werden exemplarisch Systeme herangezogen, die sich mindestens einem dieser Kriterien zuordnen lassen. Es zeigt sich, dass kaum ein System alle Aspekte auf sich vereinigt.

In Kapitel 1.5 wurden bereits drei Ebenen erläutert, auf denen „face-to-face“-Szenarios technisch unterstützt werden können: 1) Software/Tools 2) Realisierung des Szenarios 3) Organisatorische Einbettung. Diese drei Ebenen sollen im folgenden auch für den Systemvergleich herangezogen werden und können außerdem den oben genannten Kriterien zugeordnet werden: Die Beschreibungen der Tools werden jeweils auch die Nutzung der visuellen Sprachen verdeutlichen, zum Aspekt Realisierung des Szenarios wird neben technischen Ansätzen auch das Nutzungsszenario beschrieben und unter dem Aspekt organisatorische Einbettung wird die Art der Nutzung und Einbettung in Arbeitsabläufe geschildert. Die Systeme werden in alphabetischer Reihenfolge vorgestellt. Die Einordnung gemäß der „Zielrichtung“ bezieht sich auf die in Kapitel 2 eingeführten Verwendungsarten von visuellen Sprachen.

3.2.1 Belvedere

- *Zielrichtung.* Lernunterstützung; Kommunikationsmedien und Wissensprodukte
 - *Schwerpunkt.* Vermittlung wissenschaftlicher Argumentationsweise
 - *Kriterien.* Nutzung visueller Sprachen, Verwendung im „face-to-face“-Szenario
- Software/Tool.** Das Ziel des Belvedere-Systems (Suthers, Weiner et al., 1995; Suthers, Toth & Weiner, 1997; siehe Abbildung 11) besteht darin, Schülerinnen und Schülern wissenschaftliches Argumentieren beizubringen.

Die Motivation besteht darin, dass bei jungen Studierenden häufig Probleme beobachtet werden können, wichtige Elemente aus einer umfangreichen Debatte herauszugreifen und relevante Argumentationskriterien zu finden. Hinzu kommt ungenügendes Grundlagenwissen in den diskutierten Bereichen und außerdem eine nicht ausgereifte Motivation, einen Inhaltsbereich zu diskutieren. Das Lernziel besteht darin, bestimmte Regeln der Argumentation zu vermitteln:

- Hypothesen erklären Daten
- Hypothesen werden ohne Daten nicht akzeptiert
- Mehrere Belege, die auf eine Hypothese hinführen sind besser als ein einzelnes gutes Datum
- Man soll positive aber auch negative Belege finden
- Man muss zusätzliche Belege finden, wenn zwei Hypothesen durch die gleichen Belege unterstützt werden.

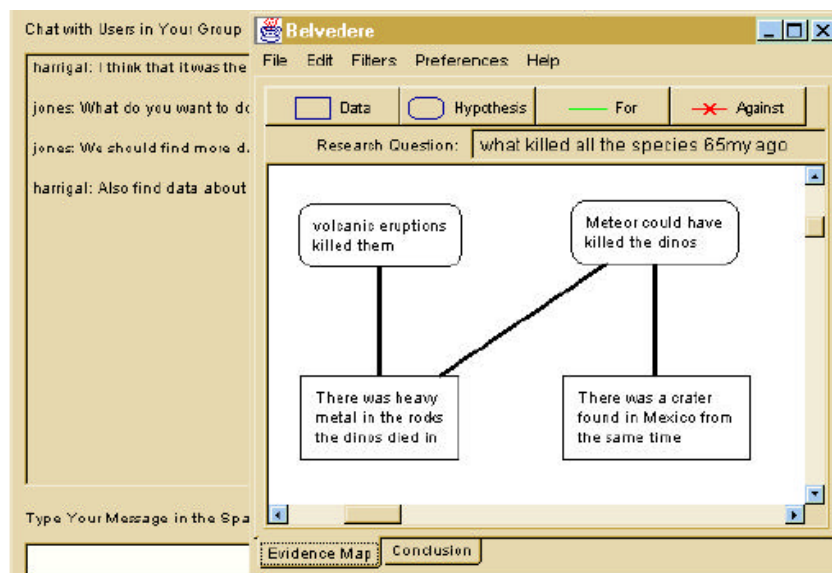


Abbildung 11

Ein „inquiry diagram“ im Belvedere-System (Suthers, 1999C).

Das Arbeitsprodukt bei Belvedere ist ein Diagramm. Eine visuelle Sprache stellt dafür Knoten und Relationen zur Verfügung, wobei sich die Knoten über ihre Form unterscheiden. Sie stellen Hypothesen oder Daten dar, die durch die Relationen zueinander in Verbindung gesetzt werden. Den Relationen können durch die Dicke der Linien sogenannte „belief levels“ zugeordnet werden. Wenn Diagramme erstellt werden, wird erwartet, dass die oben formulierten Regeln möglichst ausgeschöpft werden. Dies wird durch zwei Agenten angeleitet und unterstützt, die diese Regeln überprüfen und von Zeit zu Zeit Hilfestellungen dazu geben.

Das Belvedere-System hat bereits mehrere Entwicklungsstufen hinter sich. In der Version, die in Suthers, Toth und Weiner (1997) vorgestellt wurde, wurde vor allem auf einen großen Umfang der visuellen Sprache verzichtet. Gab es vorher drei Arten,

Diagramme aufzubauen, so gibt es darin nur noch eine. Das beruht auf Erfahrungen nach denen sich die Bedeutung der einzelnen Primitive häufig überlagerte und deshalb zu Ungenauigkeiten führte.

Realisierung des Szenarios. Belvedere 2.x (Suthers, Toth & Weiner, 1997, Suthers 1999C) stellt „shared“ Workspaces für die Diagramme und ein „Chat“ zur unstrukturierten Diskussion zur Verfügung. Trotz dieser scheinbaren Ausrichtung auf die örtlich verteilte Situation, ist Belvedere für den Klassenraum gedacht. Dabei arbeiten zwei oder mehr Personen miteinander, erstellen zusammen in den Workspaces ihre Diagramme, interagieren aber zusätzlich mündlich.

Organisatorische Einbettung. Der Schwerpunkt von Belvedere liegt bei der kooperativen Aufarbeitung und Ko-Konstruktion von Informationen in Form von wissenschaftlichen Argumentationsnetzen. Im Klassenraumszenario sollen weitgehend reale wissenschaftliche Probleme ergründet werden. Dazu besteht die Möglichkeit externe Materialien zu referenzieren. Auch Experimente und Analysen sollen mit in den Lernablauf integriert werden auch wenn sie nicht von dem Belvedere-System selbst begleitet werden, dass nur ein Ausschnitt des Lernens begleiten soll.

Das Belvedere-System wird vor allem dazu herangezogen, um zu untersuchen, ob und inwieweit Repräsentationen geeignet die Kommunikation begleiten können („representational guidance“, „salience“: Suthers, 1999A, 1999B, 1999C). Trotzdem wird ein technisch wie didaktisch relativ komfortables Szenario erstellt.

Weniger Anstrengungen werden darauf investiert, die jeweiligen Ergebnisse weiter zu verwerten, sie beispielsweise als Diskussionsanreiz anderen Gruppen zur Verfügung zu stellen oder beste Ergebnisse vorzuhalten. Entsprechend wird die Suche nach ergänzenden Diagrammen oder deren Integration nicht unmittelbar unterstützt oder angestrebt. Belvedere ist deshalb nicht unbedingt für länger andauernde Entwicklungen umfangreicherer Sachzusammenhänge zu verwenden. Auch bleibt unklar, wie mit größeren Diagrammen, die komplexere Domänen aufarbeiten, umgegangen wird.

3.2.2 Chips

- *Zielrichtung.* Kommunikationsmedien und Wissensprodukte
- *Schwerpunkt.* Unterstützung von Gruppenarbeit
- *Kriterien.* Nutzung visueller Sprachen, Verwendung im „face-to-face“-Szenario, Nachhaltigkeit

Software/Tool. Bei dem Chips-System (Wang & Haake, 2000; Abbildung 12) handelt es sich um ein kooperatives Hypermediasystem, dass begleitend zu Kommunikations-

situationen eingesetzt werden soll. Es baut teilweise auf dem Sepia-System auf (wird im Folgenden erläutert), erweitert dieses jedoch maßgeblich und folgt deutlich unterschiedlichen Konzepten. Chips soll genauso bezüglich individueller Anforderungen adaptierbar sein, wie in Bezug auf die Anforderungen einer Gruppe. Sowohl der gemeinsame Kommunikationsbereich („shared information space“) als auch Prozesse sollen definiert oder angepasst werden können. Damit sollen vier zentrale Anforderungen erfüllt werden: 1) Design des gemeinsamen Kommunikationsbereiches 2) Anpassung der Kooperationsmittel 3) Zugriffskontrolle 4) Anpassung des gemeinsamen Userinterfaces.

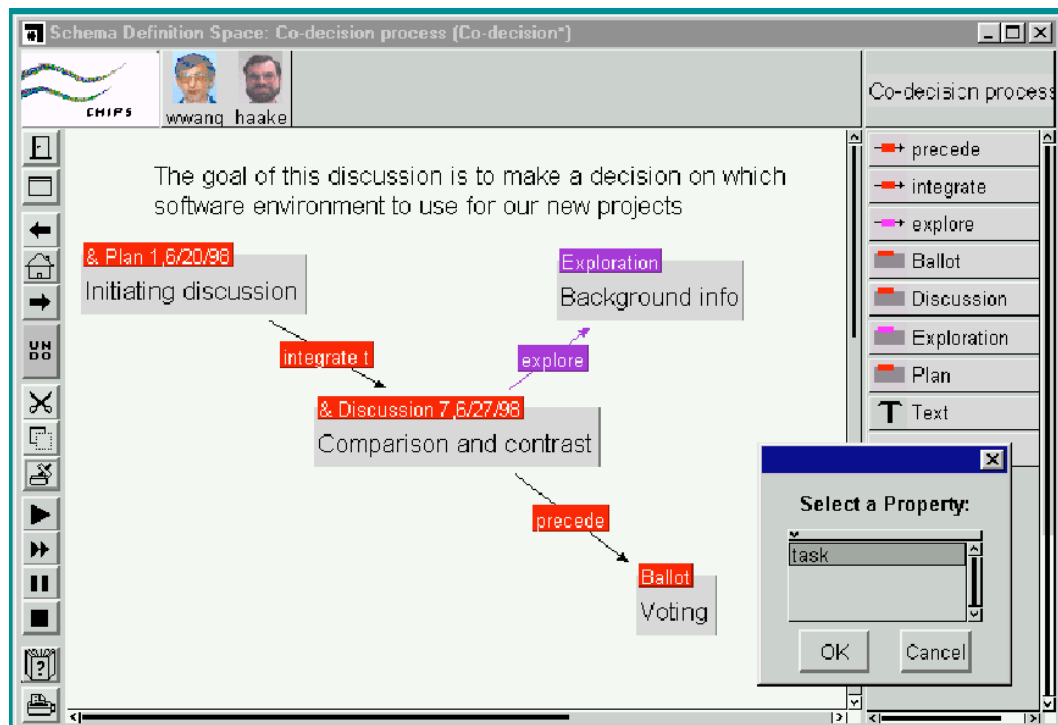


Abbildung 12

Darstellung eines Workflows in Chips. Rechts: Palette mit den Objekten der visuellen Sprache. Links: Allgemeine Funktionen.

„Activity spaces“ zeigen die einzelnen potentiell vernetzten Seiten an und stellen das adaptierbare Interface zur Verfügung. Die Seiten enthalten wiederum Knoten, Links und Multimedia-Objekte, wobei die Links an Knoten gebunden sind und auf andere Seiten verweisen: Organisationale Links unterliegen strukturellen Constraints, referenzielle Links entsprechen allgemeinen Hyperlinks. Zusätzlich können Relationen zwischen Knoten einer Seite dargestellt werden.

Einem „activity space“ wird ein Userinterface zugeordnet, über das Kontroll- und Browserfunktionen zugänglich sind und über das die Kooperationsparameter während einer Session festgelegt werden können. Organisationseinheiten (z.B. Personen oder Gruppen) können Rollen zugeordnet werden, an die Zugriffsrechte gebunden sind.

Auch Prozesse werden in den „activity spaces“ dargestellt bzw. definiert (siehe Abbildung 12). Dafür können beispielsweise Aufgabenknoten über Links verbunden werden, die um Attribute für Zeit und Transitionen erweitert werden.

Anpassbar sind die Interfaces, die eine visuelle Sprache zur Verfügung stellen, indem neue Objekte durch User definiert werden. Diese werden ebenfalls in die Palette mit aufgenommen, um so Vorlagen für bestimmte Arbeitsweisen anzulegen, die später wieder als Templates verwendet werden können. Weiterhin sind die Objekte selbst durch User anpassbar, indem ihre Attribute geändert werden.

Realisierung des Szenarios. Chips soll sowohl in synchronen wie in asynchronen, in verteilten wie „face-to-face“-Situationen Unterstützung liefern.

Organisatorische Einbettung. Ähnlich wie Sepia scheint Chips ein weitgehend geschlossenes System zu sein. Schnittstellen zu anderen „Welten“ wie beispielsweise zum Internet, XML und XSLT werden zumindest in Wang und Haake (2000) nicht thematisiert. Chips scheint deshalb in erster Linie als wissenschaftliche Versuchsumgebung entwickelt worden zu sein.

Durch die Zugriffskontrolle und die Definitionsmöglichkeiten in Bezug auf die Gruppenzusammensetzung sind komplexe Organisationsstrukturen bestimmbar. Undeutlich bleibt jedoch, wie die Erstellung der Navigationsnetze über kleine Teilaspekte hinaus unterstützt wird, inwieweit also eine Unterstützung im Sinne eines „corporate memory“ vorgesehen ist.

Die Anpassung der Interfaces an die jeweiligen Bedürfnisse ist weitgehend, gleiches gilt auch für die Prozessdefinitionen. Es ergibt sich eine anpassbare, domänenunabhängige Arbeitsumgebung. In Bezug auf eine „computational semantic“ der Objekte sind die User allerdings weitgehend auf ein Entwicklerteam angewiesen, das gewünschte Operationen implementiert.

Die Prozessunterstützung kann auch an einen Workflow bezüglich der Inhaltsknoten gebunden werden. Auch darin hebt sich dieses Tool von anderen Entwicklungen ab. Ein wenig fraglich erscheint es, ob Projektplanungsaufgaben nicht teilweise sinnvoller in tabellarischer Weise dargestellt werden könnten, als in Graphen.

3.2.3 CoLab: Cognoter/Argnoter

- *Zielrichtung.* Kommunikationsmedien und Wissensprodukte
- *Schwerpunkt.* Cognoter: Kooperatives Vorbereiten von Präsentationen; Argnoter: Argumentation zu konkurrierenden Vorschlägen
- *Kriterien.* Nutzung visueller Sprachen, Verwendung im „face-to-face“-Szenario

Software/Tool. Argnoter und Cognoter (Stefik et al., 1987) wurden als Tools des CoLab („collaborative laboratory“) vorgestellt, das bei XEROX PARC entwickelt und realisiert wurde (siehe Abbildung 13).



Abbildung 13

Aufnahme einer CoLab-Installation [CoLab].

Cognoter: Beim Vorbereiten von Präsentationen wird davon ausgegangen, dass zuerst Ideen gesammelt werden (Brainstorming), die dann sortiert (Organisation) und überarbeitet (Evaluation) werden müssen. Die Grundideen für eine Präsentation werden in möglichst kurzen Beiträgen als Knoten von Graphen notiert. Für diese Phase gibt es hauptsächlich Verhaltenshinweise, die Bezug nehmen auf die Brainstormingmethode. In der Organisationsphase werden gerichtete Kanten angelegt, um eine Reihenfolge zu definieren, in der die Aspekte präsentiert werden sollen. Auch Gruppierungen sind möglich, in denen verschiedene Punkte unter einem Aspekt zusammengefasst werden. Jede Ideengruppe wird in einem eigenen Fenster angezeigt. Bei der Evaluation wird die Struktur der Beiträge überarbeitet und ergänzt.

Argnoter: Vorbereitete Vorschläge z.B. für Designs oder Programmwürfe werden von Mitgliedern einer Arbeitsgruppe eingebracht. Die Vorbereitung gehört zu den „Spielregeln“ bei der Verwendung des Tools. Es wurde festgestellt, dass in so vorbereiteten Sitzungen stärker diskutiert wurde, da die Vorschläge oft divergieren und so die Kontroversen eher in den Vordergrund treten. In der Argumentationsphase sollen die einzelnen Vorschläge durch Pro- und Kontra-Argumente abgewogen werden. Dies führt dann direkt zu einer Evaluationsphase, in der den Argumenten „beliefs“ zugeordnet werden: Eine Aussage kann von einer Person geglaubt werden, von einer anderen nicht. Daraus werden „belief sets“ gebildet, die bestimmten Standpunkten oder Ansichten zugeordnet werden können.

Realisierung des Szenarios. Wie in Abbildung 13 zu erkennen ist, gehören mehrere vernetzte Computer und eine elektronische Tafel zu diesem Szenario. Das Labor ist für die „face-to-face“-Situation ausgelegt. Gearbeitet wird in „shared“ Workspaces, die über eine Datenbank synchronisiert werden. Technisch führte das zu Problemen bei der parallelen Eingabe von Text.

Organisatorische Einbettung. Die organisatorische Einbettung besteht in Form der „Spielregeln“, die besagen wie die Tools verwendet werden. Insbesondere die Argnoter-Methode wurde ebenso auch ohne Tool nur mit Hilfe eines „schwarzen Brettes“ praktiziert. Andere Aspekte wie beispielsweise Dokumentationen und Retrieval werden nicht problematisiert.

3.2.4 Csile

- *Zielrichtung.* Lernunterstützung; Kommunikationsmedien und Wissensprodukte
- *Schwerpunkt.* Kritische Wissenskonstruktion beim Lernen
- *Kriterien.* Nachhaltigkeit

Software/Tool. Csile (computer supported intentional learning environments) (Scardamalia & Bereiter, 1992; Scardamalia, Bereiter, Brett et al., 1992) wurde als Lernsystem für Schülergruppen entwickelt.

Die Gruppe soll mit dem System Wissen konstruieren und dabei kritisch hinterfragen. Die Arbeitsweise besteht darin, dass einzelne Personen Notizen verfassen, wie sie in Abbildung 14 zu sehen sind. Zu jeder Notiz können Grafiken oder weitere Notizen annotiert werden, allerdings können nur die Autoren bzw. Autorinnen einer Notiz diese ändern oder löschen. Sie werden jedoch benachrichtigt, wenn ein Beitrag kommentiert wird.

Für Csile 2.0 (Scardamalia & Bereiter, 1992) wurde eine Teilung in thematische Arbeitsbereiche („thematic spaces“) und Arbeitsumgebungen („environments“) vorgesehen. Erstere lassen eine Sortierung nach Themen zu, so dass Beiträge zu bestimmten Themen assoziierbar sind. Die Arbeitsumgebungen (Erklärungen, Funktionsweisen, Exploration, Bedeutungen) bieten bestimmte Sichten an und stellen dafür bestimmte Tools zur Verfügung. Bei der Frage nach Funktionsweisen wäre es beispielsweise sinnvoll Animationen zu integrieren. Bei der Darstellung von Bedeutungen sollen später „Concept Maps“ erstellt werden.

Das System soll außerdem Automatismen und Hilfen zur Verfügung stellen, wie beispielsweise eine Rechtschreibprüfung für Schlüsselwörter und Hinweise auf angrenzende Beiträge.

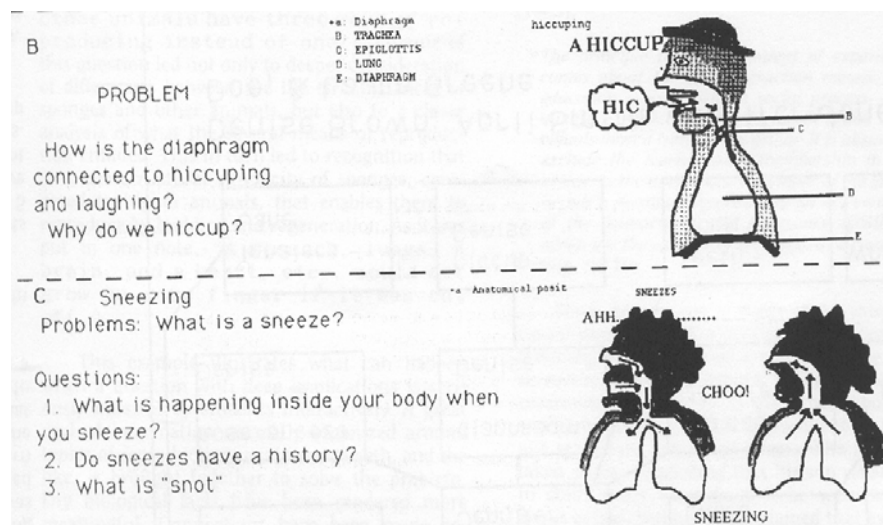


Abbildung 14

Skizzen und Annotationen im Csile-System (Scardamalia, Bereiter, Brett, et al., 1992).

Realisierung des Szenarios. Entwickelt wurde das System für eine asynchrone und verteilte Arbeitssituation. Die Notizen - Graphiken und textuelle Eingaben – werden zu einer zentralen Datenbank hinzugefügt und können von anderen annotiert werden. Diese Kooperation kann sowohl unter Schülerinnen und Schülern einer Klasse stattfinden, wie zwischen denen einer Schule, also auch mit unterschiedlichem Lernniveau und auch außerhalb des Schul-kontextes.

Durch dieses Vorgehen sollen Schülerinnen und Schüler dazu animiert werden, über die Inhalte, ohne denn Druck einer mündlichen Situation, zu reflektieren und durch die verteilte Situation sollen sie lernen, unabhängig eigene Gedanken zu entwickeln.

Organisatorische Einbettung. Der Csile-Ansatz ist ein kooperativer Ansatz zur Wissenskonstruktion. Durch ihn soll es den Lernenden, im Gegensatz zu traditionellen Lehrmethoden, erleichtert werden, eigenen Fragestellungen zu folgen („intentional learning“; Scardamalia & Bereiter, 1994). Scardamalia und Bereiter (1994) bezeichnen diesen Lernprozess als „second-order“, da man sich nicht asymptotisch einem Lernziel nähert, sondern sich die Lernziele während des Lernens ändern können sowie das Niveau der Ziele.

Angestrebt wird ein Prozess, der mit dem eines rezensierten Journals vergleichbar ist. Dieser Prozess wird zwar nicht erzwungen aber doch durch die Art der möglichen Eingaben sehr nahe gelegt. Aus dem Vergleich zum akademischen Publizieren werden auch intrinsische Motivationen abgeleitet, mit solch einem System zu arbeiten, dem Wunsch nach Anerkennung, dem Wunsch, etwas zu beeinflussen und dem Wunsch an relevanten Diskursen teilzuhaben.

Eine Prozessunterstützung findet statt in Anlehnung an diesen Publikations- und Rezensionsprozess. Zum einen müssen Beiträge „veröffentlicht“ werden. Die Lehrerin oder der Lehrer oder eine Gruppe entscheidet darüber, welche Beiträge vom „zu publizieren“ zum „publiziert“-Status wechseln. Um zu veröffentlichen, muss für den Beitrag formuliert werden, worin seine besondere Aussage liegt.

Weiterhin werden Autorinnen und Autorinnen darüber benachrichtigt, wenn auf sie Bezug genommen wurden. Das heißt, dass nicht nur veröffentlichte Beiträge von anderen gelesen werden können, diese aber einen besonderen Grad der Ausarbeitung erreicht haben.

Das Szenario und die verwendete Datenbanktechnologie ist hier weit mehr als eine zufällige Zusammenführung, die die Kooperation ermöglichen soll. Die Verteiltheit der Personen wird didaktisch motiviert (verbesserte Reflexion, unabhängige Gedanken, Eigeninitiative). In der Verwendung einer Datenbank wird ein Vorteil zum Einsatz eines reinen Hypermediasystems gesehen, in dem alle Verknüpfungen zwischen Materialien explizit über Links dargestellt sind. Mit der Datenbank können über Suchen dagegen immer neue Suchkriterien aufgestellt und so immer neue Verbindungen gefunden werden.

3.2.5 Dolphin

- *Schwerpunkt.* Kommunikationsmedien und Wissensprodukte
 - *Zielstellung.* Gesprächsunterstützung
 - *Kriterien.* Verwendung im „face-to-face“-Szenario, Nutzung visueller Sprachen
- Software/Tool.** Das Dolphin-System (Streitz, Geißler et al., 1998) setzt auf der Sepia-Architektur (wird nachfolgend ebenfalls erläutert) auf und adaptiert diese für Besprechungssituationen („face-to-face“).

In einem Dolphin-Workspace (Abbildung 15, rechts) können Bilder, Texteingaben und handschriftliche Eingaben kombiniert werden. Zur Strukturierung werden Knoten angeboten, die Inhalte zu einem Objekt zusammenfassen. Sie werden einfach durch das handschriftliche Umranden von Inhalten erstellt und erscheinen daraufhin in einer grau schattierten Box.

Es existieren zwei Möglichkeiten, um Knoten zueinander in Relation zu setzen: Einerseits können Links in einem Workspace erstellt werden, andererseits können Knoten in andere Knoten gelegt werden, so dass eine „dreidimensionale“ Struktur entsteht. Aus einer abstrakteren Sicht auf die Knoten können auch Relationen zwischen Knoten unterschiedlicher „Ebenen“ angelegt werden. Die Knoten in Dolphin sind zwar nicht typisiert aber es ist möglich, Dokumente aus dem Sepia-System zu importieren, die

getypte Knoten enthalten. Gleichmaßen können Dolphin-Dokumente für Sepia exportiert werden.

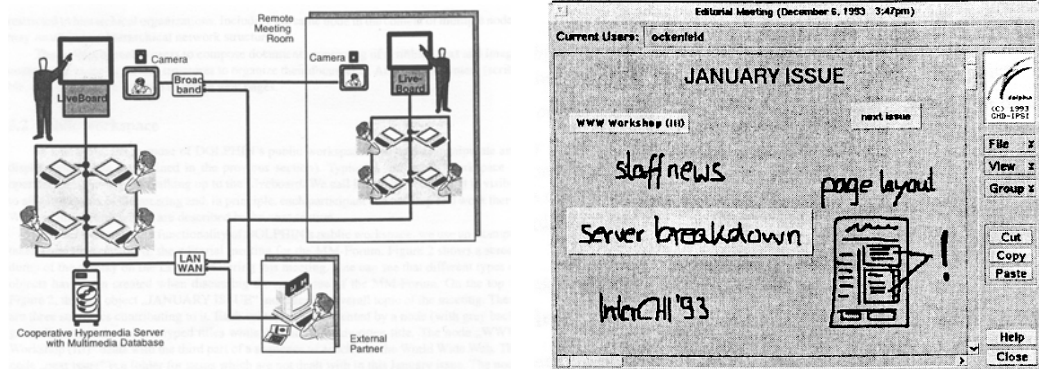


Abbildung 15

Links: Das Dolphin-Szenario; Rechts: Ein Workspace im Dolphin-System.

Realisierung des Szenarios. Mit Dolphin soll nicht nur das „face-to-face“-Szenario unterstützt werden, sondern es soll explizit ein nahtloser Übergang zwischen unterschiedlichen Arbeitsszenarien möglich sein. Dazu gehört die direkte Arbeit an der elektronischen Tafel, die „face-to-face“-Gruppenarbeit in einem Raum mit vernetzten Rechnern für die Beteiligten und einer elektronischen Tafel sowie die Möglichkeit, externe Personen bzw. Computer einzubinden oder sogar weitere Arbeitsgruppen (Abbildung 15, links).

Durch die Interoperation mit Sepia, das auch asynchrones Abreiten unterstützt, werden asynchrone Arbeitsphasen stärker mit den synchronen integriert.

Organisatorische Einbettung. Es wird davon ausgegangen, dass „meeting support“ die Vielzahl von Aktivitäten, beginnend mit einer Agenda und Entwürfen, Ideen und Notizen, hin zu Arbeitsplänen und Dokumenten unterstützen muss. Dies wird jedoch nur teilweise und implizit erreicht, z.B. indem vorbereitete Inhalte mit in die Diskussion eingebracht werden können. Als Mittel stehen dafür öffentliche und private Workspaces zur Verfügung zwischen denen Inhalte transferiert werden können. Strukturierte Inhalte können in Sepia entworfen werden.

3.2.6 gIBIS

- *Schwerpunkt.* Kommunikationsmedien und Wissensprodukte
- *Zielstellung.* Design rationale
- *Kriterien.* Nutzung visueller Sprachen, Nachhaltigkeit

Software/Tool. gIBIS (Conklin & Begemann, 1987) ist ein Hypertext-Werkzeug, mit dem Design-Überlegungen in der Gruppe unterstützt werden sollen und das gleichzeitig der Dokumentation der Begründungen dient. Dafür erlaubt es, Argumente und Standpunkte grafisch über drei verschiedene Inhaltstypen zu strukturieren: Issues, Positions, Arguments. Für diese wird ein Übersichtsgraph dargestellt, in dem die jeweiligen Inhaltsknoten durch Relationen miteinander in Verbindung gesetzt werden. Die Knoteninhalte werden in separaten Fenstern dargestellt.

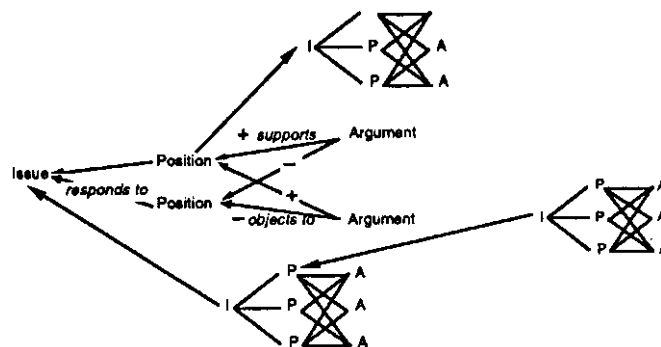


Abbildung 16

Schematische Darstellung der gIBIS-Darstellungen. „Issues“ werden durch verschiedene „Positions“ aufgegliedert. „Arguments“ nehmen dazu Stellung.

Ausgegangen wird davon, dass durch unterschiedliche Interessen jeweils andere Standpunkte eingebracht werden und diese in einen Zusammenhang für die eigentliche Design-Aufgabe gebracht werden müssen. In den Graph können und sollen Skizzen und Notizen genauso wie Design-Entscheidungen und Begründungen aufgenommen werden. Die Design-Überlegungen werden asynchron editiert, wodurch die Navigation durch die entstandenen Netze eine große Rolle spielt. Diese wird durch einen Browser unterstützt, der das Netz im Überblick abbildet, in dem jedoch nur die Struktur erkennbar ist, sowie durch einen lokalen Browser, der Teile des Netzwerkes vergrößert („zoom“), und mit den Details für die Knotentypen darstellt. Der lokale Browser stellt auch die syntaktische Korrektheit der Netze sicher.

Die Navigation wird weiterhin durch Suchen ermöglicht. Dafür werden Beispielknoten erstellt, für die dann Entsprechungen im Graph gesucht werden („query by example“, Conklin & Begemann, 1987, S. 249).

Realisierung des Szenarios. gIBIS unterstützt das asynchrone Arbeiten. Es dient dabei als Kommunikationsmedium für die Tele-Kooperation. Konkrete „face-to-face“-Diskussionen sind nicht explizit eingeplant, das heißt, dass alle Argumente über das System ausgetauscht werden müssen. Unklar bleibt, wie die Nutzerinnen und Nutzer die

Veränderungen erkennen und verfolgen können, ohne durch das gesamte Netz stöbern zu müssen.

Technisch wird die Kooperation im LAN durch „locking“ von Knoten und durch Benachrichtigungen bzgl. inhaltlichen Veränderungen unterstützt.

Organisatorische Einbettung. Die ursprüngliche Motivation, ein solches System zu entwickeln, bestand darin, den Design-Prozess selbst untersuchen zu können und weiterhin in der Frage, wie große Mengen an informellen Inhalten genutzt, indexiert oder auch wiedergefunden werden können. Mit dieser Ausrichtung kann gIBIS nicht nur als Wegbereiter für viele Systeme zum „design rationale“ gewertet werden, sondern stellt außerdem bereits viele Fragen zum Thema Wissensmanagement, das eigentlich erst 10 Jahre später aktuell wurde. Von dort stammt vor allem die Idee, visuelle Repräsentationen für die Analyse von Design-Ideen einzusetzen, sowie die Annahme, dass Ideen ein ganz zentrales Element solcher Wissensprozesse sind. Auch die Relevanz, die Wissensentwicklung zu dokumentieren, wird dort bereits hervorgehoben.

3.2.7 MindManager

- *Schwerpunkt.* Kommunikationsmedien und Wissensprodukte
- *Zielstellung.* „Mind Mapping“
- *Kriterien.* Nutzung visueller Sprachen

Software/Tool. MindManager (Abbildung 17) von Mindjet ist vermutlich das am weitesten verbreitetste „Mind Mapping“-Tool auf dem Markt ([MindManager]⁸). Das Tool dient der Ideenbildung durch Brainstorming, Vorbereitung von Meetings und durch den strukturierenden Ansatz beispielsweise auch der Projektorganisation. Die damit zu entwickelnden „Maps“ (Karten) werden im Wesentlichen in Baumstrukturen dargestellt. Nur mit einer Art von Relationen können auch einzelne Äste miteinander assoziiert werden. Der Typ eines Knotens wird meistens über ein Symbol verdeutlicht, das z.B. die Art referenzierter Medien symbolisiert. Ansonsten können auch Graphiken in den Arbeitsbereich eingefügt werden. Jede Art von Text wird in Knoten eingegeben oder durch referenzierte Medien verknüpft.

Das System übernimmt die Anordnung der Knoten im Workspace. Dies ist einerseits hilfreich, da eine gleichmäßige Verteilung der Eingaben angestrebt wird, andererseits aber auch verwirrend, wenn automatische Veränderungen den „Bildeindruck“ zerstören.

⁸ Erläutert wird das System im Kontext von Kreativitätswerkzeugen auch in einem Artikel der Zeitschrift c't (20/98, Jungbluth).

Realisierung des Szenarios. In der ursprünglichen Version war MindManager ein „single user“-System. Aktuell wurde es aber zu einem System erweitert, das die synchrone Kopplung zulässt. Dafür muss ein Konferenzserver mit installiert werden, an den sich die Applikationen anmelden müssen. Zur Kopplung muss eine Sitzung freigegeben werden, wofür auch Zugriffsrechte vergeben werden können. Der Server wartet eine Liste aller aktuellen Konferenzen, die auch permanent eingerichtet werden können.

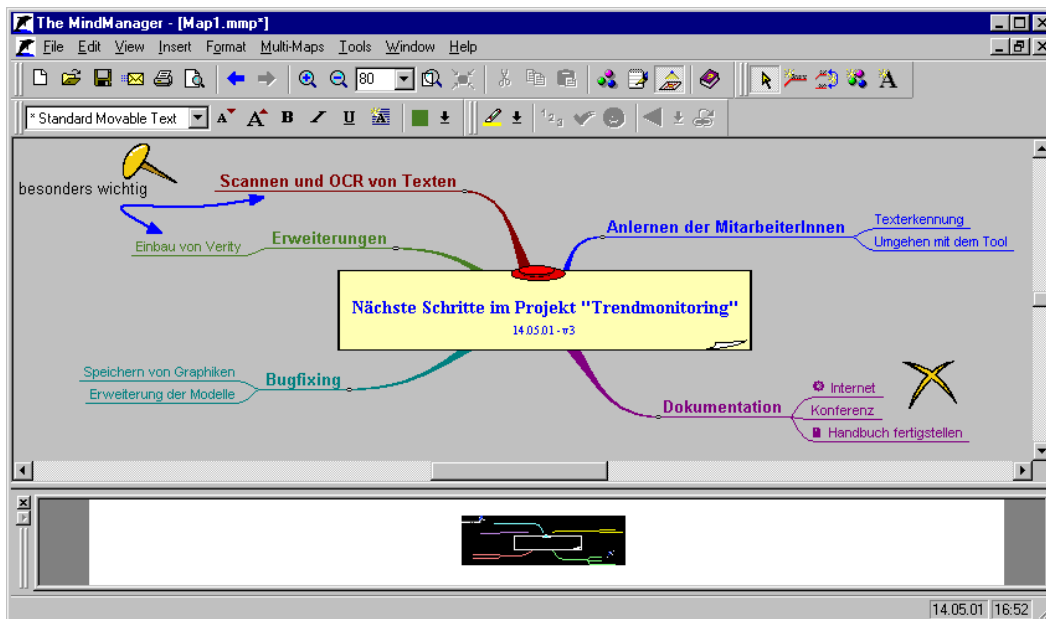


Abbildung 17

Das Interface des Tools MindManager.

Organisatorische Einbettung. Ein „Mind Mapping“-Tool kann vielseitig eingesetzt werden, von der Vorbereitung bis zur Dokumentation. Konkrete Prozesse werden hier nicht unterstützt, durch die Exportmöglichkeit zu Standardformaten wie PowerPoint, Word, Outlook und html können allerdings Medienbrüche beim Wechsel zu anderen Tools vermieden werden. Somit ist durchaus eine flexible Integration des Tools in existierende Prozesse möglich.

3.2.8 OneNote

- *Schwerpunkt.* Kommunikationsmedien und Wissensprodukte
- *Zielstellung.* Organisieren von Notizen
- *Kriterien.* Nutzung visueller Sprachen, Nachhaltigkeit

Software/Tool. OneNote von Microsoft ([OneNote]) kann hier nur als Ausblick verstanden werden, denn dabei handelt es sich um ein Tool der Office-Gruppe, das erst

Mitte 2003 auf den Markt kommen soll⁹. Das Tool stellt mehrere Strukturierungsmöglichkeiten zur Verfügung, wie sogenannte Notebooks, Files und Seiten. Auf den Seiten können Bilder, Text, Handschrift und Audio-Informationen angeordnet und verschoben werden. Inhalte können per Drag&Drop aus Web-Browsern integriert werden und behalten als Information ihre Internetseite. Als Bestandteil der Office-Familie besteht zu den Parallelprodukten eine Exportmöglichkeit. Ergebnisse können auch im Web publiziert werden.

Für die erstellten Daten werden Suchen angeboten und es wird besonderer Wert auf die Datensicherheit gelegt.

Realisierung des Szenarios. Das System ist als „single user“-System geplant und zielt vermutlich auf eine deutlich verbesserte Verwendung als Notizbuch im Vergleich zu Handhelds.

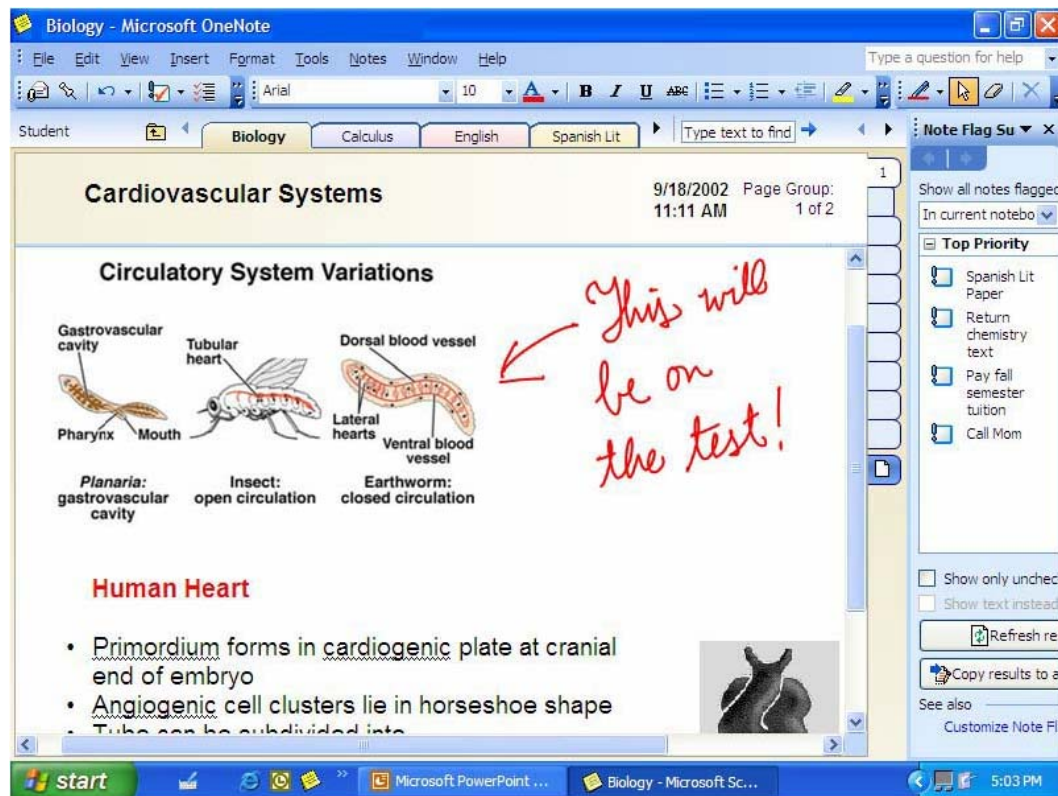


Abbildung 18

Oberfläche und Nutzungsbeispiel des Tools OneNote.

Organisatorische Einbettung. Das Szenario eröffnet komfortable Notizmöglichkeiten. Das kann weite Bereiche abdecken, angefangen vom „Mind Mapping“ bis hin zur Planung von Aufgaben und Terminen. Es unterstützt vor allem den Übergang dieser

⁹ Die Beschreibung stützt sich deshalb nur auf kurze Informationen aus dem Internet.

alltäglichen Notizen in ein digitales Format mit Übergang zu weiteren (Word, etc.) und zielt damit unter anderem auf die Vermeidung von Medienbrüchen. Der Schritt, die Inhaltsaufbereitungen und Datensicherungen auch kooperativ zu nutzen, wurde mit diesem Tool bisher noch nicht vollzogen.

3.2.9 Sepia

- *Schwerpunkt.* Kommunikationsmedien und Wissensprodukte
- *Zielstellung.* Planung und Erstellung von Hypertexten
- *Kriterien:* Nutzung visueller Sprachen, Nachhaltigkeit

Software/Tool. Das Sepia-System (Streitz, Haake, et al., 1992; Streitz, Haake, et al., 1998) soll Autorinnen und Autoren kognitiv bei der Erstellung von Hypertexten unterstützen (Abbildung 19).

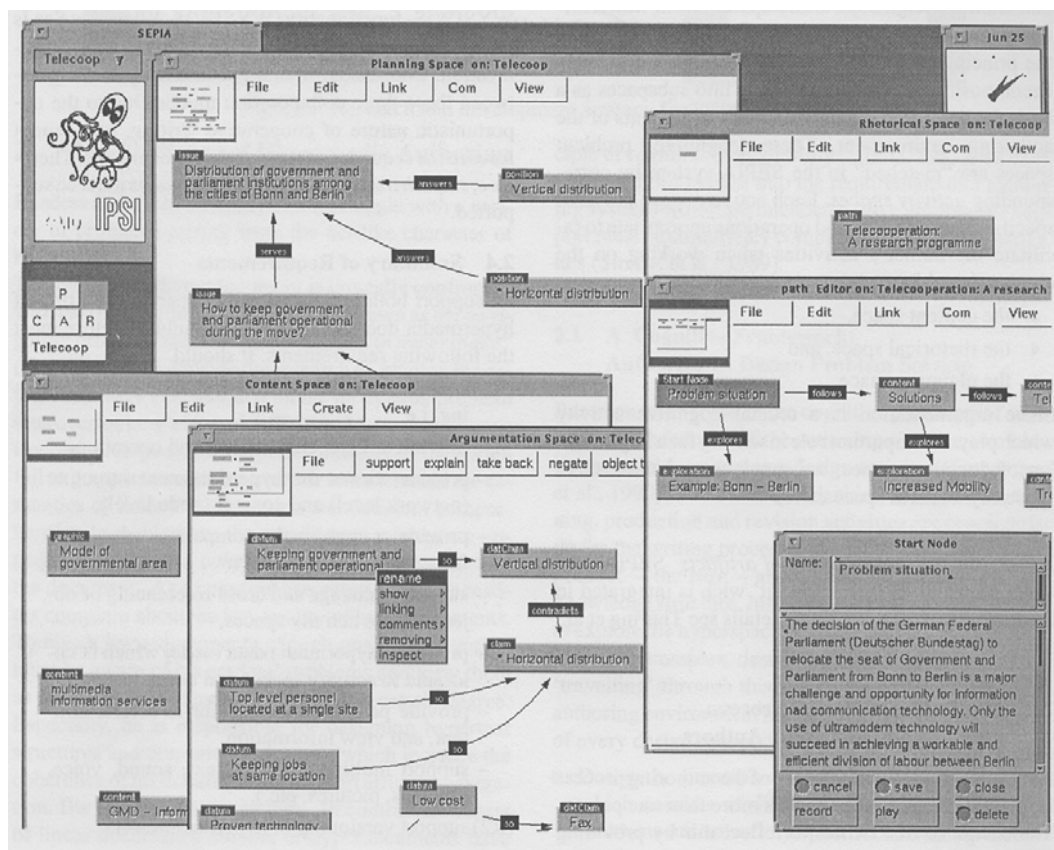


Abbildung 19

Das Sepia-System mit mehreren geöffneten „activity spaces“.

Der Schreibprozess wird als Design- und Problemlöseprozess verstanden, bei dem Planungs-, Produktions- und Revisionsphasen durchlaufen werden. Bei diesem gemeinsamen Schreiben müssen vor allem Probleme auf der inhaltlichen, der rhetorischen und der Planungsebene gelöst werden, für die unterschiedliche „activity spaces“ zur

Verfügung gestellt werden, ergänzt um einen zusätzlichen Argumentations-Workspace. Ein „activity space“ zeichnet sich durch eine vordefinierte visuelle Sprache aus Knoten- und Kantentypen aus. Die Knoten selbst tragen keine operationale Semantik. Wie bei Dolphin können „dreidimensionale“ Strukturen angelegt werden, indem Knoten in andere verschoben werden.

Realisierung des Szenarios. Das Sepia-System entstand aus der Überzeugung, dass große Hypertexte fast immer von Gruppen erstellt werden. Es wurde für die örtlich verteilte Kooperation entwickelt, die wiederum durch unterschiedliche Kooperationsmodi strukturiert wird. Es steht ein „single user“-Modus zur Verfügung, eine lose Kopplung und eine starke Kopplung. Diese drei Modi unterscheiden sich insbesondere dadurch, inwieweit die einzelnen Autoren und Autorinnen über die Arbeit anderer an gleichen Knoten oder Knotengruppen informiert werden und gleichzeitige Arbeit erlaubt wird, bzw. einzelne Objekte gesperrt werden.

Die Sepia-Architektur baut auf einer „Hypermedia Engine“ auf, die die anfallenden Daten, in einer „Hyperdocument“-Datenbank verwaltet. Für die einzelnen Objekte (Knoten und Links) werden darin Inhalte zusammen mit Attributen abgelegt, so dass sie auch in unterschiedlichen „activity spaces“ angezeigt werden können.

Ein kooperativer Hypermediaserver organisiert den „multi-user“-Zugriff auf Dokumente. Jeder Client beinhaltet einen „broadcast listener“ und einen „broadcast server“ für die Synchronisation der Arbeit in „shared“ Workspaces.

Organisatorische Einbettung. Ein Prozess bei der Erstellung von Hypertexten wird implizit durch die „activity spaces“ unterstützt. Eingebettet in den „rhetorical space“ ist außerdem ein „construction kit“, das dabei helfen soll, am Ende den eigentlichen Hypertext zu erstellen, um den Medienbruch zwischen der Textplanung und dem eigentlichen Hypertext zu vermeiden.

Eine Verknüpfung der Ergebnisse, der Hypertexte, im Sinne eines „corporate memory“ wird nicht unterstützt. Darin wäre es beispielsweise sinnvoll, Autoren und Autorinnen über Änderungen zu informieren, wie beispielsweise bei Csile. Undeutlich bleibt in den Beschreibungen, wie mit größeren Netzen umgegangen werden kann. Unklar bleibt auch, inwieweit das potentiell asynchrone Szenario die Erstellung der Hypertexte unterstützt, wie beispielsweise mit der dezentralen Veränderung von Inhalten umgegangen wird und wie die eigentlichen Abstimmungsprozesse dann ablaufen.

Das System ist sehr umfangreich, an hohe technische Voraussetzungen gebunden und integriert viele Komponenten, angefangen bei den „activity spaces“ für den Autorenprozess, über die Ermöglichung kooperativen Arbeitens bis hin zur persistenten

Datenhaltung. Es scheint jedoch weniger komponentenorientiert implementiert, so dass es schwierig adaptierbar zu sein scheint. So könnten beispielsweise die Kooperationskomponenten eher eigenständig und visuelle Sprachen leichter definierbar sein.

3.3 Erfahrungen bei der Verwendung visueller Sprachen

Problemzerlegungen. In Conklin und Begemann (1987) werden in Bezug auf die Nutzung des gIBIS-Systems verschiedene Erfahrungen aufgezeigt. Dazu gehört, dass eine Schwierigkeit in der zu frühen Zergliederung von Problemen bestand. Das sei vor allem dann nicht produktiv, wenn das eigentliche Problem noch nicht gut verstanden ist. Gleiches wird auch von McKerlie und MacLean (1994) im Rahmen von QOC-Studien erwähnt. QOC, Questions-Options-Criteria (MacLean et al., 1991), ist nicht an ein System gebunden, sondern stellt ein Schema dar, nach dem Design-Probleme analysiert werden können. Sie ist dem IBIS-Schema sehr ähnlich, grenzt sich aber davon ab, als dass darin die Analyse und Reflexion durch die Designer angeregt wird, während bei der Umsetzung von IBIS in gIBIS vor allem die Dokumentation der entstehenden Ideen unterstützt wird (McKerlie & MacLean, 1994).

Löwgren (1994) analysiert den Design-Prozess weitergehend auf Basis der QOC-Methode und kommt zu einer Charakterisierung von Design-Prozessen als „evolutionär“ oder „argumentativ“. „Evolutionär“ besagt, dass sich die Argumente, die zu Entscheidungen geführt haben, zu diesen korrespondieren und jeweils den aktuellen Stand reflektieren, nicht aber die Historie und Gründe die zu Veränderungen geführt haben.

Bei „argumentativen“ Herangehen werden verschiedene Alternativen zu einem Zeitpunkt gegeneinander abgewogen werden, um daraufhin beispielsweise zu einer Entscheidung zu kommen. Im weiten Sinn könnte behauptet werden, dass gIBIS den „evolutionären“ Ansatz realisiert, während die QOC-Methode auf den analytischen Ansatz abzielt.

In der Untersuchung wurde beobachtet, dass beim „evolutionären“ Ansatz eine Tendenz besteht, vorschnell in subjektiv bevorzugte Ideen zu verfallen und andere dann nicht mehr zu verfolgen oder gar nicht erst versucht wird, andere Ideen zu finden.

Die Unterstützung der „argumentativen“ Herangehensweise, die darauf abzielt den gesamten Argumentationsraum aufzuspannen, hat den Designern insofern geholfen, als dass sie stark subjektive Entscheidungen später revidierten, da diese Inkonsistenz durch die Darstellung transparent wurde. In McKerlie und MacLean (1994) wird auch auf die Eigenart verwiesen, dass eher versucht wird, stützende Kriterien für einen Vorschlag zu finden als Gegenargumente.

Diese Erfahrungen sind insofern wichtig, als dass visuelle Sprachen generell den Zerlegungsaspekt in sich tragen, welcher gerade der Klärung von Sachverhalten dienen soll. Wenn die diskussionsbegleitende Zergliederung jedoch gerade zur Vernachlässigung von Aspekten führt, wie vielleicht in einem „evolutionären“ Ansatz, so wäre das fatal.

Ein weiteres für gIBIS erkanntes Problem bestand darin, dass die Autorinnen und Autoren allein durch die Idee des Hypertextes dazu angehalten waren, die Knoteninhalte so zu formulieren, dass sie weitgehend selbsterklärend waren, denn der Kontext, der in normalen, linearen Texten eher gegeben ist, geht deutlich verloren. Leser können auf beliebigen Wegen zu diesem Knoten gelangen. Dieses Problem ist nach meiner Ansicht grundlegend mit der Idee des Hypertextes verwoben.

Einbeziehen der Visualisierung in die Gesprächssituation. Im Kapitel 4 entsteht im Ergebnis eine zyklische Strategie, in der sich Notations- und Diskussionsphasen abwechseln. Eine entsprechende Beobachtung wurde in Stefik et al. (1987) in Bezug auf das Cognoter-System beschrieben. Auch dort lösen sich reine Diskussionsphasen ab mit Phasen, in denen alle Beteiligten parallel Eingaben anfertigen über die dann später – als Gesprächsfokus – wieder diskutiert wird. Dort wird außerdem beschrieben, dass zweidimensionale Gruppierungen ein Übergangsstadium bilden, zwischen reinen Sammelphasen und Phasen, in denen Relationen eingefügt werden.

In Suthers (1999C) wird die Beobachtung gemacht, dass die externe Repräsentation in der „face-to-face“-Situation weniger dazu dient, in bzw. mit dieser Repräsentation zu argumentieren, als dazu, sie als Basis für die Argumentation zu verwenden, als Stichwortgeber. Er begründet das so, dass die Repräsentation nicht die eigentlich Kommunikation manifestiert, sondern diese *durch* das Medium stattfindet.

Einbindung in Arbeitssituationen. Bei der Verwendung von Cognoter und Argnoter zeigte sich, dass das gemeinsame Erarbeiten von Inhalten stärker konsensorientiert verläuft, als Diskurse, in die einzelne Personen bereits vorbereitete Beiträge einbringen. Dies deckt sich mit den Erfahrungen aus der in Kapitel 4 beschriebenen Vorstudie, in der auf Basis von vorbereiteten Beiträgen eher disputiert wurde und kann bewusst methodisch eingesetzt werden.

Bezüglich Csile wird vor allem darauf hingewiesen, dass Funktionalitäten immer auch einen Sinn im didaktischen Ablauf haben müssen. Dort gab es die Möglichkeit, Schlüsselwörter für Beiträge zu vergeben, um so eine Abstraktion zu erhalten. Dieser Schritt der Abstraktion wurde jedoch nicht ausgenutzt. Er war weder in den Unterrichtsprozess explizit integriert, noch in die Abläufe, die durch das System nahegelegt werden.

Ein großer Wert bei Csile bestand darin, dass die Beteiligten einen Fortschritt der Inhalte erkennen konnten und wollten. Dies ist sicherlich gerade wegen der verteilten Arbeitssituation wichtig: jede und jeder will sehen, was mit den Ergebnissen passiert. In einer „face-to-face“-Arbeitssituation ist dies anders, da der Progress der Inhalte ja gerade zusammen in der Gruppe unternommen wird.

Bei itIBIS (Conklin & Burgess-Yakemovic, 1996), einer einfachen, textbasierten Umsetzung des IBIS-Schemas, wurde allein ein Erfolg darin gesehen, dass die Gruppenmitglieder die IBIS-Methode eintrainierten. Dies führte zu produktiveren Arbeitstreffen: Es wurde danach besser erkannt, wann man sich vom eigentlichen Thema weg bewegte, und man konnte einfacher wieder auf den Ausgangspunkt zurückkommen.

Die QOC-Methode war erfolgreich, um auf die zentrale Fragestellung zu fokussieren, diese nicht aus den Augen zu lassen und konkurrierende Ideen darzustellen (McKerlie & MacLean, 1994).

Wiederverwendung. Bei der Verwendung von itIBIS (Conklin & Burgess-Yakemovic, 1996) wurde deutlich, dass die Aufschlüsselung der Design-Ideen einerseits den Designern selbst hilft, sich an diese später wieder zu erinnern. Andererseits, beim Ausscheiden einer für das Design zentralen Person aus der Arbeitsgruppe, konnte die Gruppe das Produkt auf Basis der dokumentierten Design-Entscheidungen einigermaßen problemfrei zu Ende entwickeln. Generell konnten Fragen der Art „Warum haben wir das so gemacht?“ rückverfolgt werden. Im Vergleich zwischen verschiedenen Arbeitsgruppen, die mit und ohne die IBIS-Methode gearbeitet hatten, führte deren Verwendung zu einer transparenteren Aufschlüsselung der Anforderungen und Begründungen.

Da die Ergebnisse der IBIS-Methode und der QOC-Methode eher den „Design Space“ darstellen, fehlt eine Darstellung der Entwicklung der Ideen und teilweise der verworfenen Ideen (McKerlie & MacLean, 1994).

3.4 Synopse

Aus den vorangegangenen Kapiteln wurde deutlich, dass Visualisierungen einen breiten Einsatz beim Konstruieren von Wissen haben. Da sie außerdem sehr gut verwendet werden können, um Kommunikationssituationen durch weitere Ausdrucksmittel und Strukturelemente anzureichern, sind Visualisierung insbesondere für den Informationsaustausch in Gruppen geeignet.

Die Arbeitshypothese dieser Arbeit besteht darin, dass visuelle Sprachen dazu geeignet sind, um mit ihnen sonst flüchtige Inhalte aus diversen Arbeitssituationen zu notieren.

Deren potentiell semi-formale Darstellung bietet außerdem viele Optionen, die Inhalte Computersystemen zugänglich zu machen. Damit sind alle möglichen Funktionalitäten gemeint, die sich einerseits auf die Einbindung der Produkte in Arbeitsprozesse beziehen können, wie das Retrieval oder die Unterstützung eines Wissensflusses im Allgemeinen. Andererseits sind damit auch solche Funktionen gemeint, die die Kommunikationssituation selbst komfortabler gestalten, wie verbesserte Referenzen zwischen Beiträgen, ausgefeilte Sprachelemente, Perspektiven, Sortierungen, Feedback, etc. Diese Überführung von flüchtigem, dynamischem Wissen in eine technisch handhabbare Form soll insbesondere die Nachhaltigkeit der besprochenen Ergebnisse verbessern. Hierbei handelt es sich um ein zentrales Ziel im Wissensmanagement.

Im Verlauf dieser Arbeit entstand die Vermutung, dass zwar diverse Systeme existieren, die visuelle Sprachen in der einen oder anderen Weise verwenden, dass aber genau dieser Überführungsprozess meistens nur in Teilen bedacht wird. Das gilt genauso für die Unterstützung eines Prozesses durch das System, wie für die Einbettung des System in einen organisatorischen Ablauf. Mit einer Prozessunterstützung ist dabei nicht ein Workflowsystem gemeint, das Arbeitsvorgänge strikt vorgibt, sondern die freie Einbindung von Tools in den Arbeitsalltag, worin trotzdem wiederkehrende Abläufe stattfinden. Ein Beispiel dazu ist Projektarbeit, die von den ersten Entscheidungen und Planungen hin zu Ergebnissen verläuft und die dokumentiert werden muss.

Für den Bereich des Wissensmanagement ergäbe sich daraus ein großer Mangel in Bezug auf den Mehrwert von Systemen, die z.B. nur einzelne Arbeitsphasen unterstützen.

Die Ursachen dafür liegen natürlich meistens in einer anderen Zielstellung, die sich die jeweiligen Entwicklerinnen und Entwickler gesetzt haben. Genau deshalb möchte ich ein offenes Feld im Bereich der Systementwicklungen andeuten, das gerade für das Wissensmanagement interessant sein sollte.

In der folgenden Zusammenschau werden deshalb Kriterien für Systeme vorgestellt, die nach meiner Ansicht relevant dafür sind, ob sie einen Arbeitsprozess unterstützen, in dem gerade dynamisches Wissen in einen Zyklus der Externalisierung und Internalisierung eingebracht werden kann. Darunter fallen auch Kriterien dafür, ob die potentiellen Möglichkeiten visueller Sprachen ausgenutzt werden. Die Kriterien werden den drei Ebenen zugeordnet, auf denen Diskussionen technisch unterstützt werden können (Kapitel 1.5.2).

3.4.1 Software/Tool

1. *Domänenunabhängigkeit* ist wichtig, um ein Visualisierungshilfsmittel alltäglich verwenden zu können.
2. Unrealistisch ist die Forderung einer wirklich formalen Repräsentation, um während normaler Arbeitsabläufe Wissensdomänen zu erstellen. Sinnvoll sind dagegen Möglichkeiten, Notizen digital aber *handschriftlich* anfertigen zu können, um flexible Notiztools zu erhalten. In diesem Zusammenhang könnte auch Handschrifterkennung eingesetzt werden.
3. „*Epistemic games*“ werden in Collins und Ferguson (1993) gleichermaßen als analytische wie konstruktive Vorgehen eingeführt. Das „epistemic game“ fügt einer epistemischen Form die Regeln hinzu, wie mit ihr umgegangen werden soll (Auf diese Begriffe wird im Kapitel 6.1 vertieft eingegangen.). Die Tabelle ist beispielsweise eine ganz grundlegende epistemische Form, die ganz unterschiedlich eingesetzt werden kann. Zu einer epistemischen Form können also mehrere epistemische Spiele existieren. Diesen konkreten Zusammenhang einer Form mit einer Umgehensweise bezeichne ich im Weiteren auch als Methode. Verschiedene solcher strukturellen Methoden sind mit visuellen Sprachen relativ leicht umsetzbar, wie Brainstorming, Meta-Plan, Argumentationsnetze oder manche Entscheidungsmethoden, und passen typischerweise zu bestimmten Phasen in einem Arbeitsprozess. Deshalb ist es sinnvoll mehrere solcher Methoden zur Verfügung zu haben und außerdem, Inhalte von einer Methode in eine andere zu übernehmen.
4. Wegen der Forderung nach Domänenunabhängigkeit, der flexiblen Nutzung, dem Wunsch nach Methodenkatalogen und der Unterstützung eines Arbeitsprozesses wäre es sinnvoll, dass Systeme an die jeweiligen Anforderungen von Nutzergruppen *adaptiert* werden können. Denkbar wäre das, indem Nutzerinnen und Nutzern die Möglichkeit gegeben wird, die Syntax von visuellen Sprachen und die Semantik von Operationen auf deren Objekten zu definieren. Eine Engineering-orientierte Variante bestünde z.B. darin, Architekturen bereitzustellen, zu denen Sprachen und Methoden als „Plug-In“ hinzugefügt werden können. Wang und Haake (2000, S. 129) zitieren vier Arten, auf die ein System adaptierbar sein kann: Es kann generische Objekte zur Verfügung stellen (flexibel), es kann parametrisierbar sein, es kann leicht integriert werden oder das System kann selbst verändert werden („tailorable“). Mittel dazu können beispielsweise „Templates“ oder Scriptsprachen sein.
5. *Feedback* muss nicht auf eine konkrete Domäne Bezug nehmen. Es kann z.B. auch Rückmeldungen über die Gruppenarbeit liefern sowie Auswertungen liefern, die

unter Umständen Bezug auf angewendete Methoden nehmen. Gerade diese Auswertungen harmonieren gut mit einer semi-formalen Darstellung mit visuellen Sprachen, wenn sie nicht die konkreten Inhalte analysieren, sondern auf strukturellen oder abstrakten Eigenschaften der Sprachobjekte beruhen.

3.4.2 Realisierung des Szenarios

6. Werden Diskussionen als Szenario verwendet, um Inhalte zu externalisieren, diese flüchtigen mündlichen Beiträge zu fixieren und zu dokumentieren, um diese später wieder aufzugreifen, dann beinhaltet dies Übergänge zwischen verschiedenen Arbeitsszenarios: Es wird synchron und asynchron gearbeitet, individuell und gemeinsam, verteilt und im gleichen Raum, es finden Vorbereitungen und Nachbereitungen statt, Materialien werden verwendet und Inhalte neu einbezogen. Vor diesem Hintergrund ist es sinnvoll, unterschiedliche Szenarien zu unterstützen, bzw. die Übergänge zwischen diesen. In Bezug auf die Realisierung des Szenarios werden dazu die Kriterien *verteilt*, „*face-to-face*“, *synchron* und *asynchron* aufgenommen. Werden mehrere davon mit einbezogen, so wird das zumindest als Indiz dafür gewertet, dass Rücksicht auf komplexere Arbeitsabläufe genommen wird. (Die nächste Rubrik, „Organisatorische Einbettung“, beinhaltet angrenzende Kriterien.)
7. „*Shared objects*“ sind hier sowohl im Sinne einer Bedeutung zu verstehen, die in einer Gruppe geteilt wird, als auch im technischen Sinn als Objekte in „shared“ Workspaces. In der „face-to-face“-Situation muss nach dieser Beschreibung eine gemeinsam konstruierte Bedeutung nicht an ein technisches „shared object“ gebunden sein. Die Feststellung, dass mit technischen „shared objects“ umgegangen wird, wird als Indiz dafür gewertet, dass „kollaborative“ Situationen eingeplant sind. Darunter sei hier das wirklich gemeinsame Arbeiten am gleichen Wissensprodukt zu verstehen, das zu einem inhaltlichen Austausch führt, einer Wissenskommunikation. Technische „shared objects“ sind also ein Indiz dafür, dass die gemeinsame Wissenskonstruktion unterstützt wird.

3.4.3 Organisatorische Einbettung

8. Die *Interoperabilität* zwischen den Tools und Medien ist sinnvoll, um Medienbrüche zu vermeiden, denn es muss bedacht werden, dass in einer Organisation verschiedenste Materialtypen für die unterschiedlichen Arbeitszwecke verwendet werden, die unter Umständen in eine Kooperation und Kommunikation mit einbezogen werden sollen. In gleicher Weise sollen Kommunikationen in einen

Arbeitskontext mit einbezogen werden und Ergebnisse auch außerhalb des Kommunikationskontextes zugänglich sein. Bei einem Wissensmanagement-Ansatz ist nach meiner Ansicht nicht davon auszugehen, dass jede gewünschte Funktionalität aus einem umfassenden System heraus zur Verfügung gestellt wird. Vielmehr ist von einem Zusammenspiel vieler einzelner Tools oder Komponenten auszugehen.

9. *Repositories* dienen formal als „Gruppendächtnis“ und bilden eine Grundlage für die Wiederverwendung von Informationen. Die Ergebnisse aus den Kommunikationen sollten darin aufgenommen und auch wiedergefunden werden können. Außerdem ist es wünschenswert Beziehungen zwischen den Inhalten des *Repositories* herstellen zu können.
10. Um in größeren Datenbeständen Informationen wiederfinden zu können, sollte ein *Informationretrieval*-Ansatz integriert sein. Dieser ist abhängig von der verwendeten Technologie. Dieser Aspekt schließt sich also direkt an den der Interoperabilität an. Abgelegte Formate sollten beispielsweise von handelsüblichen Suchmaschinen indexierbar sein. Andere Möglichkeiten sind Verschlagwortungen und deren Verwaltung in Datenbanken oder generell Metadaten-Ansätze. Graphisch oder Hypermedia-artig aufgearbeitete *Repositories* können auch einen explorativen Zugang ermöglichen.
11. Durchaus abhängig von der Art der Gruppenunterstützung kann die „*notification*“ – gemeint ist die Benachrichtigung über Ereignisse in den Datenbeständen – ein sehr wichtiges Konzept der Kooperation und Wissenskonstruktion sein.
12. Ein Visualisierungs-Tool kann zwar technisch für bestimmte Arbeiten geeignet sein und auch Arbeitsprozesse unterstützen, sollte aber auch in entsprechende *nicht-technische Arbeitsprozesse* organisatorisch mit einbezogen werden.

Nach diesen Kriterien werden im Folgenden die vorgestellten Systeme eingeordnet (Tabelle 5, Tabelle 6, Tabelle 7). Die Tabelle 5 widmet sich der ersten Gruppe der Kriterien, die die Software selbst charakterisieren. Alle in Betracht gezogenen Systeme sind domänenunabhängig (Tabelle 5), was eine Voraussetzung für die allgemeine Kommunikationsunterstützung ist. 7 der 10 Systeme verbinden ihren Ansatz mit einer oder mehreren Methoden („epistemic games“). Ausschließlich das Sepia-System stellt auch einen abgestimmten Methodenkatalog zur Verfügung. Chips bietet die Möglichkeit, „epistemic forms“ zu definieren, um so gruppenspezifische Prozesse begleiten zu können, wobei die Art des Umgangs mit diesen Darstellungen selbst schon ein Ergebnis der Kooperation ist. Gerade durch diesen erheblichen Freiheitsgrad geht jedoch die Anleitung verloren, die Sepia durch abgestimmte Methoden anbietet. Das CardBoard bietet zwar die

Möglichkeit, visuelle Sprachen extern zu spezifizieren, ist in der bisher vorgestellten Form aber noch nicht an eine bestimmte Anwendung adaptiert. Das Dolphin-System wiederum verzichtet weitgehend auf Methoden und unterstützt dafür ganz wesentlich die „dreidimensionale“ Strukturierung von Inhalten. Csile bietet in der Grundfunktionalität zwar keine expliziten Strukturierungen an, die Kommunikation wird aber maßgeblich durch den Lernprozess gesteuert, in den die Systemverwendung einbezogen wird. Allerdings integrieren neuere Versionen auch schon spezifische Methoden, wie beispielsweise „Concept Mapping“.

Software/Tool	Domänen-unabhängigkeit	Handschrift-eingabe	„epistemic games“ „epistemic forms“	Adaptierbarkeit	Feedback
Belvedere	+	-	∅	-	+
CardBoard	+	-	+	+	-
Chips	+	+	+	+	-
Cognoter/ Argnoter	+	-	∅	-	-
Csile	+	-	-	-	-
Dolphin	+	+	-	-	-
gIBIS	+	-	∅	-	-
MindManager	+	-	∅	-	-
OneNote	+	+	-	-	-
Sepia	+	-	+	-	-

Tabelle 5

Eigenschaften und Funktionen der Systeme.

+: Kriterium wird erfüllt bzw. Funktionalität ist vorhanden.

o: Wird nur in Bezug auf „epistemic games“ und „epistemic forms“ vergeben und bedeutet, dass genau eine Methode unterstützt wird.

-: Kriterium wird nicht erfüllt bzw. Funktionalität ist nicht vorhanden.

Die meisten Ansätze zum „meeting support“ bzw. zur Kommunikationsunterstützung gehen davon aus, dass Arbeitsphasen durch adäquate Methoden begleitet werden sollten. Dies kann einerseits durch die Verwendung verschiedener Tools gelöst werden, an die dann aber ein hoher Anspruch in Bezug auf die Interoperabilität bestünde. Andererseits kann ein einzelnes Tools verschiedene Methoden anbieten. Dies hat den Vorzug, dass die verwendeten Objekte eher interagieren können. Offen bleibt nach meiner Ansicht, ob ein umfangreicher Methodenkatalog besser vorgegeben wird, ob generell ein adaptiver Ansatz wie im Chips-System verfolgt werden sollte, oder ob ein Mischkonzept anzustreben ist. In allen Fällen ist die Verwendung von visuellen Sprachen, für die Syntax, Operationen, Constraints, usw. von Vorteil.

Die Option, im Kommunikationsprozess Feedback zu liefern, wird außer vom Belvedere-System nicht genutzt. Denkbar sind hier beispielsweise Analysen der Kooperation (Mühlenbrock, 2001; Mühlenbrock & Hoppe, 1999; Mühlenbrock & Hoppe, 2001), die ohne das System kaum durchführbar sind, zumindest nicht so, dass das Ergebnis gleich

wieder in die Gruppe zurück reflektiert werden kann. Auch Beteiligung kann bei größeren Gruppen interessant sein, insbesondere mit Bezug zum Thema.

Tabelle 6 gibt einen Überblick, welches Szenario mit welchem System unterstützt wird. In den Spalten sind die vorher genannten Kriterien so zusammengefasst worden, dass redundante Aussagen weitgehend vermieden werden. Beispielsweise ist davon auszugehen, dass für ein Szenario mit einer örtlich verteilten Gruppe, die synchron kooperiert, „shared objects“ eine Voraussetzung der gemeinsamen Kommunikation sind (erste Spalte). In diesem Kontext meint „shared objects“ deshalb sowohl die technisch gekoppelten Objekte als auch deren Bedeutung, die aus der Kommunikation heraus entsteht.

Realisierung des Szenarios	Verteilt, Synchron, „Shared Objects“	„Face-to-face“, „Shared Objects“	Asynchron, „Shared Objects“
Belvedere	Ø	+	-
CardBoard	Ø	+	-
Chips	+	+	+
Cognoter/ Argnoter	Ø	+	-
Csile	-	-	+
Dolphin	Ø	+	-
gIBIS	-	-	+
MindManager	+	Ø	+
OneNote	-	Ø	-
Sepia	+	Ø	+

Tabelle 6

Einordnung der Systeme in Bezug auf die Szenarien, die mit ihnen realisiert werden können.

+ : Es handelt sich um das mit dem Tool angestrebte Szenario.

o : Das Szenario ist technisch möglich, wird aber nach der Toolbeschreibung nicht angestrebt.

- : Das Szenario wird technisch nicht unterstützt.

In einem „face-to-face“-Szenario (zweite Spalte), kann zwar technisch betrachtet synchron oder asynchron kooperiert werden, im Fall der asynchronen Kooperation ist aber die Tatsache, sich am gleichen Ort zu befinden, selten noch relevant, es sei denn dafür, dass schnell zwischen zwei Szenarien gewechselt werden kann. Im Fall des „face-to-face“-Szenarios wird deshalb davon ausgegangen, dass synchron kooperiert wird. Die „shared objects“ können dabei sowohl über technisch gekoppelte Objekte realisiert sein als auch über einen einzelnen gemeinsamen Fokus, ohne eine technische Kopplung vorauszusetzen.

Für den Fall einer asynchronen Kooperation wird in der dritten Spalte immer auch der Umgang mit „shared objects“ vorausgesetzt, auch hier nicht nur im technischen Sinn, sondern auch im Sinn einer Wissenskommunikation. Damit wären solche kooperationsunterstützenden Systeme ausgeschlossen, die beispielsweise nur gemeinsame Kalender anbieten.

Tabelle 7 ordnet die Systeme bezüglich ihrer Funktionalitäten ein, die hier als sinnvoll für kooperatives Wissensmanagement angesehen werden. Für die Bewertungen „+“ und „Ø“ wurde fast immer präzisiert, wofür sie vergeben wurden, da die Kriterien relativ allgemein sind und durch die unterschiedlichsten Ansätze ausgefüllt werden können. Ob ein „+“ oder ein „Ø“ vergeben ist, ist eine weitgehend subjektive Einschätzung, die sich daran orientiert, welche Technologien heute üblich oder möglich sind.

Organisatorische Einbettung	Interoperabilität	Repositories	Information-retrieval	Notifikation	Reflexion über den Wissensbestand	Einbettung in nicht-technischen Arbeitsprozess
Belvedere	–	Ø (Lernmaterial)	Ø (Lernmaterial)	–	–	+
CardBoard	–	–	–	–	–	–
Chips	Ø (Integration externer Medien)	Ø (Hypertext)	Ø (Navigation)	Ø (E-Mails)	+	+
Cognoter/Argnoter	–	–	–	–	–	+
Csile	Ø (Integration externer Medien)	+	+	+	+	
Dolphin	Ø (Integration externer Medien)	Ø (Hypertext)	Ø (Navigation)	–	–	–
glBIS	–	+	+	–	+	Ø (im Versuchs-Settings)
MindManager	Ø (Integration externer Medien)	–	–	Ø (E-Mail)	+	–
OneNote	+	Ø (privates Repository)	+	– (nicht kooperativ)	+	–
Sepia	Ø (Integration externer Medien)	Ø (Hypertext)	Ø (Navigation)	–	–	–

Tabelle 7

Einordnung der Systeme danach, wie sie organisatorische Belange unterstützen.

Die Interoperabilität beschränkt sich in den meisten Fällen auf die Integration von externen Medien, was als Minimum angesehen werden muss. Oft handelt es sich oft um große, kompakte Systeme, die Ergebnisse, die mit ihnen erstellt werden, anderen

Systemen nicht zur Verfügung stellen. Unterstützt wird meistens der Prozess zum Wissensprodukt, wobei schließlich die Produkte nur vom Tool selbst gehandhabt werden können. Eine Art der Interoperabilität kann durch Import und Export von Standardformaten erreicht werden. Im Rückgriff auf die visuellen Sprachen können an dieser Stelle sogar „intelligente“ Dokumentationsprozesse angestoßen werden, die sich die Struktur der Wissensprodukte zu eigen machen, um daraus stärker formatierte Dokumente zu generieren oder Inhalte beispielsweise im Web zu publizieren. XML bietet sich heute für solche frei gestaltbaren Dokumente an.

Standardformate sind außerdem dann nützlich, wenn auf allgemeine Suchmaschinen zurückgegriffen werden soll. Dieser Anforderungen kommt von den aufgeführten Systemen keines nach, evtl. nicht einmal OneNote, das für die „Microsoft-Welt“ entwickelt wird.

Nach meiner Ansicht begrenzt die Verwendung einer eigenen Hypertextmaschine wie in Sepia, Chips und Dolphin die Einsatzmöglichkeiten. Die Hypertexte sind nur über die Maschine zugänglich und setzen diese für eine Kooperation voraus. Es sind deshalb technische sowie unter Umständen lizenzrechtliche Probleme zu erwarten, wenn das LAN verlassen wird und im WAN kooperiert werden soll. Eine Ausnahme stellt da das Web als Hypertextplattform dar, das in den genannten Fällen aber nicht als Plattform verwendet wird.

Eine Hypertextmaschine stellt die Navigation zur Verfügung, um den Informationsraum zu durchlaufen. Dabei ist man auf die Güte und Art der Links angewiesen. Für eine Pflege eines Repositories sind aber außerdem navigationsunabhängige Zugänge, z.B. durch Suchmaschinen, zu Inhalten sinnvoll, um neue Verbindungen herstellen zu können. Darin bestand für das Csile-System der Grund, sich für eine zentrale Datenbank zu entscheiden und keinen Hypertextansatz zu verfolgen.

Notifikationen werden kaum in die Arbeitsprozess mit einbezogen. Die einzige Ausnahme bildet das Csile-System. Generell können zwar E-Mails zur Benachrichtigung verwendet werden, manchmal ist diese Funktion auch in die Systeme integriert, sie sind aber nicht unbedingt als besonderes „feature“ zu werten, wenn nicht zumindest unter bestimmten Voraussetzungen Automatisierungen angeboten werden. Benachrichtigungen können einen asynchronen Arbeitsprozess maßgeblich steuern, indem sie auf Änderungen hinweisen, einen Arbeitsbedarf ankündigen, Interessengruppen realisieren, etc.

Vom Engineering-Standpunkt besteht ein deutlicher Bedarf an flexibleren Programmarchitekturen.