Prüfer:	o. Prof. Hon.-Prof. Dr. D. Roller
Betreuer:	Dipl.-Inform. Monika Bihler
Beginn am:	6. Dezember 1995
Beendet am:	5. Juni 1996
CR-Nummer:	H.5, I.7.2, K.6

Diplomarbeit-Nr. 1353

Adaptive Informationsaufbereitung

Annette Maile

Universität Stuttgart
Institut für Informatik
Lehrstuhl Grundlagen der Informatik
- Graphische Systeme -
Breitwiesenstr. 20 - 22
70565 Stuttgart

Kurzfassung

Am Institut für Informatik der Universität Stuttgart werden in der Abteilung *Graphische Systeme* im Rahmen des Projekts **POWER** (**P**roduct modelling in **o**bject-oriented Databases **w**ith **e**fficient Methods for **R**etrieval) die Anforderungen an eine integrierte, umfassende Produktdatenmodellierung untersucht und richtungsweisende Konzepte für ganzheitliche, bereichsübergreifende Informationssysteme im CAD/CAM-Bereich entwickelt.

In dieser Diplomarbeit wird das Konzept eines Architekturmodells für die adaptive Informationsaufbereitung in bereichsübergreifenden, betrieblichen Informationssystemen entwickelt.

Dazu wird zunächst das betriebliche Umfeld untersucht, um die unterschiedlichen Informationsbedürfnisse von Mitarbeitern zu analysieren und Informationsaufbereitungsarten sowohl für verschiedene Funktionsgruppen, als auch innerhalb einer Abteilung zu erarbeiten.

Im Anschluß daran werden die Konzepte neuer Informationssysteme - wie *Hypermedia, intelligente Agenten* und *computerunterstützte Lernsysteme* - daraufhin untersucht, inwieweit sie für die Realisierung der adaptiven Informationsaufbereitung im betrieblichen Umfeld geeignet sind.

Darauf aufbauend wird in Kapitel 4 das Konzept eines Architekturmodells für die adaptive Informationsaufbereitung entwickelt. Das Konzept soll offen sein, so daß es jederzeit um Informationsaufbereitungsarten erweitert werden kann.

Anschließend wird das Architekturmodell anhand einiger Bereiche einer Beispielfirma implementiert.

Die Implementierung der Arbeit erfolgt in *Perl* an HP-Workstations der Serie 9000/700.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

"Der Schlüssel zum Erfolg eines Produktes, sei es ein technisches Produkt oder eine Dienstleistung, liegt in der Akzeptanz der Benutzer." [Böcker et al. 1993, S. 1]

Informationssysteme haben in vielen Bereichen und so auch im betrieblichen Umfeld Einzug erhalten. Dabei besteht noch die Schwierigkeit, daß für jeden Bereich unterschiedliche Systeme, sogenannte Insellösungen, existieren. Dies führt zu einer uneffektiven Informationsverwaltung, da Daten mehrfach und in unterschiedlichen Formaten gespeichert sind. Der Datenaustausch zwischen den verschiedenen Bereichen ist infolgedessen äußerst schwierig. Daher besteht die Notwendigkeit für ein bereichsübergreifendes, einheitliches Informationssystem, welches Datenaustausch und Datenfluß zwischen den Funktionsbereichen ermöglicht. Da ein solches System über eine sehr hohe Funktionalität verfügen muß, um alle Bereiche abdecken zu können, steigt die Datenflut und damit die Unübersichtlichkeit stark an. Manche Anwender sind überfordert, aus dem Informationssystem die für sie relevanten Daten herauszufiltern. Infolge verkürzter Produktlebenszyklen und sich ständig ändernden Markt- und Wettbewerbsverhältnissen müssen Informationen jedoch kurzfristig verfügbar sein.

Aufgrund der unterschiedlichen Anforderungen der Anwender an die Systeme, basierend auf ihren stark variierenden Aufgabenbereichen, Kenntnissen und Wissen in Bezug auf Computersysteme, resultiert die Forderung, die Anwendungssysteme an den Benutzer und an seine Bedürfnisse anzupassen. Dabei sollen die Informationen nicht so dargestellt werden, wie sie abgespeichert sind, sondern individuell an den Anwender angepaßt präsentiert werden, um die Akzeptanz von Informationssystemen im betrieblichen Umfeld zu erhöhen. Die Aufgabe der adaptiven Informationsaufbereitung besteht darin, dem Benutzer aufgrund seines Anwenderprofils die angeforderten Daten aus dem betrieblichen Informationssystem in individueller Form zu präsentieren. Dabei müssen besonders zwei Gesichtspunkte - die Darstellungsform und der Detaillierungsgrad der Information - in Betracht gezogen werden. Die Darstellungsform ist von den unterschiedlichen Aufgabenbereichen und den damit verbundenen Verwendungszwecken

abhängig. Während beispielsweise in der Konstruktionsabteilung eine Konstruktionszeichnung des Produkts benötigt wird, braucht die Marketingabteilung eine dreidimensionale, farbige Graphik für werbewirksame Zwecke. Im Bezug auf den Detaillierungsgrad sind unterschiedliche Ansichten zu unterscheiden. So werden beispielsweise Konstruktionszeichnungen des gesamten Produkts, einer Baugruppe oder eines Einzelteils benötigt. Zudem muß die Möglichkeit bestehen, eine Konstruktionszeichnung in einer Gesamtansicht oder einen Ausschnitt auf Pixelebene bearbeiten zu können.

Ziel dieser Arbeit ist, ein Konzept eines Architekturmodells für die adaptive Informationsaufbereitung im betrieblichen Umfeld zu entwickeln. Das Architekturmodell soll offen und jederzeit um zusätzliche Dateiformate und Filter erweiterbar sein.

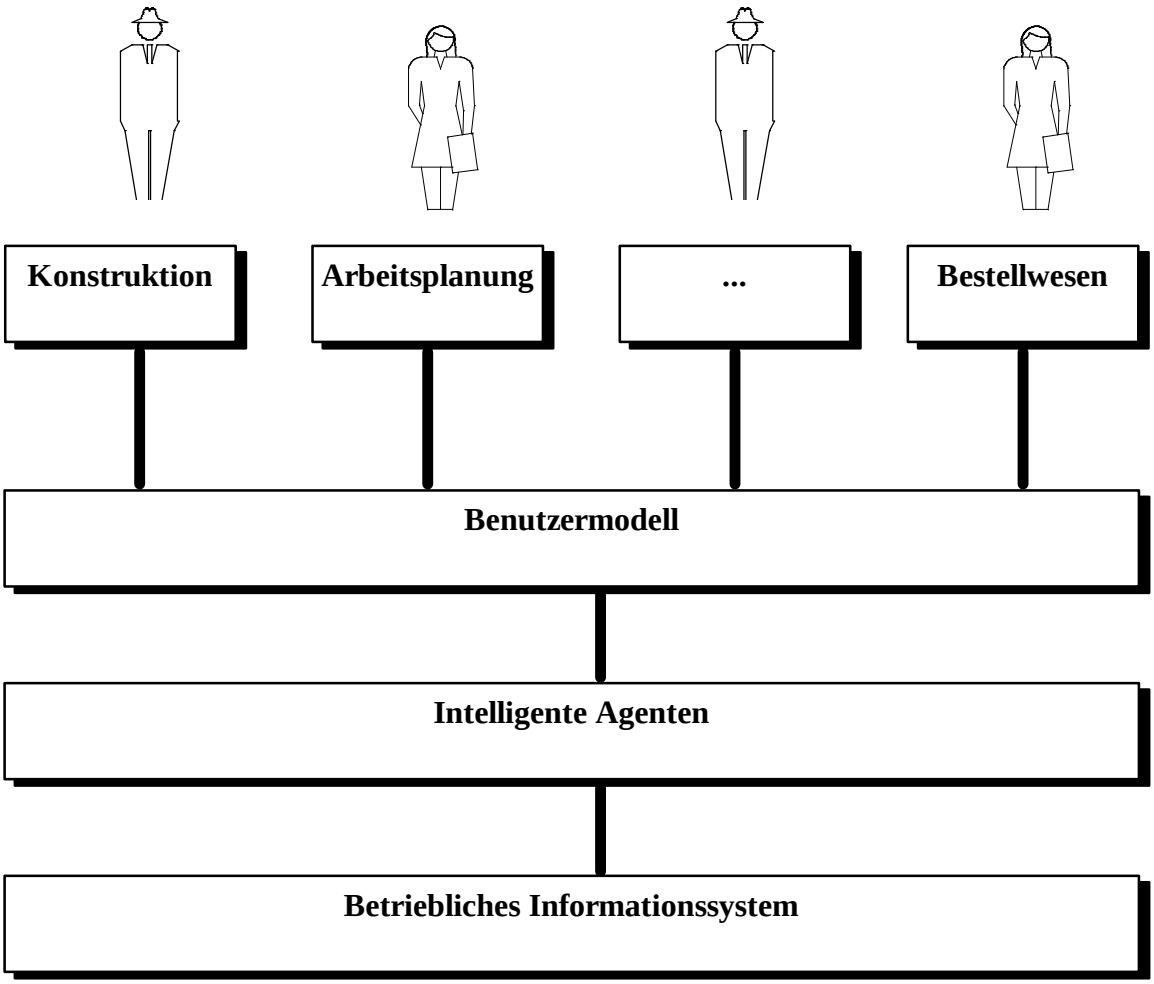


Abbildung 1.1: *Struktur eines Architekturmodells für die adaptive Informationsaufbereitung*

Dabei soll die Darstellung der Informationen mediaunspezifisch sein, d.h., es sollen Text, Graphik, Audio und Video unterstützt werden können.
Abbildung 4.1 zeigt den groben Aufbau des Modells. Die Benutzer oder gegebenenfalls -gruppen werden in einem Benutzermodell beschrieben. Spezielle Programme - *intelligente Agenten* - haben die Aufgabe, basierend auf übergebenen Parametern, Informationen aus dem betrieblichen Informationssystem zu filtern und aufzubereiten. Bei Anfrage eines Benutzers wird aufgrund seines Eintrages im Benutzermodell ein Agent aufgefordert, die gewünschte Information aus dem betrieblichen Informationssystem zu beschaffen und entsprechend aufzubereiten.

Die Grundlage für das Architekturmodell sind die Informationsbedürfnisse der verschiedenen Anwender in einem Unternehmen. Infolgedessen werden in Kapitel 2 einige Funktionsbereiche eines fertigenden Industriebetriebes auf ihre Aufgaben hin untersucht. Dabei werden die jeweils benötigten Daten und ihre Aufbereitungsarten aufgezeigt. Dies ist die Basis für die Zuordnung der verschiedenen Informationsaufbereitungsarten zu den entsprechenden Benutzerprofilen. Anhand einer Beispielfirma wird ein Szenario entwickelt, auf dessen Grundlage später die Implementierung erfolgt.

Im darauffolgenden Kapitel werden die Konzepte neuer Informationssysteme - *Hypermedia*, *intelligente Agenten* und *computerunterstützte Lernsysteme* - auf ihre Verwendbarkeit für die Realisierung adaptiver Informationsaufbereitung untersucht.

In Kapitel 4 wird darauf aufbauend ein Konzept eines Architekturmodells für die adaptive Informationsaufbereitung entwickelt. Zunächst werden die Bestandteile des Modells allgemein und anschließend konkret für die Beispielfirma beschrieben.

In Kapitel 5 erfolgt die Implementierung des Modells anhand der Beispielfirma.

2 Betriebliches Umfeld

In diesem Kapitel werden ausgewählte Bereiche eines Betriebes auf ihre Aufgaben und Informationsbedürfnisse hin untersucht, um mögliche Einsatzgebiete der adaptiven Informationsaufbereitung im betrieblichen Umfeld zu ermitteln. Dies erfolgt deshalb nur auszugsweise, da eine Gesamtanalyse im Rahmen dieser Diplomarbeit zu umfangreich wäre und zudem die einzelnen Bereiche je nach Art des Betriebes variieren.
Zunächst werden im Überblick die Aufgabenbereiche der selektierten Unternehmenszweige aufgezeigt. Daraus resultieren die benötigten und zu erstellenden Daten der jeweiligen Funktionsgruppen. Diese Daten sind eine wichtige Basis für die adaptive Informationsaufbereitung, da die von den jeweiligen Benutzergruppen benötigten Daten mit Hilfe einer geeigneten Benutzermodellierung in zweckmäßiger Form dargestellt werden sollen. Hierzu ist zunächst zu untersuchen, welche Aufbereitungsarten benötigt werden.
Am Beispiel einer fiktiven Firma werden die aufgezeigten Aufgabenstellungen konkretisiert.

2.1 Die fiktive Firma “Baukado“

Die Firma *Baukado* ist ein Unternehmen, das Bauklötze aus Holz herstellt und in Form von drei verschiedenen Baukästen vertreibt. Die unterschiedlichen Baukästen des Sortiments setzen sich aus folgenden Einzelteilen zusammen:

1. Baukasten **Einfach** bestehend aus:

   8 Quadern 48 mm x 24 mm x 24 mm

   8 Würfeln 24 mm x 24 mm x 24 mm

   4 Quadern 48 mm x 24 mm x 12 mm

   4 Quadern 72 mm x 24 mm x 12 mm

   2 “Brücken“ 72 mm x 24 mm x 24 mm und zugehörigen

   2 “Brückenausschnitten“ 54 mm x 20 mm x 24 mm

2. Baukasten **Standard** bestehend aus:

   Bauklötzen aus *Einfach*

   1 Holzwagen, in den die Bauklötze passen

3. Baukasten **Exklusiv** bestehend aus:

   Bauklötzen aus *Einfach*

   1 Lokomotive mit 2 Holzanhängern, in die die Bauklötze passen

Jeder Baukasten enthält bunte Bauklötze in den Farben rot, gelb, blau und grün. Die Farbgebungen können dabei variieren. Abbildung 2.1 zeigt die verschiedenen Bauklötze aus dem Baukasten *Einfach*.

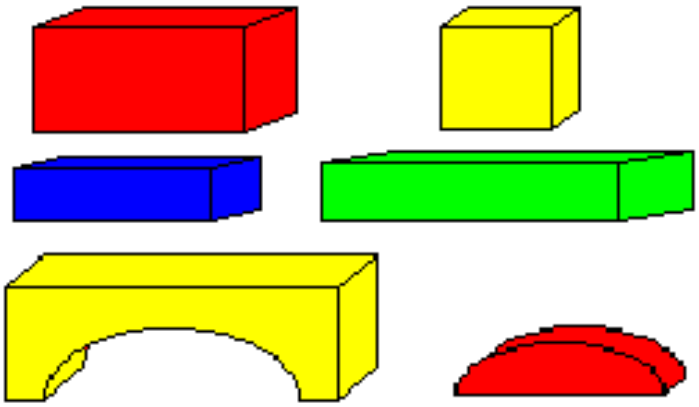


Abbildung 2.1: *Bauklötze aus Baukasten Einfach*

Den gesamten Baukasten *Standard* mit zugehörigem Wagen zeigt Abbildung 2.2.

Die Aufgabenbereiche von *Baukado* liegen in der Konstruktion und Fertigung der Bauklötze und Wagen, Verpackung und Vertrieb der Baukästen etc., wobei der Ablauf aller zum Produktionsprozeß gehörenden Arbeitsgänge optimal gestaltet sein soll.

Abbildung 2.2: *Baukasten Standard*

Im nächsten Abschnitt werden nun die Aufgaben ausgewählter Funktionsgruppen - *Forschung und Entwicklung*, *Fertigung*, *Beschaffung und Materialwirtschaft* und *Rechnungswesen* - eines Unternehmens untersucht und Teilbereiche beispielhaft anhand der Firma *Baukado* konkretisiert.

2.2 Betriebliche Funktionen

Ein Betrieb setzt sich aus mehreren Funktionsbereichen zusammen, die für die Entwicklung bis zum Verkauf eines Produkts notwendig sind. Zugrunde gelegt wird ein Industriebetrieb mit Fertigung. Zu untersuchen sind die Anforderungen an die benötigten und zu erstellenden Daten der einzelnen Unternehmensbereiche für eine durchgängige, integrierte Produktentwicklung. Dies ist die Basis für die adaptive Informationsaufbereitung, da die verschiedenen Arten von Informationen erst einmal bereitgestellt werden müssen, um sie nach den Anforderungen der einzelnen Benutzer entsprechend aufzubereiten. Zudem sollen auch die potentiellen Datenflüsse zwischen den verschiedenen Unternehmensbereichen aufgezeigt werden.
Da der Umfang einer Gesamtanalyse der Funktionsgruppen eines Unternehmens den Rahmen dieser Diplomarbeit überschreitet, werden beispielhaft folgende Bereiche untersucht:

1. **Forschung und Entwicklung**
2. **Fertigung**
3. **Beschaffung und Materialwirtschaft**
4. **Rechnungswesen**

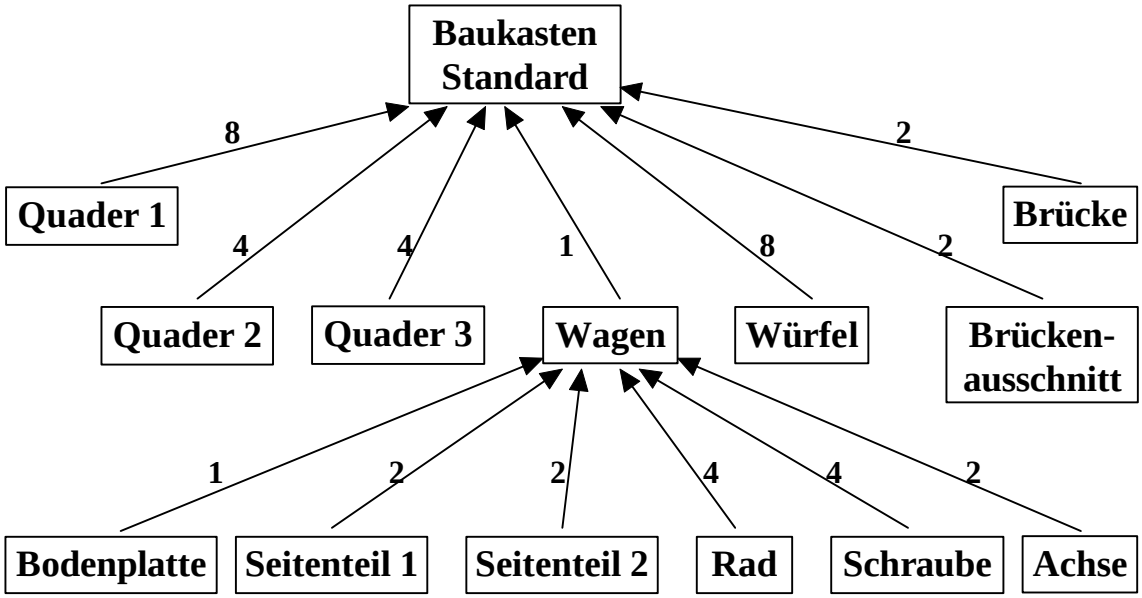
Eine ausführliche Beschreibung der verschiedenen Funktionsgruppen findet der interessierte Leser in [Mertens et al. 1992], [Schwarze 1994] und [Steinbuch 1991].

2.2.1 Forschung und Entwicklung

Die Aufgabenbereiche des Betriebszweiges *Forschung und Entwicklung* umfassen neben der Weiterentwicklung bzw. Neuentwicklung von Produkten auch Eignungsprüfungen von Rohstoffen und Materialien sowie Vorgaben von Produktionstechniken. Beispielsweise ist bei der Firma *Baukado* bei der Auswahl der Lacke darauf zu achten, daß sie toxikologisch unbedenklich sind, damit keine Gefahr für Kleinkinder besteht.

In der Konstruktionsabteilung wird mit Hilfe von **CAD**-Systemen (**C**omputer **A**ided **D**esign) eine Zeichnung des Produkts erstellt, wobei, wenn möglich, Normteile und vorhandene Teile verwendet werden sollen. Bei der Firma *Baukado* werden beispielsweise alle Einzelteile, d.h., die verschiedenen Bauklötze und Wagen, mit allen benötigten Maßangaben gezeichnet. Außerdem werden Zeichnungen mit der Anordnung der Bauklötze in den entsprechenden Wagen erstellt.
Ebenso werden Stücklisten der Erzeugnisse erstellt, d.h., das Produkt wird in seine Bestandteile zerlegt. Der sogenannte Gozintograph stellt dar, wie sich ein Endprodukt struktur- und mengenmäßig aus Zwischenprodukten und Rohstoffen zusammensetzt.
Abbildung 2.3 zeigt einen Gozintographen für den Baukasten *Standard*. Dieser besteht aus:

8 Stück Quader 1	48 mm x 24 mm x 24 mm
4 Stück Quader 2	48 mm x 24 mm x 12 mm
4 Stück Quader 3	72 mm x 24 mm x 12 mm
1 Wagen	
8 Stück Würfel	24 mm x 24 mm x 24 mm
2 Stück Brückenausschnitt	48 mm x 20 mm x 24 mm
2 Stück Brücke	72 mm x 24 mm x 24 mm



End-, Zwischenprodukt, Rohstoff

a $\xrightarrow{n}$ b b besteht aus n-mal a

Abbildung 2.3: *Gozintograph für den Baukasten Standard*

Der Wagen wiederum besteht aus:

1 x Bodenplatte	250 mm x 144 mm x 8 mm
2 x Seitenteil 1	220 mm x 24 mm x 8 mm
2 x Seitenteil 2	128 mm x 24 mm x 8 mm
4 x Rad	Ø 50 mm x 12 mm
4 x Schraube	M4 x 40
2 x Achse	150 mm x 18 mm x 18 mm

Stücklisten werden zur besseren Lesbarkeit normalerweise in Tabellen verwaltet. Aus dem Gozintographen können die Gesamtstückzahlen aller Einzelteile ermittelt werden,

die für die Erstellung des Produkts notwendig sind. Abbildung 2.4 zeigt die Stückliste für den Baukasten *Standard* mit den zugehörigen Teilenummern.

Stückliste Baukasten Standard		
Teil	**Teile-Nr.**	**Anzahl**
Quader 1	500 001	8
Quader 2	500 002	4
Quader 3	500 003	4
Würfel	500 004	8
Brückenausschnitt	500 005	2
Brücke	500 006	2
Bodenplatte	500 007	1
Seitenteil 1	500 008	2
Seitenteil 2	500 009	2
Rad	500 010	4
Schraube	500 011	4
Achse	500 012	2

Abbildung 2.4: *Stückliste des Baukasten Standard*

Mit Hilfe von **CAE** (**C**omputer **A**ided **E**ngineering) wird das Produkt als Modell im Rechner abgebildet. Auf der Grundlage verschiedener Berechnungen können unterschiedliche Auswirkungen simuliert werden. Durch diese Vorgehensweise wird oftmals die Herstellung eines realen Prototypen hinfällig.
Bei der Firma *Baukado* kann beispielsweise simuliert werden, wie sich bei Änderung der Größe der Bauklötze entsprechend die Größe des Wagens ändern muß.

Die Potentiale für die adaptive Informationsaufbereitung liegen im Unternehmensbereich *Forschung und Entwicklung* in der Bereitstellung von Konstruktionszeichnungen der einzelnen Bauklötze und Wagen, um daraus Baukästen zu konstruieren. Zudem sollen Stücklisten angefordert werden können, wobei komplexe CAD-Programme die

Stücklisten automatisch erstellen, d.h., bei Änderung der Zeichnungen werden auch die Stücklisten aktualisiert.

2.2.2 Fertigung

Die Aufgabe dieses Bereichs besteht darin, neben der eigentlichen Produktion auch den Ablauf der Fertigung aller Teile zu optimieren. Unterstützung erhält dieser Sektor dabei in der Regel von **PPS** - Systemen (**P**roduktions**P**lanung und -**S**teuerung).

Die Ziele sind u.a.:

- hohe Kapazitätsauslastung
- geringe Durchlaufzeiten
- niedrige Lagerbestände
- Einhaltung von Terminen

Vor Fertigungsbeginn werden zunächst Arbeitspläne mit Hilfe computerunterstützter Planung (**CAP**) erstellt werden, die Informationen darüber enthalten, wie und in welcher Reihenfolge die einzelnen Teile zu bearbeiten sind und welche Betriebsmittel eingesetzt werden müssen. Bei der computerunterstützten Planung wird in *Dialogarbeitsplanung* und *vollmaschinelle Arbeitsplanung* unterschieden. In der Dialogarbeitsplanung werden bestehende, ähnliche Arbeitspläne und alle erforderlichen Daten, wie beispielsweise Stücklisten und Fertigungsdaten aus der Entwicklungsabteilung, vom Rechner bereitgestellt. Die Aufgabe des Bearbeiters besteht darin, die Pläne zu aktualisieren und eventuelle Sonderfälle zu erkennen. Bei der vollmaschinellen Arbeitsplanung erarbeitet der Rechner mit Hilfe von Planungslogik, Entscheidungstabellen und Expertensystemen die Arbeitspläne. Voraussetzung hierfür sind die Daten aus der Konstruktion, wie Form, Maße, Material etc. der Teile. Die vollmaschinelle Arbeitsplanung ist hauptsächlich für die Fertigung mechanischer Teile geeignet [Steinbuch 1991].

Bei *Baukado* sieht die Erstellung des Arbeitsplanes für den *Quader 1* mit den Maßen 48 mm x 24 mm x 24 mm folgendermaßen aus:

1. Abrichten des Rohstoffes Kantholz ungehobelt 30 mm x 30 mm x 3000 mm auf die Maße 24 mm x 24 mm mit Hilfe eines Dickenhobels, um glatte und winklige Werkstücke zu erhalten. Benötigte Zeit: 3 Minuten.
2. Fasen der Längskanten mit Hilfe einer Fräsmaschine, um die durch das Abrichten entstandenen scharfen Kanten zu brechen. Zeit: 2 Minuten.
3. Ablängen auf 48 mm mit Hilfe einer Kappsäge. Zeit: 20 Sekunden.
4. Fasen der Querkanten mit Hilfe einer Fräsmaschine. Zeit: 1 Minute.
5. Schleifen aller Kanten mit Hilfe einer Tellerschleifmaschine. Zeit: 2 Minuten.
6. Grundieren der Holzstücke durch Eintauchen in ein Farbbecken mit Grundierungsfarbe. Zeit: 10 Sekunden.
7. Trocknen der Holzstücke in einer Trocknungsanlage. Zeit: 10 Minuten.
8. Feinschliff der grundierten Holzstücke. Zeit: 2 Minuten.
9. Lackieren der Holzstücke durch Eintauchen in ein Farbbecken mit Buntlack. Zeit: 10 Sekunden.
10. Erneutes Trocknen der Holzstücke. Zeit: 10 Minuten.
11. Nachlackieren der Kontaktpunkte der Tauchvorrichtung. Zeit: 10 Sekunden

Der Arbeitsplan wird in Form einer Tabelle mit den Spalten *Nr.* (Reihenfolge), *Arbeitsgang*, *Werkstück*, *Arbeitsplatz* und benötigte *Zeit* ausgegeben (Abbildung 2.5).

Nach der Fertigstellung der Arbeitspläne ermitteln PPS-Systeme zunächst den Primärbedarf, d.h., den Bedarf an Ressourcen. Auf der Grundlage von Absatzprognosen für die Endprodukte wird die Anzahl der zu fertigenden Teile aus den Erzeugnisstrukturen berechnet. Aus den Arbeitsplänen werden alle Arbeitsplätze ermittelt, die zur Fertigung benötigt werden. Das PPS-System schätzt nun ab, ob gewisse Kapazitäten über- oder unterlastet sind.

Die Materialbedarfsplanung ist meist auch in PPS-Systemen enthalten. Dies wird ausführlich in Kapitel 2.2.3 beschrieben.

Arbeitsplan Quader 1				
Nr.	**Arbeitsgang**	**Werkstück**	**Arbeitsplatz**	**Zeit**
1	Abrichten	Kantholz 30 x 30 x 3000	Dickenhobel	1 min
2	Fasen	Kantholz 24 x 24 x 3000	Fräsmaschine	2 min
3	Ablängen	Kantholz 24 x 24 x 3000	Kappsäge	2 s
4	Fasen	Holzstücke 48 x 24 x 24	Fräsmaschine	5 s
5	Schleifen	Holzstücke 48 x 24 x 24	Tellerschleifmaschine	10 s
6	Grundieren	Holzstücke 48 x 24 x 24	Farbbecken 1	2 s
7	Trocknen	Holzstücke 48 x 24 x 24	Trocknungsanlage	10 min
8	Feinschliff	Holzstücke 48 x 24 x 24	Werkbank	20 s
9	Lackieren	Holzstücke 48 x 24 x 24	Farbbecken 2	2 s
10	Trocknen	Holzstücke 48 x 24 x 24	Trocknungsanlage	10 min
11	Nachlackieren	Holzstücke 48 x 24 x 24	Werkbank	10 s

Abbildung 2.5: *Arbeitsplan für die Erstellung von Quader 1*

Eine weitere Aufgabe von PPS-Systemen ist die Durchlaufterminierung, d.h., die Ermittlung der Starttermine der einzelnen Arbeitsgänge. Eine Methode, die sogenannte Rückwärtsterminierung, ermittelt vom gegebenen Endtermin aus rückwärts, unter Berücksichtigung aller Arbeitsvorgänge, den Starttermin.

Beispiel:

Abbildung 2.6 zeigt die Berechnung des Starttermins für die Fertigung des *Quaders 1* bei vorgegebenem Endtermin. Der Starttermin für *Arbeitsgang 11* ist gleichzeitig der Endtermin für *Arbeitsgang 10*. Starttermin für *Arbeitsgang 10* ist wiederum Endtermin für *Arbeitsgang 9 etc.* Der so ermittelte Starttermin für *Arbeitsgang 1* liegt 23 min und 41 s vor dem gegebenen Endtermin.

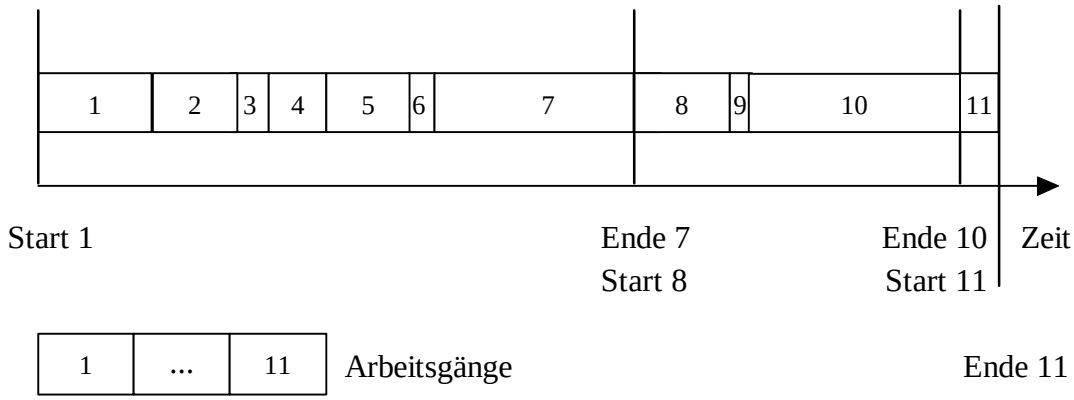


Abbildung 2.6: *Bestimmung des Starttermins für Quader 1*

Ein weiteres Ziel der Optimierung der Produktion ist die gleichmäßige Kapazitätsauslastung bei gleichzeitiger Einhaltung aller Termine. Arbeitsplatzüber- bzw. -unterlastungen müssen frühzeitig erkennbar sein, damit Maßnahmen ergriffen werden können, diese zu umgehen. Die Über- bzw. Unterlastungen lassen sich aus den verfügbaren und benötigten Kapazitäten ermitteln.
Aus der Durchlaufterminierung und der Kapazitätsauslastung resultiert die Werkstattsteuerung. Diese legt fest, welcher Auftrag zu welcher Zeit an einem Arbeitsplatz bearbeitet wird.

Die eigentliche Fertigung erfolgt mit Hilfe von **CAM** (**C**omputer **A**ided **M**anufacturing). Mittels CNC- oder DNC-Maschinen werden Teile physisch gefertigt und eventuell von Robotern montiert bzw. transportiert. Die Verpackung der Endprodukte gehört gegebenenfalls ebenso zur Fertigung.

Die computergestützte Qualitätssicherung der Produkte erfolgt durch **CAQ** (**C**omputer **A**ided **Q**uality Assurance). Entweder werden alle Produkte auf Qualität geprüft oder nur Stichproben entnommen. Des weiteren werden nicht nur Endprodukte, sondern schon Vorstufen überprüft. Wenn ein Zwischenprodukt schon den Qualitätsansprüchen nicht genügt, so muß es nicht noch weiter verarbeitet werden, um dann festzustellen, daß das Endprodukt ebenfalls unter der akzeptablen Grenze liegt.

Der Sektor *Fertigung* benötigt die Informationen in Form von Stücklisten und Fertigungsdaten aus der Entwicklungsabteilung. Mit Hilfe bestehender, ähnlicher Arbeitspläne können dann neue Arbeitspläne erstellt werden, die wiederum Grundlage für Maschinenbelegungspläne und Durchlaufterminierung sind.

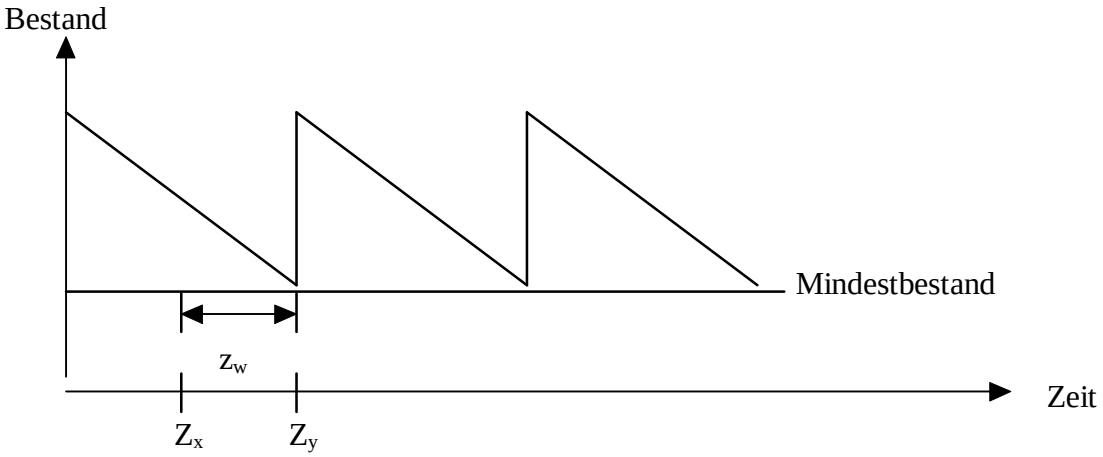
2.2.3 Beschaffung und Materialwirtschaft

Die Funktionen von *Beschaffung und Materialwirtschaft* umfassen die Bedarfsermittlung für eine bestimmte Periode, das Bestellwesen, die Lagerbestandsführung und die Materialbewertung.

Anhand von Stücklisten und Produktionsplänen wird der Bedarf aller Materialien bestimmt. Die Bestelldisposition ermittelt mit Hilfe der Bedarfsmengen, den Lagerbeständen und Mindestbeständen der einzelnen Materialien die Bestellmengen. Vereinfacht setzt sich die Bestellmenge aus den Bedarfsmengen abzüglich des aktuellen Bestandes und zuzüglich des Mindestbestandes zusammen. Die so ermittelte Bestellmenge ist später auch maßgebend für die Wareneingangskontrolle, die die ankommende Ware auf Menge und Qualität überprüft. Dieselben Daten benötigt auch die Buchhaltung zur Prüfung von Lieferantenrechnungen.

Der Bestellzeitpunkt kann folgendermaßen ermittelt werden:

Auf der Grundlage von Lagerabgangsprognosen, die aus der vorherigen Periode resultieren, wird der Zeitpunkt bestimmt, an dem der Bestand seine vorgegebene Minimalgröße erreichen wird. Dieser Zeitpunkt abzüglich der Wiederbeschaffungszeit ergibt dann den Bestellzeitpunkt. Dies wird in Abbildung 2.7 anschaulich dargestellt.



z_w: Wiederbeschaffungszeit
Z_y: Zeitpunkt, an dem Bestand = Mindestbestand ist
Z_x: Tag Z_y - z_w = Bestellzeitpunkt

Abbildung 2.7: *Bestimmung des Bestellzeitpunkts*

Bei *Baukado* wird zunächst ermittelt, wieviel Rohstoff für die Fertigung eines Baukastens erforderlich ist. Dafür wird zuerst der Bedarf für die verschiedenen Bauklötze berechnet und mit der entsprechenden Anzahl multipliziert. Abbildung 2.8 zeigt die Berechnung des Bedarfs an Holz für den Baukasten *Einfach*. Der *Bedarf* bedeutet dabei die benötigte Menge an Rohstoff, d.h., die Länge eines Kantholzes 30 x 30 x 3000 mm, zur Fertigung der verschiedenen Bauklötze.
Bei *Quader 1* kann aus einem Stück Kantholz der Länge 50 mm ein Stück gefertigt werden. Bei *Quader 2* hingegen können aus einem Stück von 50 mm Länge zwei Stück erstellt werden, indem man den zuerst erhaltenden *Quader 1* einmal teilt, da *Quader 2* nur halb so tief ist, wie *Quader 1*. Entsprechend gilt für die weiteren Bauklötze, daß aus 75 mm Rohstoff zwei Q*uader 3*, aus 25 mm 1 *Würfel* und aus 75 mm je eine *Brücke* und je ein *Brückenausschnitt* gefertigt werden können. Unter Umständen kann es kostengünstiger sein, die *Brücke* und das Teil *"Brückenausschnitt"* separat aus zwei Holzstücken zu fertigen. Im Fallbeispiel wird aber angenommen, daß aus einem Holzstück beide Teile hergestellt werden.

Bezeichnung	**Bedarf**	**daraus herstell-bare Anzahl**	**Gesamtbedarf für Baukasten *Einfach***
Quader 1	50 mm	1	400 mm
Quader 2	50 mm	2	100 mm
Quader 3	75 mm	2	150 mm
Würfel	25 mm	1	200 mm
Brücke u. Brückenausschnitt	75 mm	je 1	150 mm
		Summe	**1000 mm**

Abbildung 2.8: *Materialbedarfsberechnung für den Baukasten Einfach*

Nun muß der Bedarf für die Fertigung eines gesamten Baukastens ermittelt werden. Bei *Quader 1* ist der Bedarf für den gesamten Baukasten **8 x 50 mm = 400 mm.**

Bei *Quader 2* ist der Bedarf **2 x 50 mm = 100 mm.**

Aus den Summen aller Einzelteile ergibt sich ein Gesamtbedarf von 1000 mm für den Baukasten *Einfach*. Dies bedeutet wiederum, daß aus einem Kantholz mit den Maßen 30 x 30 x 3000 mm drei Baukästen *Einfach* gefertigt werden können.

Der nächste Schritt wäre nun die Ermittlung des Bedarfs für die Herstellung des Wagens und der Bedarf an Grundierungsfarbe, Lack etc. Dies soll hier der Einfachheit halber außer Acht gelassen werden.
Für die Berechnung der Bestellmenge werden Bedarfsmengen, Bestände und Minimalbestände benötigt.

- In der nächsten Periode sollen 1800 Baukästen *Einfach* gefertigt werden, d.h., es werden 600 Kanthölzer 30 x 30 x 3000 mm benötigt.
- Der Bestand an Kanthölzern sei 500 Stück am 01.03.1996.
- Der Minimalbestand für Kanthölzer sei 100 Stück.

Die Bestellmenge errechnet sich folgendermaßen:

Bestellmenge = Bedarfsmenge - Bestand + Minimalbestand

Bestellmenge = 600 Stück - 500 Stück + 100 Stück = 200 Stück

Für die Berechnung des Bestellzeitpunktes werden Lagerabgangsprognosen, Bestände, Minimalbestände und Wiederbeschaffungszeit benötigt.

- Die Lagerabgangsprognose sei für die nächste Periode 60 Baukästen *Einfach* pro Tag, d.h., es werden 20 Kanthölzer 30 x 30 x 3000 mm pro Tag benötigt.
- Der Bestand an Kanthölzern sei 500 Stück am 01.03.1996.
- Der Minimalbestand für Kanthölzer sei 100 Stück.
- Die Wiederbeschaffungszeit für den Rohstoff betrage 5 Tage.

Auf der Grundlage dieser Daten kann nun der Zeitpunkt bestimmt werden, an dem der Bestand seinen Minimalbestand erreicht:

500 St. - 100 St. / 20 St./Tag = 20 Tage

d.h., am 21.03.96 ist der Bestand gleich dem Minimalbestand.

Der Bestellzeitpunkt errechnet sich aus dem errechneten Datum abzüglich der Wiederbeschaffungszeit:

21.03.1996 - 5 Tage = 16.03.1996

Der nächste Schritt ist die Auswahl eines geeigneten Lieferanten. Die Informationen werden den Lieferantenstammdaten entnommen. Bei der Bestimmung des Lieferanten spielt nicht nur die Verfügbarkeit und der Preis der Materialien eine Rolle, sondern auch die Zuverlässigkeit. Bei Überschreitung von Lieferterminen wird ein Eintrag in die Stammdaten vorgenommen, was die Lieferantenwahl beim nächsten Mal beeinflussen wird.

Eine weitere Aufgabe des Bereichs *Beschaffung und Materialwirtschaft* umfaßt die Lagerbestandsführung, die sich mit der Buchung sämtlicher Lagerbewegungen, d.h., Abgänge und Zugänge befaßt. Der neue Bestand ergibt sich aus:

neuer Bestand = alter Bestand + Zugänge - Abgänge

Zu festgesetzten Zeitpunkten muß eine Inventur durchgeführt werden. Dadurch werden Bestandsdifferenzen ermittelt und gegebenenfalls Bestände korrigiert.
Neben der mengenmäßigen Bestimmung aller Materialien spielt auch die wertmäßige Bestimmung eine wichtige Rolle. Diese Werte sind die Grundlage für die Kostenrechnung. Die Bewertungsansätze, beispielsweise Preise aus Lieferantenrechnungen, werden den Materialstammdaten entnommen.

Der Bereich *Beschaffung und Materialwirtschaft* benötigt zunächst Stücklisten, um den Bedarf an Rohstoffen für die Fertigung eines Baukastens zu ermitteln. Aus Produktionsplänen werden die geplanten Mengen der zu fertigenden Baukästen entnommen. Unter Berücksichtigung von Beständen, Minimalbeständen und Lieferzeiten können daraus Bestellmengen und Bestellzeitpunkte bestimmt werden.
In diesem Bereich sind bei Abfrage von Informationen über die Rohstoffe beispielsweise nur die Bestände interessant und nicht deren Beschaffenheit.

2.2.4 Rechnungswesen

Die Aufgabenbereiche des Sektors *Rechnungswesen* umfassen hauptsächlich Kosten- und Leistungsrechnung und die Buchhaltung. Dabei ist die Kosten- und Leistungsrechnung aufgegliedert in Kostenartenrechnung, Kostenstellenrechnung und Kostenträgerrechnung.
Im folgenden werden die Aufgaben dieser Bereiche behandelt, wobei die Kostenträgerrechnung anhand der Beispielfirma detailliert beschrieben wird.

Kostenartenrechnung

Die Kostenartenrechnung liefert die Basis für die Kostenstellen- und Kostenträgerrechnung. Die Daten aus der Finanzbuchhaltung werden übernommen und nach Art der Kosten systematisiert. Die gesamten angefallenen Kosten werden in ihre fixe und variable Kosten gegliedert.

Kostenstellenrechnung

In der Kostenstellenrechnung werden die Kostenarten den jeweiligen Kostenstellen zugeordnet, um innerbetriebliche Leistungen zu verrechnen. Zusätzlich können die angefallenen tatsächlichen Kosten den Plankosten und den Leistungen gegenüber gestellt werden.

Kostenträgerrechnung

Die Kostenträgerrechnung befaßt sich mit der Kalkulation der Kosten der Produkte. In der Vorkalkulation werden die voraussichtlichen Kosten für die Herstellung der Produkte ermittelt.
Die benötigten Daten für die Aufstellung der Herstellkosten werden den Artikelstammdaten, Stücklisten und Arbeitsplänen entnommen.
Die Herstellkosten berechnen sich wie folgt:

Herstellkosten =	Materialeinzelkosten
+	Materialgemeinkosten
+	Fertigungslöhne
+	Maschinenkosten
+	Restgemeinkosten

dabei sind:

- Materialeinzelkosten = Menge x Wert

  Die Menge ist der Bedarf an Rohstoffen und stammt aus der Materialbedarfsrechnung. Der Wert wird den Materialstammdaten entnommen.

- Materialgemeinkosten

  resultieren aus der Kostenstellenrechnung.

- Fertigungslöhne = Minuten x Lohnsatz

  Die Minuten sind durch die Arbeitspläne gegeben. Den Lohnsatz erhält man aus der Kostenstellenrechnung.

- Maschinenkosten = (Menge x Stückzeit + Rüstzeit) x Maschinenstundensatz

  Die Menge wird den Stücklisten, die Stückzeit und die Rüstzeit den Arbeitsplänen und der Maschinenstundensatz der Kostenstellenrechnung entnommen.

- Restgemeinkosten

  sind in den Kostenstellen festgesetzt.

Am Beispiel des *Quaders 1* sollen nun dessen Herstellkosten berechnet werden. Der Einfachheit halber werden die Herstellkosten für 60 Stück *Quader 1* errechnet, da diese aus einem Kantholz 30x30x3000 mm gefertigt werden können.

Materialeinzelkosten

1 Kantholz 30 x 30 x 3000 mm x 3,50 DM = 3,50 DM

Materialgemeinkosten

Die Materialgemeinkosten seien 1,73 DM.

Fertigungslöhne

Die Fertigungslöhne beziehen sich nur auf die Arbeitsgänge an der Werkbank, d.h., für Feinschliff und Nachlackieren. Die anderen Arbeitsgänge erfolgen vollautomatisch und werden daher bei den Maschinenkosten berücksichtigt.

Die Zeiten für die beiden Arbeitsgängen ergeben sich aus:

Feinschliff	60 x 20 s = 20 min
Nachlackieren	60 x 10 s = 10 min
Gesamtzeit	30 min
Der Lohnsatz sei	50 DM / h

Fertigungslohn = 30 min x 50 DM / h = 25,00 DM

Maschinenkosten

Die Maschinenkosten für die übrigen Arbeitsgänge errechnen sich wie folgt, wobei die Rüstzeiten nicht berücksichtigt sind:

Abrichten	1 min x 50 DM / h = 0,83 DM
Fasen	2 min x 50 DM / h = 1,67 DM
Ablängen	59 x 2 s x 50 DM / h = 1,63 DM
Fasen	60 x 5 s x 50 DM / h = 4,16 DM
Schleifen	60 x 10 s x 50 DM / h = 8,33 DM
Grundieren	60 x 2 s x 50,00 DM / h = 1,67 DM
Trocknen	60 x 10 min x 0,10 DM / h = 1,00 DM

Lackieren	60 x 2 s x 3 DM / h = 1,67 DM
Trocknen	60 x 10 min x 0,10 DM / h = 1,00 DM
Gesamtkosten	21,96 DM

Restgemeinkosten

Die Restgemeinkosten seien 7,56 DM.

Herstellkosten

Die Herstellkosten berechnen sich nun aus:

Materialeinzelkosten	3,50 DM
Materialgemeinkosten	1,73 DM
Fertigungslöhne	25,00 DM
Maschinenkosten	21,96 DM
Restgemeinkosten	7,56 DM
Gesamtkosten	59,75 DM

Die Herstellkosten für einen *Quader 1* betragen somit

59,75 DM / 60 Stück ≈ 1,00 DM

Die Gesamtherstellkosten für den Baukasten *Einfach* berechnen sich aus der Summe der Herstellkosten aller dazugehörigen Teile.

Die Nachkalkulation berechnet die tatsächlich angefallenen Kosten nach der Fertigstellung des Produkts. Mit einem Soll-Ist-Vergleich können dann die Ergebnisse aus der Vor- und Nachkalkulation verglichen und eventuelle Abweichungen festgestellt und die Ursachen dafür behoben werden.

Eine weitere Funktion des Bereichs *Rechnungswesen* ist die Buchhaltung. Diese unterteilt sich in Neben- und Hauptbuchhaltung. Die Nebenbuchhaltung gliedert sich wiederum in Debitoren-, Kreditoren- und Anlagenbuchhaltung.

Debitorenbuchhaltung

Grundlage für die Debitorenbuchhaltung sind die aus der Rechnungsschreibung hervorgegangenen *offenen Posten Debitoren*. Die Hauptaufgabe ist die Überwachung der Zahlungseingänge. Bei Zahlungseingang eines Rechnungsbetrages ist dieser Satz von den offenen Posten zu entfernen. Bei Überschreitung von fälligen Zahlungsterminen werden Mahnungen an die Kunden geschickt.

Kreditorenbuchhaltung

Basis der Kreditorenbuchhaltung sind eingegangene Rechnungen. Die Rechnungen werden zunächst daraufhin geprüft, ob die berechnete Ware bestellt und auch geliefert wurde. Ebenso werden die Rechnungen auf vereinbarte Rabatte und Zahlungsbedingungen untersucht. Danach werden sie in einer Datei *offene Posten Kreditoren* verbucht. Nun muß noch festgelegt werden, nach Fälligkeit oder Priorität, wann die Rechnung bezahlt werden soll.

Anlagenbuchhaltung

Aufgaben der Anlagenbuchhaltung sind die Ermittlung von buchhalterischen und kalkulatorischen Abschreibungen. Die Daten hierfür erhält man aus den Anlagenstammdaten. Ebenso werden mittels dieser Daten Inventarverzeichnisse erstellt.

Hauptbuchhaltung

Die Aufgabe der Hauptbuchhaltung besteht darin, die Sachkonten zu bearbeiten. Sie erhält von der Nebenbuchhaltung Sammelbuchungen und verarbeitet diese weiter. Die Sachkonten sind dabei die Gegenkonten zur Nebenbuchhaltung. Außerdem wird von der Hauptbuchhaltung der Abschluß, d.h., Bilanz und Gewinn- und Verlustrechnung, erstellt.

Der Bereich Rechnungswesen benötigt Daten aus der Finanzbuchhaltung für die Kostenartenrechnung. Diese ist wiederum Grundlage für die Kostenstellenrechnung. Für die Berechnung der Herstellkosten werden Stücklisten, Teildaten aus den Materialstammdaten, Teildaten aus der Kostenstellenrechnung und Arbeitspläne benötigt. Für die Buchhaltung werden Daten aus den Kundenstammdaten, Lieferantenstammdaten etc. benötigt. Die Anforderungen an die adaptive Informationsaufbereitung liegen darin, nur die jeweils relevanten Attribute aus den Stammdaten bereitzustellen.

2.3 Zusammenfassung

In diesem Kapitel wurden die ausgewählten Funktionsgruppen *Forschung und Entwicklung*, *Fertigung*, *Beschaffung und Materialwirtschaft* und *Rechnungswesen* des betrieblichen Umfelds kurz erläutert und anhand von Teilbereichen der Beispielfirma *Baukado* konkretisiert.

Die zum Betrieb gehörenden Daten werden in einem Informationssystem gespeichert. Die verschiedenen Funktionsbereiche eines Unternehmens benötigen jedoch die Informationen in unterschiedlicher Form aufbereitet, beispielsweise als Graphik, Tabelle oder Text. Ebenso werden oft nur Teile der bereitgestellten Informationen, wie beispielsweise einzelne Attribute der Stammdaten, benötigt.

Abbildung 2.9 zeigt beispielhaft, daß verschiedene Abteilungen nur bestimmte Attribute aus den Materialstammdaten benötigen. Entwicklung und Fertigung müssen Zugriff auf *Material*, die Kostenträgerrechnung auf den *Preis* und das Bestellwesen auf den *Bestand* und den *Minimalbestand* haben.

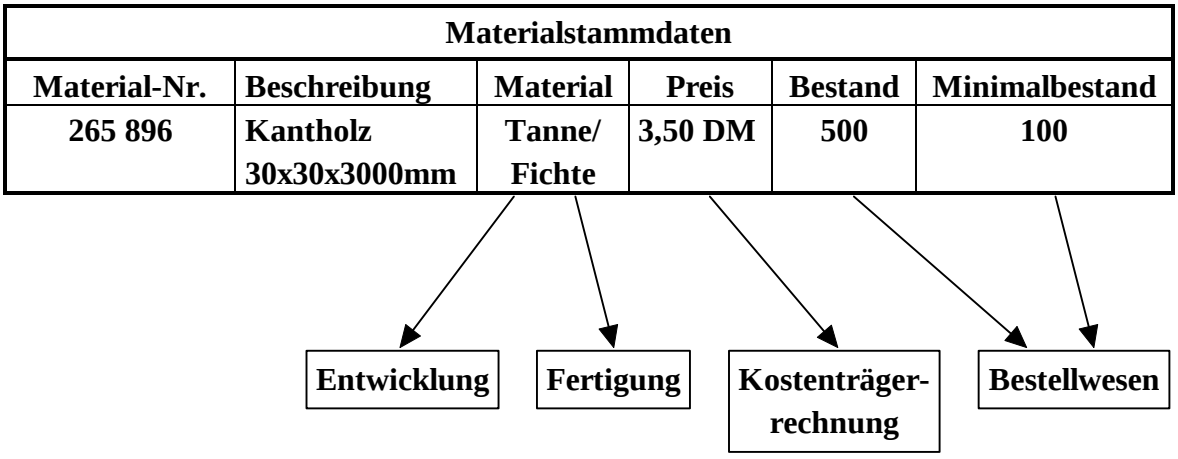


Abbildung 2.9: *Darstellung relevanter Attribute für verschiedene Abteilungen*

Um die Potentiale für die adaptive Informationsaufbereitung herauszuarbeiten, wurden die einzelnen Funktionsbereiche daraufhin untersucht, welche Daten zur Verarbeitung benötigt und welche Daten produziert werden. Auf dieser Grundlage wird abgeleitet, welche Informationen auf Anfrage ausgegeben werden und wie diese aufzubereiten sind.

Im folgenden Kapitel sollen nun Konzepte neuer Informationssysteme, wie *Hypermedia, intelligente Agenten* und *computerunterstützte Lernsysteme*, daraufhin untersucht werden, inwieweit sie für die adaptive Informationsaufbereitung nützlich sind.

3 Konzepte neuer Informationssysteme

Viele der in den letzten Jahren entwickelten Informationssysteme brachten nicht den gewünschten Erfolg in der Praxis. Es genügt nicht, Produktionsdaten lediglich zu speichern, vielmehr muß das Auffinden von Informationen für die Benutzer einfach gestaltet sein, da der "normale" Anwender nicht unbedingt ein Datenbankexperte ist. Daher besteht die Notwendigkeit, neue Konzepte für Informationssysteme zu entwickeln [Bihler et al. 1996]. Hierbei muß beachtet werden, daß nicht nur die Bereitstellung der Daten von Bedeutung ist, sondern auch verschiedene Präsentationsformen der Daten entsprechend den Bedürfnissen der Benutzer ermöglicht und generiert werden müssen. Hier liegen Potentiale für die adaptive Informationsaufbereitung durch benutzergerechte Präsentation der Informationen die Flexibilität und Akzeptanz des Systems zu steigern.
Im folgenden wird dargestellt, welche Komponenten notwendig sind, um ein intelligentes Informationssystem aufzubauen. Anschließend werden die Konzepte der Elemente, die für die Realisierung der adaptiven Informationsaufbereitung geeignet sind, analysiert.

Ein Informationssystem dient der Speicherung, Wiedergewinnung, Verknüpfung und Auswertung von Informationen [Duden Informatik 1993]. Informationssysteme setzen sich aus Datenbanken, Datenbankmanagementsystemen und Anwendungsprogrammen zusammen. In den Datenbanken sind die Daten formatiert und klar strukturiert abgelegt. Die Datenbankmanagementsysteme stellen Funktionen zum Zugriff auf die Daten in den Datenbanken zur Verfügung. Die Anwendungsprogramme fordern über Datenbankmanagementsysteme Daten aus den Datenbanken zur Weiterverarbeitung an.

Um Datenbankanfragen so intuitiv wie möglich zu gestalten, werden neue Konzepte benötigt. Solch ein neues Konzept stellen "intelligente Datenbanken" dar. Sie werden wie folgt definiert:

> *"... intelligent databases can be loosely defined as databases that manage information in a natural way, making that information easy to store, access and use."* [Parsaye et al. 1989 S. 1]

Um intelligente Informationssysteme aufzubauen, müssen weitere Fachgebiete integriert werden. Intelligente Informationssysteme sind eine Mischung aus traditionellen

Datenbanktechnologien mit neuen Fachbereichen, wie *Dokument-Management* und *Information Retrieval*, *Hypermedia* und *Objektorientierung* (siehe Abbildung 3.1). Die Erfahrungen aus der Anwendung der traditionellen Datenbanktechnologien bilden die Grundlage für neue Informationssysteme. Da traditionelle Datenbanken nur einfache Datenstrukturen verwalten können, müssen objektorientierte Ansätze zur Darstellung komplexer Datenstrukturen sowie *Dokument-Management* zur Verwaltung von Dokumenten integriert werden. Die Vorteile der Anwendung von Hypermedia in Informationssystemen liegen in der Darstellungsmöglichkeit der Informationen mit Hilfe unterschiedlicher Medien, wie Text, Graphik, Audio und Video.

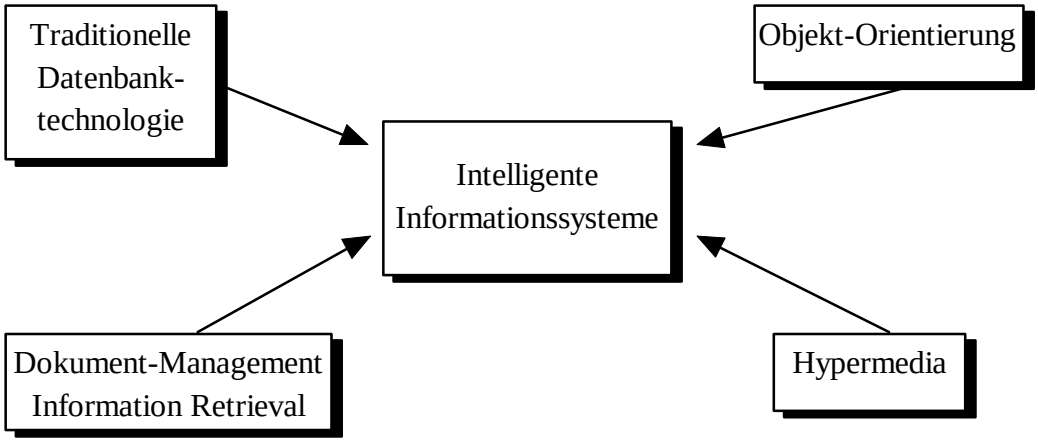


Abbildung 3.1: *Mischung von traditionellen Datenbanktechnologien mit neuen Fachgebieten zu einem neuen Konzept von Informationssystemen*

Aufgrund dessen wird in diesem Kapitel von den oben genannten Fachgebieten das Konzept von Hypermedia ausführlich vorgestellt. Die anderen aufgeführten Bereiche, die sich mit der Speicherung und Wiedergewinnung der Daten befassen, werden in weiteren Arbeiten im Rahmen des Projekts *POWER* behandelt.
Eine weitere Komponente für die Realisierung von adaptiver Informationsaufbereitung ist das Konzept der *intelligenten Agenten.* Diese sind für die Informationsbeschaffung und -aufbereitung zuständig.
Zusätzlich wird die Konzeption von *computerunterstützten Lernsystemen*, die auch adaptive, d.h.,an den Wissensstand angepaßte Informationssysteme sind, vorgestellt. Das Ziel von Lernsystemen ist, wie bei Informationssystemen, den Benutzer von einem uninformierten in einen informierten Zustand zu führen.

Die Konzepte werden daraufhin untersucht, inwieweit sie für die Realisierung von adaptiver Informationsaufbereitung geeignet sind.

3.1 Hypermedia

Hypermedia ist auf der Grundlage von *Hypertext* insofern weiterentwickelt worden, daß zusätzlich zu Text- und Graphikelementen weitere Medien, wie Audio und Video, unterstützt werden. Für die adaptive Informationsaufbereitung stellt *Hypermedia* ein großes Potential dar, da die Datenausgabe in unterschiedlicher Form ermöglicht wird. Die Konzeption der Organisation der Daten ist jedoch dieselbe wie bei *Hypertext*, weshalb nun das Konzept von *Hypertext* vorgestellt wird.

3.1.1 Hypertext

Das Konzept von *Hypertext* ist die Zerlegung eines Textes in kleinere, überschaubare Informationseinheiten, die mit Hilfe eines Graphen verwaltet werden. Die Informationseinheiten, die als Textfragmente realisiert werden, sind die Knoten des Graphen und die Verbindung dieser Knoten stellen die Kanten des Graphen dar. Zur Darstellung der Existenz einer Kante wird ein Wort kenntlich gemacht, indem es beispielsweise eine andere Farbe als der restliche Text aufweist oder unterstrichen ist. Auf Anklicken dieser Markierung erscheint die referenzierte Informationseinheit. Abbildung 3.2 stellt dar, daß bei Anklicken des markierten Worts *Grundlagen der Informatik* des Hypertextdokuments "*Fakultät Informatik*" die referenzierte Informationsseinheit, nämlich das Hypertextdokument "*Grundlagen der Informatik*", erscheint. Durch die Verknüpfungen (*Links*) von Textfragmenten wird die Linearität im Gegensatz zu konventionellen Texten aufgebrochen. Die *Links* sind vergleichbar mit Fußnoten oder Verweisen. Allerdings besteht die Gefahr von *Hypertext* darin, daß sich der Benutzer durch ständige Verzweigung zu weiteren Informationseinheiten im *Hypertext* verliert (auch lost-in-Hyperspace genannt) und damit die ursprünglich gesuchte Information in den Hintergrund tritt.
Eine ausführliche Beschreibung von *Hypertext* wird in [Kuhlen 1991] gegeben.

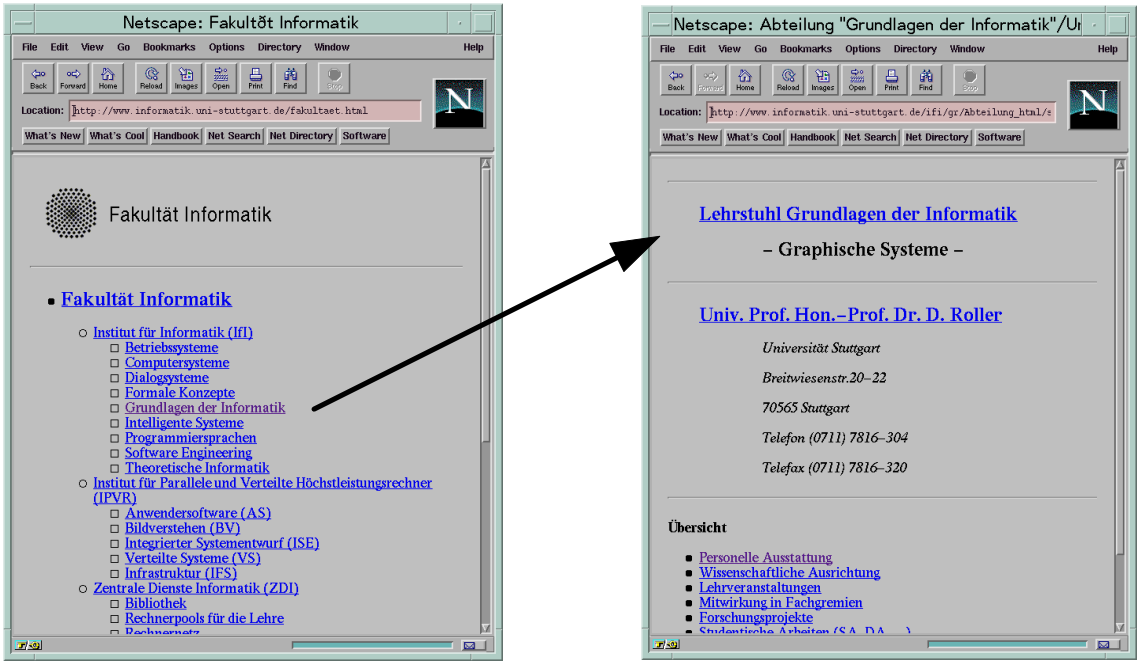


Abbildung 3.2: *Konzept eines Hypertext-Systems*

Die Beschreibung der Hypertextdokumente erfolgt beispielsweise mit *HTML* (**H**yper**T**ext **M**arkup **L**anguage). Mit *HTML* können Überschriften, Textformatierungen, wie **fett**, *kursiv*, unterstrichen etc. und Verweise (sogenannte *Links* und *Anker*) auf andere Hypertextdokumente definiert werden. Ebenso ist die Generierung von Formularen mit Eingabefeldern, Check Boxen, Radio Buttons etc. möglich. Da *HTML* lediglich die logische Struktur, nicht aber das Layout eines Hypertextdokuments festlegt, wird zur Darstellung ein Browser benötigt. Beispiele hierfür sind *NCSA Mosaic* und *Netscape* im *WWW* (**W**orld **W**ide **W**eb). Im folgenden wird nun dieses hypermediale Informationssystem vorgestellt.

3.1.2 World Wide Web

Das World Wide Web (*WWW*) ist ein hypermediales Informationssystem mit weltweitem Zugriff auf Informationen des Internet. Das *WWW* basiert auf dem Client-Server-Prinzip. Der *WWW*-Server stellt Informationen zur Verfügung, die mit einem *WWW*-Client (*WWW*-Browser) abgerufen werden können. Um Kommunikation zwischen dem *WWW*-Server und dem *WWW*-Client zu ermöglichen, wird das **H**yper**T**ext **T**ransmission

Protocol (**HTTP**) benötigt. Der Client sendet eine HTTP-Anfrage an den Server und dieser leitet die gewünschte Information an den Client weiter. Zur eindeutigen Adressierung der Informationseinheiten auf dem Server werden sogenannte **U**niform **R**esource **L**ocators (**URL**s) verwendet.

Ein URL setzt sich aus folgenden Komponenten zusammen:

```
Protokoll://Server/Verzeichnis/Datei
```

Ein Beispiel hierfür ist:

```
http://www.informatik.uni-stuttgart.de/ifi/gr/
 Abteilung_html/seite1.html
```

Dies ist die Adresse (URL) des Hypertextdokuments *"Grundlagen der Informatik"* aus Abbildung 3.2.
Neben der Anforderung von Hypermediadokumenten können auch beispielsweise Informationen aus einer Datenbank abgerufen werden. Dies wird durch HTML-Formulare ermöglicht. Der Formularinhalt wird an den HTTP-Server gesendet und dort von einem ausführbarem Programm weiterverarbeitet. Das Programm generiert dynamisch zur Laufzeit ein Hypermediadokument, das an den Client zurückgeschickt wird. Das Programm kann dabei in einer Shell-Sprache oder Programmiersprache geschrieben sein. Eine ausführliche Beschreibung von Shell-Skripten ist in [Frühwacht 1995] gegeben.
Die Programme bilden die Schnittstelle zwischen dem Client und dem Server. Damit die Schnittstellenprogramme kompatibel zu allen HTTP-Servern sind, unterliegen sie einem gemeinsamen Standard, dem ***C**ommon **G**ateway **I**nterface* (**CGI**).
CGI-Programme sind in Verbindung mit Hypermedia für die adaptive Informationsaufbereitung notwendig. Aufgrund übergebener Parameter durch HTML-Formulare wird die Information gesucht und dynamisch zur Laufzeit in Form eines Hypermediadokuments ausgegeben. Im folgenden wird nun ein Überblick über den Aufbau von CGI-Programme gegeben.

3.1.3 CGI-Programme

CGI-Programme erhalten in der Regel Daten von einem HTML-Formular. Die eingegebenen Daten werden vom Browser in Name-Wert-Paare zerlegt und URL-codiert. Die URL-Codierung ist notwendig, um Daten konsistent im Netz zu versenden. Die codierten Daten werden in der Umgebungsvariablen QUERY_STRING

abgelegt. Das CGI-Programm greift auf die Umgebungsvariable QUERY_STRING zu und muß die Daten zur Weiterverarbeitung decodieren [Herrmann 1996].

Bsp.: HTML-Formular zur Eingabe von Vor- und Nachname:

```
<HTML>
<HEAD>
<TITLE>HTML-Formular zur Eingabe von Vor- und Nachname</TITLE>
</HEAD>
<BODY>
<H1>HTML-Formular zur Eingabe von Vor- und Nachname</H1>
<FORM Method=GET action=".../cgi-bin/formular-name.cgi
<p>Vorname
<INPUT TYPE=text SIZE=15 MAXLENGTH=15 name="vorname">
<p>Nachname
<INPUT TYPE=text SIZE=30 MAXLENGTH=30 name="nachname">
<INPUT TYPE=submit >
<INPUT TYPE=reset >
</FORM>
</BODY>
</HTML>
```

Bei Eingabe von "Annette" in die Variable *Vorname* und "Maile" in *Nachname* ist der Inhalt der Umgebungsvariablen QUERY_STRING:

vorname=Annette&nachname=Maile

Die Name-Wert-Paare werden durch "&" getrennt. Falls eine Eingabe ein Leerzeichen enthält, so wird es durch "+" codiert. Sonderzeichen wie z.B. " (" werden hexadezimal codiert. Im CGI-Programm wird dann der Inhalt von QUERY_STRING decodiert und weiterverarbeitet. Die Dekodierung wird in Kapitel 5 "Implementierung" beschrieben.

3.1.4 Fazit

Hypermedia eignet sich in der Hinsicht für die adaptive Informationsaufbereitung, daß unterschiedliche Medien für die Darstellung der verschiedenen Aufbereitungsformen verfügbar sind. Es bietet sich als Benutzungsoberfläche im betrieblichen Umfeld an. Zudem sind mit der Möglichkeit des Ausfüllens von Formularen und der Weiterverarbeitung der Formularinhalte durch CGI-Programme beispielsweise interaktive Datenbankabfragen möglich. Mit einer geeigneten Benutzermodellierung können mit Hilfe von CGI-Programmen die Informationen für die Benutzer adaptiv aufbereitet werden.
Der Einsatz von *WWW* im betrieblichen Umfeld bietet sich insofern an, daß einige *WWW*-Server sowie *WWW*-Browser kostenlos genutzt werden können.

Zusätzlich zur Darstellung von Informationen werden Programme zur Informationsbeschaffung benötigt. Daher wird im folgenden das Konzept von *intelligenten Agenten* untersucht.

3.2 Intelligente Agenten

Es existieren keine eindeutigen Definitionen über den Begriff *Agent*. In [Girard et al. 1992 S. 176] wird eine minimale vorgestellt:

> *"Ein Agent ist eine reelle oder abstrakte Einheit, die fähig ist, selbst und in ihrer Umgebung tätig zu sein, die über eine partielle Repräsentation dieser Umgebung verfügt, die in einer Multiagenten-Welt mit anderen Agenten kommunizieren kann und deren Verhalten auf den Konsequenzen ihrer Beobachtungen, ihres Wissens und den Interaktionen mit anderen Agenten beruht." (Übersetzung der Autorin)*

Bei dieser Definition liegt der Schwerpunkt in der Selbständigkeit und der Kommunikationsfähigkeit von Agenten. Die Selbständigkeit wirkt sich so aus, daß ein Agent gewisse Arbeiten erledigt, ohne daß der Benutzer dies interaktiv anfordert. Damit kann der Benutzer beispielsweise von Routinearbeiten entlastet werden.

In [Minsky et al. 1994 S.23] vertritt Doug Riecken seine Vorstellung, was ein Agent bzw. eine Agentur ist:

> *"Personally, I have a strong interest in the idea of agency - where many simpler things, call them agents if you will, come together and their combined actions enable*

> *expressive and knowledgeable behaviour - the way your Society of Mind[1] theory describes human intelligence.“*

Für Doug Riecken ist ein Agent ein einfaches Ding und nur durch die Zusammenarbeit mit anderen Agenten wird sachkundiges Verhalten und damit die Lösung komplexer Probleme ermöglicht.

Für Marvin Minsky hingegen gibt es keinen Hinweis, daß ein Agent etwas Einfaches erledigt, sondern vielmehr eine spezielle Aufgabe verrichtet. Er vertritt in [Minsky et al. 1994 S. 24] folgende Ansicht über einen Agenten:

> *“In the “Society of Mind“, the idea was to use the word “agent“ when you want to refer to a machine that accomplishes something, without your needing to know how it works. You call it an agent when you want to treat it as a black box.“*

Marvin Minsky definiert einen Agenten als eine Black Box, d.h., es ist nicht von Interesse für den Anwender, wie der Agent funktioniert, sondern daß er funktioniert.

Aus den verschiedenen Vorstellungen und Ansichten über Agenten kann folgendes entnommen werden:

- Agenten sind Computer-Programmme, die spezielle Fähigkeiten besitzen, um bestimmte Aufgaben zu erfüllen.
- Die Aufgaben eines Agenten können sowohl einfacher Art als auch komplex sein.
- Ein Agent muß die Fähigkeit haben, mit anderen Agenten kommunizieren zu können.
- Ein Agent ist eine Black Box.

Für die adaptive Informationsaufbereitung werden spezielle Agenten benötigt, um bestimmte Aufgaben zu erfüllen. Für die unterschiedlichen Anfragemöglichkeiten der Anwender wird jeweils ein Agent benötigt. Ein Agent ist beispielsweise für die Filterung von Daten aus einer Datenbank zuständig, ein anderer für das Laden eines CAD-Programms mit der entsprechenden Zeichnung etc. Bei Anfrage durch einen Benutzer sucht der dafür zuständige Agent die Information und gibt diese an den Benutzer in einer aufgrund seines Benutzerprofils angepaßten Form aus. Für den Anwender ist der Agent dabei eine Black Box.

[1] “The Society of Mind“, Marvin Minsky, Simon and Shuster, N.Y. 1985

Agenten können nach Art ihrer Aufgaben gruppiert werden. Es gibt *helfende Agenten* (agent guides), die den Benutzer bei seiner Arbeit dadurch unterstützen, indem sie das fehlende Wissen des Benutzers durch das eigene Wissen ergänzen. Beispiele für helfende Agenten sind kontextsensitive Hilfesysteme und Computer-Based-Training. *Benutzer-Agenten* (user agents) sind lernfähige, adaptierbare und adaptive Programme. Beispiele hierfür sind E-Mail-Systeme, die ankommende E-Mails nach Absendern sortieren, die der Benutzer definiert hat. Unterschiedliche Dateiformate werden durch Benutzer-Agenten erkannt und automatisch konvertiert. Die Aufgabe von *kooperativen Agenten* besteht darin, den Benutzer z.B. in Form von Zusatzprogrammen zur Erleichterung der Arbeit zu unterstützen. *Anthropomorphe Agenten* simulieren einen Menschen. Beispiele hierfür sind Computerspiele und Testsysteme zur Überprüfung von Software. *Autonome Agenten* erledigen Routineaufgaben unabhängig von einer Anforderung durch den Benutzer [Maes 1994], [Agents FAQ 1996].

Neben der Erfüllung einer Aufgabe durch einen einzelnen Agenten existiert ein wachsender Bedarf an Programmen, die zusammenarbeiten, d.h., die Informationen und Leistungen untereinander austauschen können, um so Probleme zu beheben, deren Lösung von einem Agenten alleine nicht bewerkstelligt werden kann.
In Mehragentensystemen ist zwar jeder Agent gegenüber anderen autonom, jedoch können durch Zusammenarbeit Synergieeffekte erzielt werden. Abbildung 3.3 zeigt beispielhaft den Ablauf einer Anfrage durch den Benutzer. Das System entscheidet, *Agent 2* anzusprechen. Dieser hält Rücksprache mit *Agent 3*, da er die Aufgabe alleine nicht bewerkstelligen kann. Die Ergebnisse beider Agenten ergeben dann die Ausgabe der nachgefragten Information.

Die Schwierigkeit bei der Kommunikation der Programme untereinander ist bedingt durch die Heterogenität derselben, da sie von verschiedenen Entwicklern in unterschiedlichen Sprachen geschrieben wurden. Beim agentenbasierten Software Engineering benutzen Agenten eine Agenten-Kommunikationssprache mit einer agentenunabhängigen Semantik. Es existieren prozedurale und deklarative Ansätze für eine universelle Kommunikationssprache, jedoch hat sich noch kein Standard durchgesetzt [Genesereth et al. 1994].

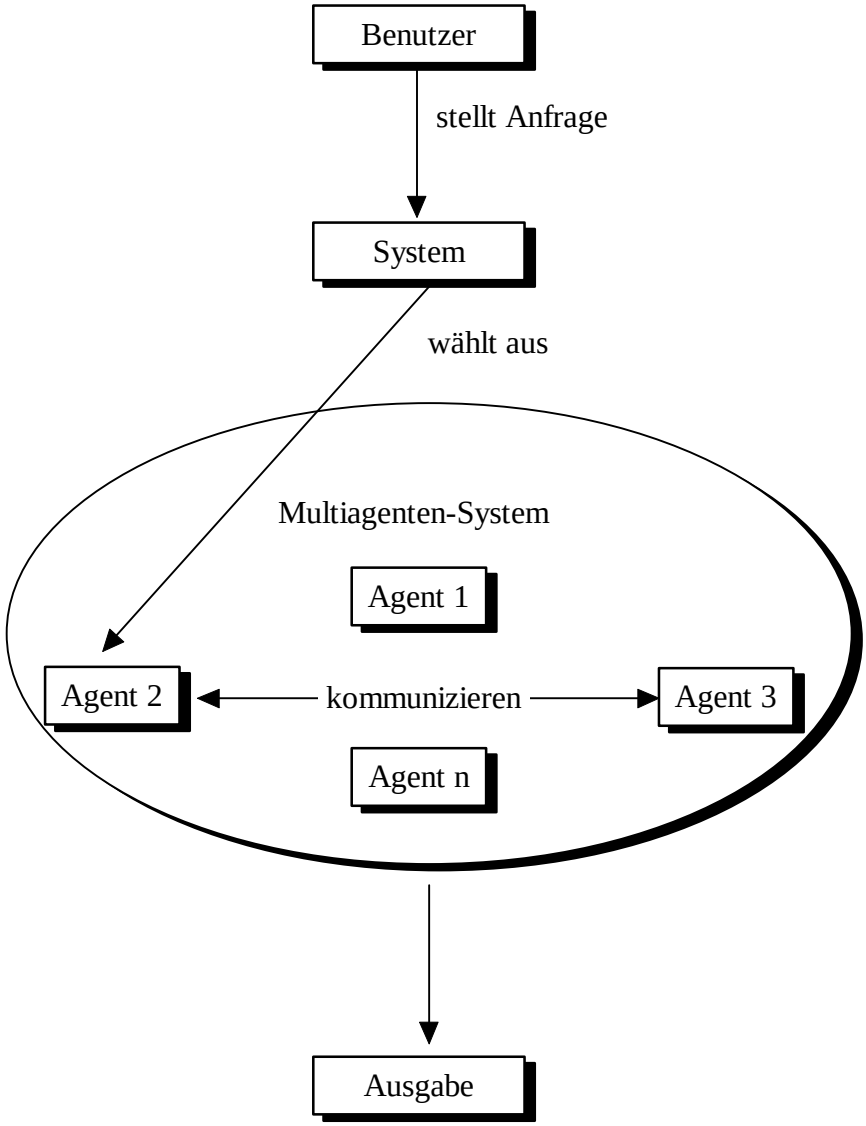


Abbildung 3.3: *Verlauf einer Anfrage in einem Multiagenten-System*

In [Sundermeyer et al. 1993] werden zwei grundlegende Arten der Kommunikation - *Shared Memory* und *Message Passing* - unterschieden. Beim *Shared Memory* werden die Informationen mittels eines gemeinsam genutzten Speichers indirekt ausgetauscht. Dies wird meist durch eine sogenannte Blackboard-Architektur realisiert, d.h., der Sender legt die Informationen an einer bestimmten Stelle ab. Der Empfänger muß diese dann von Zeit zu Zeit einholen. Der Nachteil besteht darin, daß sich Zeitverzögerungen ergeben können, da der Empfänger nicht ständig nachschaut, ob neue Informationen für

ihn vorliegen. Beim *Message Passing* dagegen wird eine Nachricht von einem Sender direkt an den Empfänger geschickt. Die Nachricht kann dabei einerseits an **einen** Agenten gesendet werden, indem seine Adresse konkret spezifiziert wird oder durch sogenanntes *Broadcasting* an **alle**. Bei der direkten Kommunikaton wirkt sich allerdings nachteilig aus, daß mit wachsender Anzahl von Agenten der Kommunikationsbedarf stark ansteigt und damit zu Überlastungen des Systems führt.

Die Agenten unterliegen bestimmten Zwängen in Bezug auf ihre Nachrichten. Diese Zwänge ergeben sich aus der Zusammensetzung einer Nachricht und dem generellem Verhalten von Agenten. So muß beispielsweise ein Agent die Wahrheit sagen. Agenten sind autonom, d.h., ein Agent darf keinen anderen Agenten dazu zwingen, eine Leistung zu erbringen, bevor dieser seine Bereitschaft anzeigt. Hat ein Agent seine Bereitschaft gezeigt, so ist er auch dazu verpflichtet, die angeforderte Leistung zu erbringen.

Agenten sind für die Realisierung adaptiver Informationsaufbereitung notwendig. Die Beschaffung und unterschiedliche Aufbereitung der gewünschten Informationen wird von verschiedenen Agenten bewerkstelligt. Bei Anforderung einer Information durch einen Benutzer entscheidet das System, welcher Agent bzw. welche Agenten für die Erzeugung der entsprechenden Ausgabe angesprochen werden müssen. Bei Einsatz von Hypermedia werden Agenten durch CGI-Programme realisiert. Bei einer Anfrage durch den Benutzer werden Parameter, die die gewünschte Information betreffen, an das CGI-Programm geschickt. Dort wird aufgrund der Art der Information der dafür zuständige Agent aufgefordert, die Information zu beschaffen und entsprechend den Anforderungen des Benutzers aufbereitet.

Im folgenden wird das Konzept von computerunterstützten Lernsystemen vorgestellt. Die Aufgabe von Lernsystemen besteht ebenso wie bei Informationssystemen darin, den Anwender von einem uninformierten Zustand in einen informierten Zustand zu führen.

3.3 Computerunterstützte Lernsysteme

Computerunterstützte Lernsysteme werden zunehmend im Bereich der Aus- und Weiterbildung eingesetzt. Im Gegensatz zu klassischen Fortbildungen, wie Seminaren etc. sind sie weder an einen Raum noch an eine bestimmte Zeit gebunden, die Kosten sind geringer und die Informationsbeschaffung kann individuell erfolgen [Roller et al. 1995].

Das Lernsystem soll in der Lage sein, sowohl Anfänger, Fortgeschrittene als auch Experten zu unterstützen. Dabei sind die Anforderungen dieser verschiedenen Gruppen sehr unterschiedlich. Ein Anfänger muß durch das System geführt werden, während ein Experte die Freiheit haben muß, Themengebiete selbst auszuwählen.

In [Meyerhoff 1994] werden unterschiedliche Arten von Lernsystemen vorgestellt:

Tutorielle Lernsysteme

Tutorielle Lernsysteme präsentieren dem Lernenden (auch Lerner genannt) Informationen und stellen Fragen dazu. Basierend auf den Antworten verzweigt das Lernsystem zu weiteren Informationen oder wiederholt den Lernabschnitt. Im System sind interaktive Elemente enthalten.

Drill

Drill ist ein sehr starres System. Es stellt Fragen und aufgrund der Antworten entscheidet das System, ob weitere Fragen gestellt werden. Der Anwender hat dabei keinen Einfluß auf den Ablauf. Drill eignet sich hauptsächlich zum Festigen von Wissen, jedoch weniger zum Erlernen von Neuem.

Simulationsprogramme

Der Lerner kann Parameter eines Simulationsprogrammes interaktiv verändern und so die verschiedenen Auswirkungen beobachten.

Problemlösungsumgebungen

Dem Lerner wird eine Aufgabe gestellt, die er mit den zur Verfügung gestellten Mitteln lösen muß, d.h., er muß den Lösungsweg selbst entwickeln.

Lernspiele

Lernspiele können unterschiedlich aufgebaut sein, d.h., es existieren oft Ähnlichkeiten zu Simulationsprogrammen oder Problemlösungsumgebungen. Der entscheidende Unterschied liegt darin, daß Programmabläufe und Darstellung in Form eines Spiels entwickelt wurden, um den Spieltrieb zum Lernen zu nutzen.

Intelligente tutorielle Systeme

Bei intelligenten tutoriellen Lernsystemen werden Informationen im Mensch-Maschine-Dialog übermittelt. Mittels Techniken aus dem Bereich der künstlichen Intelligenz werden Annahmen über den Wissensstand des Benutzers gemacht. Aufgrund dieser Annahmen entscheidet das System, welche Informationen in welcher Form dem Lerner übermittelt werden.

Die vorgestellten Lernsysteme werden in systemgesteuert und lernergesteuert unterteilt. Systemgesteuerte Systeme entscheiden, welche Informationen wie dargestellt werden, ohne daß der Lerner Einfluß darauf hat. Zu den systemgesteuerten Systemen gehören *tutorielle Lernsysteme* und *Drill*.
Der Ablauf von lernergesteuerten Systemen wird vom Anwender bestimmt. Zu diesen Systemen zählen *Simulationsprogramme, Problemlösungsumgebungen, Lernspiele und intelligente tutorielle Lernsysteme*. Diese Lernsysteme eignen sich eher für Fortgeschrittene, da diese über das Wissen verfügen, wie sie gezielt bestimmte Informationen oder Interaktionsformen auswählen können.

Die Parallelen von Lernsystemen zu Informationssystemen bestehen darin, daß sie den Anwender informieren. Die in manchen lernergesteuerten Lernsystemen realisierte benutzergerechte Aufbereitung der Informationen muß auf betriebliche Informationssysteme projiziert werden, da unterschiedliche Verständnisniveaus der Mitarbeiter und Detaillierungsgrade der Informationen in einem Betrieb verschiedenartige Ausgabearten der Informationen erfordern. Bei den Lernsystemen ist Grundlage für die Informationsaufbereitung das Wissen des Benutzers, wobei im betrieblichen Umfeld Funktion und Aufgabenbereich der Anwender im Mittelpunkt stehen. Ein gravierender Unterschied zwischen Lernsystemen und betrieblichen Informationssystemen liegt in der Aktualität der Informationen. Während bei Lernsystemen die Informationen über einen gewissen Zeitraum hinweg gleichbleibend sein können, ändern sich die Informationen im betrieblichen Informationssystem ständig, d.h., die Informationsaufbereitungen können bei Lernsystemen statisch erfolgen, während sie im betrieblichen Umfeld hauptsächlich dynamisch zur Laufzeit erzeugt werden müssen.

3.4 Zusammenfassung

In diesen Kapitel wurden Konzepte neuer Informationssysteme, wie *Hypermedi*a, *intelligente Agenten* und *computerunterstützte Lernsysteme* auf Verwendbarkeit für die adaptive Informationsaufbereitung untersucht.

Die Vorteile von *Hypermedia* für die adaptive Informationsaufbereitung liegen in der Möglichkeit der Integration von Medien wie Audio und Video in Hypertextdokumente. Somit können unterschiedliche Aufbereitungsarten realisiert werden, d.h., die Informationsaufbereitungen sind nicht auf Text und Graphik beschränkt, sondern es können z.B. auch Videos zum besseren Verständnis gezeigt werden. Mit Hilfe von CGI-Programmen können Informationen adaptiv für den Benutzer aufbereitet werden. Die mittels eines HTML-Formulars übergebenen Parameter an ein CGI-Programm sind die Grundlage für eine benutzerangepaßte Ausgabe der Informationen.

Das Konzept der intelligenten Agenten eignet sich insofern, als daß verschiedene Agenten die Aufgaben der unterschiedlichen Aufbereitungsformen übernehmen können, wobei der Anwender nicht wissen muß, wie diese Agenten funktionieren. Es besteht allerdings die Schwierigkeit der Realisierung einer geeigneten Kommunikation der Agenten untereinander. Eine weitere Aufgabe von intelligenten Agenten bei der adaptiven Informationsaufbereitung besteht darin, die verschiedenen Informationsaufbereitungsarten den entsprechenden Anwendern zuzuordnen.

Die Parallelen von Lernsystemen zu betrieblichen Informationssystemen bestehen in derselben Zielsetzung, nämlich den Anwender zu informieren. Bei beiden Systemen besteht die Notwendigkeit einer geeigneten Benutzermodellierung, um eine benutzergerechte Darstellung von Informationen zu ermöglichen. Im Gegensatz zu den Lernsystemen steht bei der adaptiven Informationsaufbereitung im betrieblichen Umfeld die Aufgabe des Benutzers und nicht primär sein Wissen im Mittelpunkt. Ein entscheidender Unterschied liegt in der Aktualität der Daten. In betrieblichen Informationssystemen ändern sich Daten ständig, so daß eine dynamische Informationsaufbereitung notwendig ist.

Im folgenden Kapitel wird ein Konzept eines Architekturmodells für die adaptive Informationsaufbereitung entwickelt. Dabei besteht die Anforderung an das Architekturmodell, daß es offen ist, d.h., es soll jederzeit um Dateiformate und zusätzliche Filter erweiterbar sein. Die Basis für das Konzept stellen die in Kapitel 2

herausgearbeiteten Informationsaufbereitungsarten im betrieblichen Umfeld sowie die in diesem Kapitel untersuchten Konzepte neuer Informationssysteme dar.

4 Architekturmodell für die adaptive Informationsaufbereitung

In diesem Kapitel wird ein Konzept für ein offenes, erweiterbares Architekturmodell für die adaptive Informationsaufbereitung entwickelt. Die Aufgabe der adaptiven Informationsaufbereitung besteht darin, aus dem betrieblichen Informationssystem die für den Benutzer relevanten Daten herauszufiltern und in geeigneter Form aufzubereiten. Für die Bereitstellung und Aufbereitung der Informationen werden intelligente Agenten benötigt. Welche Aufgaben die Agenten zu erfüllen haben, ergibt sich aus einer Analyse der Informationsflüsse innerhalb eines Betriebes. Das Ergebnis der Analyse soll zeigen, welche Informationen in welcher Form benötigt werden. In Kapitel 2 - Betriebliches Umfeld - wurden beispielhaft einige Funktionsbereiche daraufhin untersucht. Eine weitere Komponente zur Realisierung der adaptiven Informationsaufbereitung ist die Beschreibung der Benutzer. In einem Benutzermodell müssen über die Anwender Informationen abgelegt sein, deren Auswertung es ermöglicht, die Informationsbedürfnisse des Benutzers zu bestimmen. Um die entsprechenden Informationsaufbereitungen den einzelnen Benutzern zuordnen zu können, werden wiederum intelligente Agenten benötigt. Diese fungieren als Schnittstelle zwischen den Benutzern und den Agenten zur Informationsbeschaffung und -aufbereitung. Sie erhalten bei einer Anfrage durch den Benutzer Parameter zur Bereitstellung der angeforderten Information. Aufgrund der Einträge im Benutzermodell wird dann ein spezieller Agent aufgefordert, die Information zu beschaffen und entsprechend den Bedürfnissen des Benutzers aufzubereiten.

Aufgrund der oben genannten Anforderungen unterteilt sich das Architekturmodell für die adaptive Informationsaufbereitung im wesentlichen in drei Teile:

1. **Benutzermodell**
2. **Agenten**
3. **Schnittstelle zwischen Benutzermodell und Agenten**

Im Benutzermodell werden Benutzer bzw. Benutzergruppen nach ihren Aufgabenbereichen, Wissen und Präferenzen klassifiziert. Die Agenten sind für die Bereitstellung von Programmen und Daten aus dem betrieblichen Informationssystem zuständig. Die Aufgabe der Schnittstelle besteht darin, auf der Grundlage der individuellen Einträge im Benutzermodell die entsprechenden Agenten zur Ausführung der Informationsbeschaffung anzusprechen und die Informationen an den Benutzer weiterzuleiten.

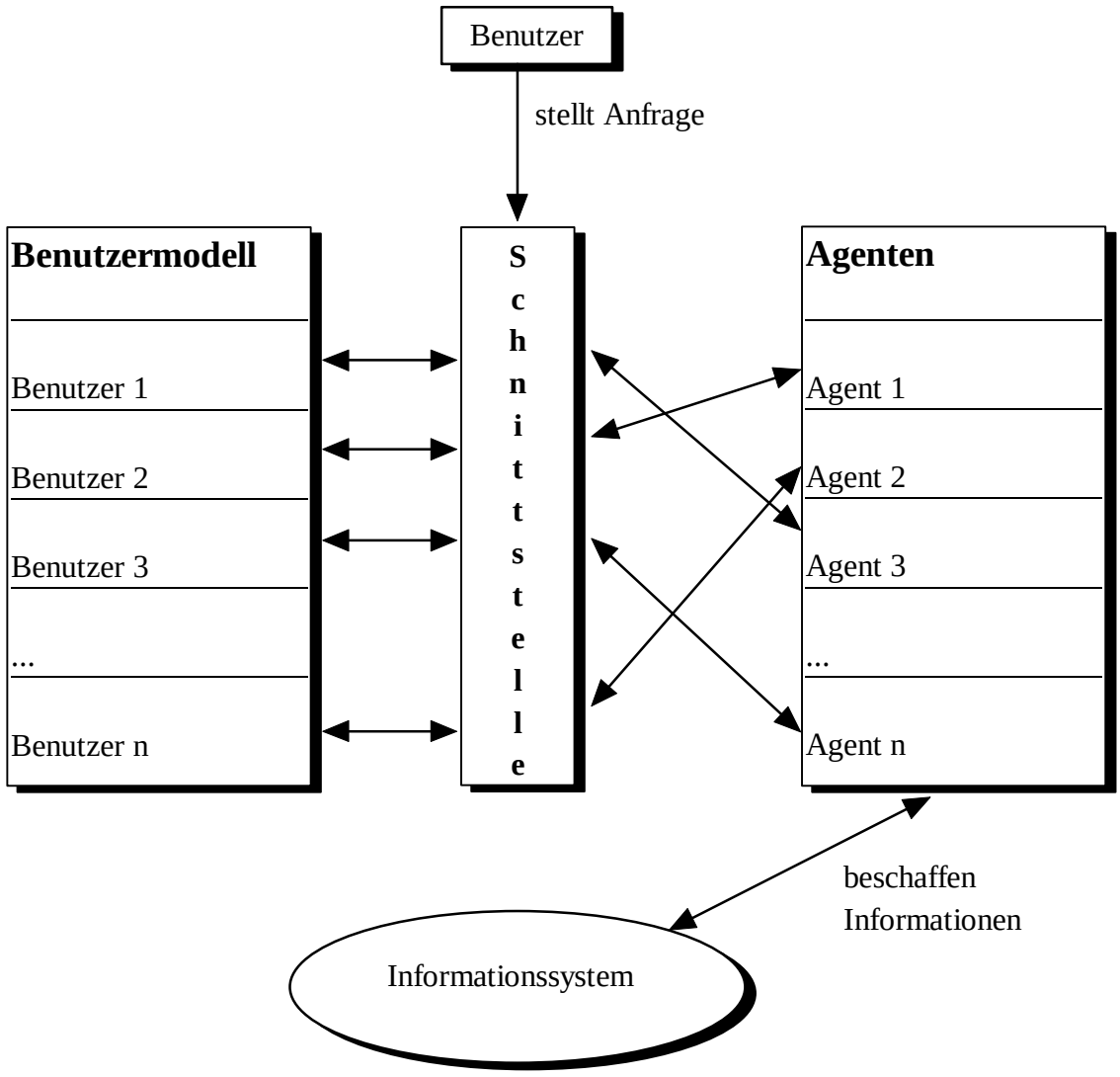


Abbildung 4.1: *Architekturmodell für die adaptive Informationsaufbereitung*

Abbildung 4.1 zeigt die Zusammenhänge der drei Komponenten des Architekturmodells. Stellt beispielsweise *Benutzer 1* eine Anfrage, entscheidet die Schnittstelle aufgrund der individuellen Einträge im Benutzermodell, *Agent 3* aufzufordern, die gewünschten Informationen zu beschaffen. Die Informationen werden dann an *Benutzer 1* in geeigneter Form ausgegeben.

Im folgenden werden die theoretischen Grundlagen für die Konzeption der drei Komponenten des Architekturmodells aufgezeigt und anhand der Beispielfirma *Baukado* konkretisiert.

4.1 Benutzermodell

Heutzutage besitzen moderne Computersysteme einen hohen Grad an Funktionalität. Die einzelnen Benutzer bzw. Benutzergruppen benötigen jedoch nur Teile der zur Verfügung stehenden Funktionen. Aus diesem Grund besteht die Notwendigkeit, Anwendungssysteme an den Benutzer und seine Aufgaben anzupassen.
In einem Benutzermodell werden die Benutzer bzw. Benutzergruppen beschrieben. Die Einträge im Modell sind die Grundlage für die unterschiedlichen Informationsaufbereitungen. In diesem Kapitel werden Inhalt und Aufbau eines Benutzermodells, sowie die Anpassungsmöglichkeiten an den Benutzer beschrieben. Abschließend wird das Benutzermodell für die Firma *Baukado* entworfen.

4.1.1 Anpassung von Anwendungssystemen an den Benutzer und seine Aufgabe

Jeder Benutzer besitzt unterschiedliche Erfahrungen, Kenntnisse und Fähigkeiten in Bezug auf Computer, dem benutzten System und seinem Aufgabenbereich [Schwab 1989], d.h., es existiert kein “Einheitsbenutzer“. Ferner hat jeder Anwender Präferenzen in Bezug auf Interaktionstechniken und Eingabegeräte. Aus diesen Prämissen resultiert die Notwendigkeit, Anwendungssysteme an die individuellen Bedürfnisse der Benutzer anzupassen.

Die Anpassung an den Kenntnisstand des Benutzers kann durch eine bewußt eingeschränkte Funktionalität erfolgen. Dadurch wird eine Überforderung des Nutzers umgangen. Beim Erlernen eines neuen Systems werden dem Anfänger zunächst lediglich

die grundlegenden Befehle zur Verfügung gestellt. Mit wachsendem Wissen hat er dann die Mö glichkeit, mehr Funktionen zu nutzen.

Durch verkürzte Produktlebenszyklen und sich ständig ändernden Marktsituationen müssen Informationen schnell verfügbar sein. Daher müssen dem Benutzer Funktionen zur Verfügung gestellt werden, die für ihn relevante Daten in kurzer Zeit aus dem Informationssystem selektieren. Der Einsatz von Filtern zur Einschränkung der Informationsflut auf den benötigten Teil vermeidet eine Überforderung des Benutzers und erhöht damit die Akzeptanz des Systems. Zudem sind Zugriffsberechtigungen zu erteilen, da bestimmte Informationen nur von berechtigten Personen eingesehen werden dürfen.

Der Benutzer soll die Mö glichkeit haben, seinen Computerarbeitsplatz so zu gestalten, wie es für ihn den größtmö glichen Nutzen ergibt. Dazu gehören z.B.:

- Interaktionstechniken

  Befehle sollten über Menüs, Icons, Kommandoeingabe etc. verfügbar sein. Beim CAD-Programm *AutoCADâ LT* z.B. hat der Benutzer die Mö glichkeit, seinen eigenen *Werkzeugkasten* (siehe Abbildung 4.2) mit Icons zu erstellen. Er kann aber jederzeit weniger häufig benutzte Befehle über Menüs abrufen. Zusätzlich kann jeder Befehl auf Kommandoebene eingegeben werden.

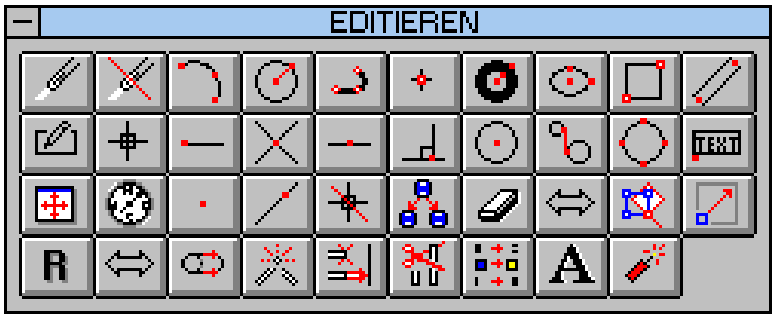


Abbildung 4.2: *Individueller Werkzeugkasten von AutoCAD® LT*

- Tastenbelegung und Namensgebung

  Mit der Definition von Tastenkombinationen werden individuelle Arbeitsweisen unterstützt. Im Textverarbeitungsprogramm *Microsoft Word 6.0* kann der Benutzer z.B. zur Ausführung von Makros oder Formatierung von Texten eigene Tastenbelegungen wählen. Die Änderung der Namen von Befehlen gibt dem Benutzer die Gelegenheit, ihm vertraute Begrifflichkeiten zu benutzen.

- Bildschirmlayout

  Zum Layout des Bildschirms gehören Hintergrund- und Vordergrundfarben sowie Schriftgrößen.

Bei der Anpassung an die Aufgabe sind Detaillierungsgrade zu unterscheiden. So sollten z.B. bei der Bearbeitung von Graphiken die Möglichkeiten bestehen, diese sowohl in einer Gesamtansicht auf einer Seite darzustellen, als auch bei genauen Arbeiten mittels eines Zooms einen Ausschnitt auf Pixelebene zu bearbeiten.

Um Anwendungssysteme an den Benutzer und an seine Aufgabe anpassen zu können, besteht die Notwendigkeit, die Informationsbedürfnisse und Präferenzen des Anwenders zu modellieren. Im folgenden werden Aufbau und Inhalt eines Benutzermodells beschrieben..

4.1.2 Inhalt eines Benutzermodells

In einem Benutzermodell werden Informationen über den Benutzer gespeichert und verwaltet. Dabei hängt die Art der Informationen stark vom Verwendungszweck des Modells ab. Bei Lernsystemen z.B. muß der Wissensstand des Benutzers modelliert werden. Auch bei Auskunftssystemen ist das Wissen des Benutzers Bestandteil des Benutzermodells, um relevante Daten herauszufiltern. In [Schwab 1989] wird *Grundy*, ein Bibliothekssystem vorgestellt, das dem Leser Romane vorschlägt. Im Benutzermodell von *Grundy* werden die Vorlieben und Abneigungen für verschiedene Themengebiete festgehalten. Die Bewertung erfolgt in Form einer Zahl von -5 (kein Interesse) bis 5 (starkes Interesse). Das Bibliothekssystem sucht dann aus einem Informationssystem, das sämtliche Bücher mit Gewichtung der einzelnen Themen enthält, jene heraus, auf die die Einträge im Benutzermodell zutreffen.

In Betrieben hängt das Benutzermodell stark von der Art des Unternehmens ab. Im folgenden werden einige Attribute vorgestellt, die Grundlage für ein Benutzermodell im betrieblichen Umfeld sind.

- **Benutzeridentität**

  Damit das System erkennt, auf welche Einträge im Benutzermodell zuzugreifen ist, muß sich der Benutzer eindeutig identifizieren. Diese Identifikation muß während einer Sitzung verfügbar sein. Beispiele hierfür sind die User-ID oder der Login.

- **Funktionsgruppenzugehörigkeit**

  Ein wichtiger Aspekt ist die Funktionsgruppenzugehörigkeit, um auf den Aufgabenbereich schließen zu können. Auf dieser Grundlage werden dem Anwender nur die Programme, Daten und Darstellungsformen zur Verfügung gestellt, die er für seine Arbeit benötigt.

- **Qualifikation**

  Das Wissen des Benutzers wird durch ein Qualifikationsattribut modelliert, in dessen Ausprägung die Qualifikation des Mitarbeiters mit *Auszubildender, ausreichend, gut, überdurchschnittlich* oder *sehr gut* beschrieben wird. Mitarbeiter mit einer höheren Ausbildung erhalten bei Beginn der Betriebszugehörigkeit ein besseres Qualifikationsattribut, jedoch kann auch durch Fortbildung eine höhere Qualifikation erlangt werden. Die Ausprägung des Qualifikationsattributs kann sich insofern auf die bereitgestellte Information auswirken, daß ein Auszubildender wichtige Daten wie z.B. eine Konstruktionszeichnung im Nur-Lese-Modus erhält und sie somit nicht verändern kann.

- **Kenntnisstand**

  Der Kenntnisstand des Benutzers wird durch *Anfänger, Fortgeschrittener* und *Experte* widergespiegelt. Dabei werden beispielsweise einem *Anfänger* nur grundlegende Befehle eines Anwendungssystems zur Verfügung gestellt. Mit wachsendem Kenntnisstand werden dem Benutzer entsprechend mehr Funktionen des Programms bereitgestellt.

- **Präferenzen**

  Die Präferenzen der Benutzer zur Gestaltung der Arbeitsoberfläche, d.h., Wahl von Hintergrund-, Vordergrundfarben und Interaktionstechniken, sollen ebenso Bestandteil des Benutzermodells sein. Eine individuell angepaßte Arbeitsumgebung erhöht die Akzeptanz des Systems und somit die Produktivität.

Die aufgeführten Attribute stellen die Grundlage für ein Benutzermodell im betrieblichen Umfeld dar. Je nach Art des Betriebes muß darauf aufbauend das Modell um spezielle Attribute erweitert werden. Wenn beispielsweise ein Unternehmen internationale Filialen besitzt, so kann die gewünschte Sprache im Benutzermodell abgelegt sein.
Im folgenden wird die Architektur eines Benutzermodells aufgezeigt.

4.1.3 Aufbau eines Benutzermodells

Der Systemdesigner legt fest, welche Attribute mit den zugehörigen Wertzuweisungen im Modell enthalten sein sollen. Dafür muß er sich einen genauen Überblick über das gesamte betriebliche Umfeld verschaffen. Das Benutzermodell soll so gestaltet sein, daß es jederzeit um Attribute oder neue Werte erweitert werden kann, um sich betrieblichen Veränderungen anpassen zu können. Zusätzlich definiert der Systemdesigner Repräsentation, zeitliche Auswirkung bei Veränderungen, Gültigkeit und Ausprägungen des Modells sowie Art und Weise der Eingabe von Benutzerdaten.

Repräsentation

Die Repräsentation des Benutzermodells soll in expliziter Form erfolgen, d.h., es existiert eine Datenstruktur für das Benutzermodell. Dadurch kann das Modell entweder vom Benutzer oder vom System aktualisiert werden, um es an die jeweiligen Bedürfnisse der Benutzer anzupassen. Eine implizite Repräsentation hat zur Folge, daß das Modell nicht verändert werden kann und für alle Benutzer gleich ist.

Zeitliche Auswirkung

In Bezug auf die zeitliche Auswirkung bei Änderung der Attributwerte des Modells existieren zwei Ansätze. Bei einem statischen Modell sind alle Werte während einer Sitzung fest. Änderungen werden erst bei der nächsten Sitzung wirksam. Beispiele hierfür sind die "*Profiles*" unter *Unix*. Bei einem dynamischen Modell hingegen wird die Änderung sofort wirksam. Bei der Notwendigkeit von häufigen Änderungen der

Attributwerte ist ein dynamisches Benutzermodell vorzuziehen. Im betrieblichen Umfeld ist ein statisches Modell ausreichend, da sich das Benutzermodell nicht all zu oft ändern wird, weil ein Mitarbeiter weder ständig seinen Aufgabenbereich wechselt, noch sich seine Qualifikation fortwährend ändert.

Gültigkeit

Die Gültigkeit eines Modells kann entweder kurz- oder langfristig sein. Ein kurzfristiges Modell gilt nur für eine Sitzung und muß somit jedesmal neu erstellt werden. Bei einem langfristigen Benutzermodell werden die Informationen abgespeichert und gelten bei jeder folgenden Sitzung. Letzteres Modell ist in betrieblichen Systemen vorzuziehen, da eine ständige Erstellung eines Modells vom Benutzer als störend empfunden wird.

Ausprägungen

Die Ausprägungen der Attribute des Modells können sehr zahlreich sein, da eventuell die Notwendigkeit besteht, daß jeder Benutzer ein eigenes Profil erhält. Ob jedoch einige Benutzer mit ähnlichen Bedürfnissen zu Benutzergruppen zusammengefaßt werden können, ist von der Art des Unternehmens abhängig.

Eingabe von Benutzerdaten

Das Füllen des Modells mit Benutzerdaten kann auf mehrere Arten erfolgen. Beispielsweise muß der Benutzer bei der ersten Sitzung in einem Eingangsinterview die verschiedenen Attribute mit den zur Verfügung gestellten Werten im Modell eintragen. Eine weitere Möglichkeit besteht darin, aus einem bestehenden Benutzermodell eines anderen Systems die relevanten Daten zu übernehmen. Dann müssen lediglich die speziellen Attribute mit Werten gefüllt werden. Der dritte Weg ist eine Default-Belegung durch das System. Der Benutzer ändert daraufhin nach und nach die Werte. Die Default-Belegung eignet sich nur bedingt für Unternehmen, da die Aufgabenbereiche der Mitarbeiter sehr unterschiedlich sein können. Denkbar ist eine Default-Belegung aufgrund der Funktionsgruppenzugehörigkeit. Für die Benutzermodellierung im betrieblichen Umfeld eignen sich hauptsächlich die erste bzw. die zweite Möglichkeit, falls ein ähnliches Benutzermodell existiert.

Der Systemdesigner legt fest, in welcher Form die Daten des Benutzermodells abgespeichert werden. Sie können beispielsweise in einer Textdatei, einer Tabelle oder in einer Datenbank abgelegt werden. Da Zugriff und Verwaltung der Daten in einer Datenbank am effizientesten sind, eignet sich diese Möglichkeit am besten.

4.1.4 Anpassung des Modells an den Benutzer

Das Benutzermodell muß von Zeit zu Zeit aufgrund sich ändernder Aufgabenbereiche, Qualifikationen etc. an den Benutzer angepaßt werden. Prinzipiell können zwei Arten der Anpassung unterschieden werden:

Systeminitiierte Anpassung

Wird das Benutzermodell ausschließlich vom System verändert, fühlt sich der Benutzer eventuell beobachtet, was sich nicht positiv auf seine Arbeit auswirkt. Außerdem können Änderungen durch das System vom Benutzer nicht gewollt sein und führen zu Verwirrung über den Grund dieser Veränderungen.

Benutzerinitiierte Anpassung

Benutzerinitiierte Anpassung bedeutet, daß der Benutzer Änderungen im Modell vornimmt. Die alleinige Anpassung durch den Benutzer ist aber nicht optimal, da der Nutzer oftmals nicht das Ausmaß der Anpassungsfähigkeit des Systems kennt.

Beide Ansätze alleine sind für die Anpassung nicht geeignet, da die systeminitiierte Anpassung zu starr und unflexibel ist und die benutzerinitiierte zu große Gefahren in Bezug auf die Datensicherheit birgt. Daher sollen einige Attribute vom System und einige vom Benutzer verändert werden können. Zudem muß die Mö glichkeit in Betracht gezogen werden, daß beispielsweise Änderungen von Attributen vom Vorgesetzten oder einem Administrator vorgenommen werden, da einige Attribute vom Benutzer nicht verändert werden dürfen. Dies ist beispielsweise bei den Einträgen, die Qualifikation oder Funktionsgruppenzugehörigkeit betreffen, der Fall. Dadurch hätte er die Mö glichkeit, auf Daten zuzugreifen und zu ändern, für die er keine Berechtigung hat.
Der Benutzer soll aber Zugriff auf die Attribute haben, die das Bildschirmlayout und die Interaktionstechniken betreffen. Ebenso soll die Änderungsmö glichkeit der Attribute des Kenntnisstandes beim Benutzer liegen, damit er weder über- noch unterfordert wird, jedoch kann ihn das System aufgrund seiner Handlungsweise aufmerksam machen, daß eine Änderung der Attribute des Kenntnisstandes erfolgen soll. Anpassungen in den Bereichen Funktionsgruppenzugehörigkeit, Qualifikation und Berechtigungen müssen durch einen Administrator oder den Vorgesetzten erfolgen, da diese Attribute maßgeblich für die Informationsaufbereitung und -bereitstellung sind.

4.1.5 Das Benutzermodell für *Baukado*

Auf der Grundlage des beschriebenen Konzeptes zum Modellaufbau soll nun für die Firma *Baukado* beispielhaft anhand einiger Funktionsgruppen konkret ein Benutzermodell erstellt werden.

Das Benutzermodell enthält folgende Attribute mit zugehörigen möglichen Ausprägungen:

Benutzeridentität

- Login der Benutzer

Funktionsgruppenzugehörigkeit

- Konstruktion
- Arbeitsplanung
- Bestellwesen
- Kostenträgerrechnung

Qualifikation

- Auszubildender
- ausreichend
- gut
- überdurchschnittlich
- sehr gut

Kenntnisstand

- Anfänger
- Fortgeschrittener
- Experte

Bildschirmhintergrundfarbe

- Farbe aus einer vorgegebenen Farbtabelle

Bildschirmvordergrundfarbe

- Farbe aus einer vorgegebenen Farbtabelle

Schriftgröße

- Auswahl einer Schriftgröße aus vorgegebenen Möglichkeiten von 8 pt bis 30 pt

Interaktionstechnik

- Menü
- Icon
- Kommandoebene

Im weiteren Verlauf werden zur Vereinfachung der Darstellung des Benutzermodells die Attribute, die die Präferenzen des Benutzers betreffen, nicht aufgeführt.
Es kann die Notwendigkeit bestehen, daß jedem Zeileneintrag eine unterschiedliche Informationsaufbereitung zugewiesen werden muß. Dies bedeutet, daß in folgendem Beispiel bei nur drei Attributen insgesamt 60 unterschiedliche Möglichkeiten bestehen, da für die *Funktionsgruppe* vier, für die *Qualifikation* fünf und für den *Kenntnisstand* drei verschiedene Einträge möglich sind.

Benutzermodell			
Benutzer	**Funktionsgruppe**	**Qualifikation**	**Kenntnisstand**
Bach	Konstruktion	sehr gut	Experte
Franke	Konstruktion	Auszubildender	Anfänger
Huber	Arbeitsplanung	gut	Fortgeschrittener
Vogel	Bestellwesen	überdurchschnittlich	Experte
Weber	Kostenträgerrechnung	gut	Experte

Abbildung 4.3: *Benutzermodell ohne Präferenzen der Benutzer*

In Abbildung 4.3 werden Eintragungen des Benutzermodells für *Baukado* ohne Präferenzen der Benutzer aufgeführt.

Der Systemdesigner legt die Zuweisung der entsprechenden Informationsaufbereitung für die einzelnen Benutzerprofile fest. Die Zuweisungen für obiges Beispiel lauten.:

Der *Experte Bach* aus der Abteilung *Konstruktion* mit der Qualifikation *sehr gut* erhält auf Anfrage veränderbare Konstruktionszeichnungen oder Stücklisten.

Mitarbeiter *Franke* aus derselben Abteilung mit der Qualifikation *Auszubildender* und dem Status *Anfänger* erhält ebenso Konstruktionszeichnungen oder Stücklisten, jedoch werden diese durch seine Änderungen nicht aktualisiert. Eine weitere mögliche Zuweisung für Mitarbeiter *Franke* wäre eine ausschließliche Bereitstellung von Test-Konstruktionszeichnungen bzw. Test-Stücklisten. So kann er die Funktionalität der Programme erlernen, wichtige Daten sind jedoch abgesichert.

Der Arbeitsplaner *Huber* mit der Qualifikation *gut* und dem Kenntnisstand *Fortgeschrittener* hat die Berechtigung, Arbeitspläne zu verändern. Die zur Aktualisierung des Arbeitsplans benötigte Stückliste erhält er jedoch im Nur-Lese-Modus, damit er diese nicht verändern kann.

Mitarbeiter *Vogel* aus der Abteilung *Bestellwesen* mit der Qualifikation *überdurchschnittlich* und dem Status *Experte* erhält zur Berechnung der *Bestellmenge* die Werte der *Bedarfsmenge*, *Bestand* und *Minimalbestand* aus dem betrieblichen Informationssystem gefiltert.

Der Benutzer *Weber* aus der Funktionsgruppe *Kostenträgerrechnung* mit der Qualifikation *gut* und dem Status *Experte* erhält zur Berechnung der Materialeinzelkosten die *Menge* der benötigten Materialien aus den Teilestammdaten und den monetären *Wert* des Materials aus den Materialstammdaten in Form einer Tabelle.

Die folgende Abbildung 4.4 zeigt die Zuweisungen der Informationsaufbereitungsarten zu den entsprechenden Zeileneinträgen im Benutzermodell. Die Informationsaufbereitungen variieren einerseits aufgrund der Funktionsgruppen-zugehörigkeit. So erhält beispielsweise ein Mitarbeiter der Funktionsgruppe *Konstruktion* eine Stückliste, die er verändern kann und ein Anwender aus der Abteilung *Arbeitsplanung* eine Stückliste, die nur lesbar für ihn ist. Innerhalb einer Abteilung variieren die Möglichkeiten der Informationsaufbereitung aufgrund unterschiedlicher

Qualifikation und Kenntnisstand. Beispielsweise erhält der Mitarbeiter aus der *Konstruktion* mit Qualifikation *sehr gut* und Kenntnisstand *Experte* eine veränderbare Konstruktionszeichnung bzw. Stückliste. Ein Mitarbeiter derselben Abteilung mit Qualifikation *Auszubildender* und Kenntnisstand *Anfänger* erhält allerdings lediglich Test-Zeichnungen bzw. Test-Stücklisten, um die Funktionalität der Programme zu erlernen.

Zuweisung von Informationsaufbereitungen			
Funktionsgruppe	**Qualifikation**	**Kenntnisstand**	**Informationsaufbereitung**
Konstruktion	sehr gut	Experte	Konstruktionszeichnung oder Stückliste veränderbar
Konstruktion	Auszubil-dender	Anfänger	Konstruktionszeichnung oder Stückliste veränderbar ohne Speicherungsmöglichkeit bzw. Test-Konstruktionszeichnung oder Test-Stückliste
Arbeitsplanung	gut	Fortgeschrittener	Arbeitsplan veränderbar oder Stückliste im Nur-Lese-Modus
Bestellwesen	überdurch-schnittlich	Experte	*Bedarfsmenge*, *Bestand* und *Minimalbestand* gefiltert aus dem Informationssystem
Kostenträger-rechnung	gut	Experte	*Menge* aus den Teilestammdaten *Wert* aus den Materialstammdaten gefiltert aus dem Informationssystem

Abbildung 4.4: *Zuweisungen von Informationsaufbereitungen resultierend aus den Zeileneinträgen im Benutzermodell*

Diese Zuweisungen sind in der Schnittstelle zwischen Benutzermodell und Agenten abgespeichert und dienen als Basis für die Ausführung der Aufbereitungen. Zunächst werden jedoch die Agenten beschrieben, die für die verschiedenen Informationsaufbereitungen benötigt werden.

4.2 Agenten

Die Aufgabe der Agenten besteht darin, den Anwendern auf Anfrage die benötigten Programme und Daten zu liefern. Dabei hat jeder Agent eine ganz spezielle Funktion. Welche Art von Agenten benötigt werden, hängt wiederum von der Art des Betriebes ab. Im folgenden werden beispielhaft einige Aufgaben von Agenten vorgestellt, die für den Ablauf eines bereichsübergreifenden, innerbetrieblichen Informationsflusses benötigt werden.

- Aufruf von Textverarbeitungsprogrammen, Tabellenkalkulationen, CAD-Programmen etc., um angeforderte Dateien zu ändern oder neue Files zu erstellen. Dabei müssen Filter und Zugriffsrechte für bestimmte Dateien oder Pfade festgesetzt werden.

- Filterung von Attributen aus einem Datenbanksystem, wobei der Anwender nur für ihn bestimmte Attribute ändern darf. Andere für seine Arbeit benötigte Attribute erhält er im Nur-Lese-Modus, da er nicht berechtigt ist, diese zu verändern.

Für verschiedene Aufgaben werden unterschiedliche Agenten benötigt, d.h., es existiert ein Agent, der ein Textverarbeitungsprogramm, ein anderer, der eine Tabellenkalkulation und wiederum ein anderer, der ein CAD-Programm etc. mit den entsprechenden Dateien lädt und Zugriffsberechtigungen setzt (siehe Abbildung 4.5).

Für die Filterung von Attributen aus einer Datenbank sind mehrere Agenten notwendig. Ein Agent bereitet beispielsweise die Attribute derart auf, daß sie verändert werden können und ein anderer beschafft lediglich die Inhalte der Attribute und gibt diese auf dem Ausgabegerät aus. Bei Aufbereitung in der Form, daß sowohl veränderbare als auch Attribute im Nur-Lese-Modus bereitgestellt werden, fordert ein weiterer Agent die oben genannten beiden Agenten zur Beschaffung der Informationen aus der Datenbank auf, diese Aufgabe zu erledigen und führt die Ausgabe der beiden Agenten zusammen.

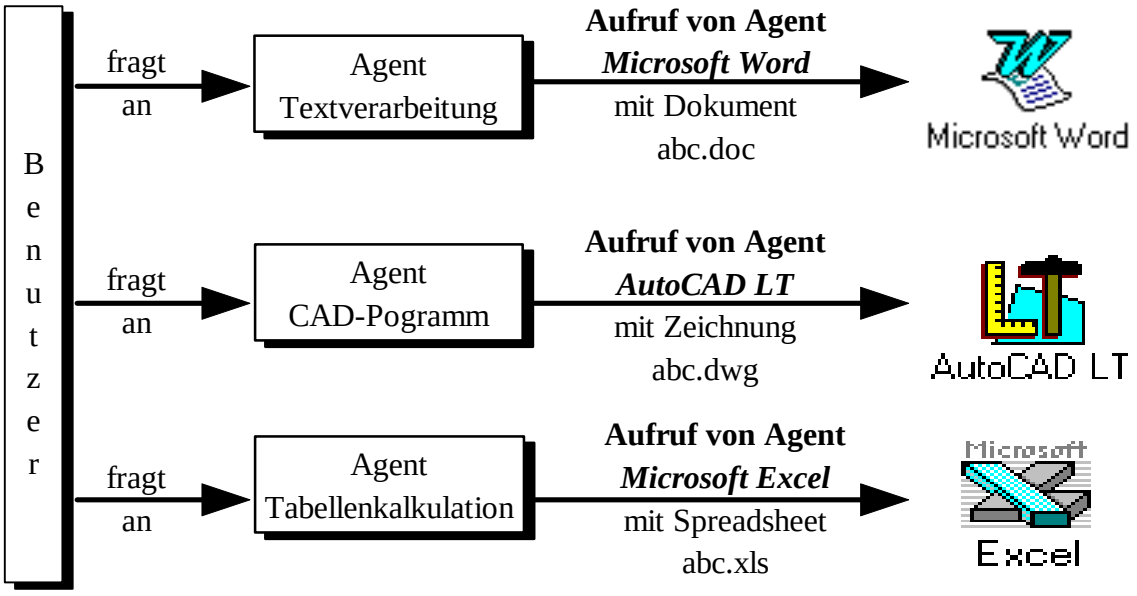


Abbildung 4.5: *Beispielagenten für Programmaufrufe mit entsprechenden Dateien*

Wenn beispielsweise ein Anwender einen Wareneingang eines Teils buchen will, so erhält er als Ausgabe den *Teilename* und die *Teile-Nr.* im Nur-Lese-Modus und den *Zugang* im Schreib-/Lese-Modus. Eine mögliche Aufbereitung zeigt Abbildung 4.6, wobei das grau unterlegte Feld *Zugang* verdeutlichen soll, daß dieser Wert im Gegensatz zu den anderen beiden Attributen veränderbar ist.

Teilename	**Teile-Nr.**	**Zugang**
Kantholz 30x30x3000	123	300

Abbildung 4.6: *Ausgabe von veränderbaren (grau unterlegt) und nicht veränderbaren Attributen aus einer Datenbank*

Im folgenden werden nun die für einige Funktionsgruppen der Beispielfirma *Baukado* relevanten Agenten vorgestellt.

4.2.1 Benötigte Agenten für *Baukado*

In Kapitel 2 wurden die Bereiche *Forschung und Entwicklung, Fertigung, Beschaffung und Materialwirtschaft* und *Rechnungswesen* auf ihre Aufgabenbereiche hin untersucht. In diesem Abschnitt werden nun die für die Informationsbereitstellung und -aufbereitung benötigten Agenten von Teilbereichen der einzelnen Funktionsgruppen aufgezeigt.

Forschung und Entwicklung

Der Teilbereich *Konstruktion* benötigt Agenten zur Bereitstellung von:

- Konstruktionszeichnungen
- Stücklisten der Baukästen

Fertigung

Der Teilbereich *Arbeitsplanung* benötigt Agenten zur Bereitstellung von:

- Stücklisten und Fertigungsdaten aus der Entwicklungsabteilung zur Erstellung oder Aktualisierung von Arbeitsplänen
- Absatzprognosen zur Ermittlung des Primärbedarfs (Bedarf an Ressourcen)
- Arbeitsplänen der Einzelteile zur Bestimmung der Reihenfolge der Bearbeitung und der benötigten Betriebsmittel
- Endterminen der Baukästen zur Ermittlung der Starttermine der einzelnen Arbeitsgänge

Beschaffung und Materialwirtschaft

Der Teilbereich *Bestellwesen* benötigt Agenten zur Bereitstellung von:

- Stücklisten und Produktionsplänen zur Bestimmung des Bedarfs aller Materialien
- Bedarfsmengen, Lagerbeständen und Mindestbeständen der Materialien zur Ermittlung der Bestellmengen
- Lagerabgangsprognosen aus der vorherigen Periode, Beständen, Mindestbeständen und Wiederbeschaffungszeiten zur Ermittlung des Bestellzeitpunktes
- Lieferantenstammdaten zur Auswahl eines geeigneten Lieferanten

- Abgängen von Endprodukten und Zugängen von Materialien für die Lagerbestandsführung

Rechnungswesen

Der Teilbereich *Kostenträgerrechnung* benötigt Agenten zur Bereitstellung von:

- Mengen der benötigten Materialien aus den Teilestammdaten und dem Wert des Materials aus den Materialstammdaten zur Bestimmung der Materialeinzelkosten
- Materialgemeinkosten aus der Kostenstellenrechnung
- Arbeitszeiten aus den Arbeitsplänen und Lohnsätzen aus der Kostenstellenrechnung zur Bestimmung der Fertigungslöhne
- Mengen aus den Stücklisten, Stückzeiten und Rüstzeiten aus den Arbeitsplänen, Maschinenstundensätze aus der Kostenstellenrechnung zur Bestimmung der Maschinenkosten
- Restgemeinkosten aus den Kostenstellen

Aus den Materialeinzelkosten, Materialgemeinkosten, Fertigungslöhnen, Maschinenkosten und Restgemeinkosten berechnen sich die Herstellkosten.

Neben den speziellen Informationsaufbereitungen müssen den verschiedenen Bereichen beispielsweise auch Zugänge zu Textverarbeitungsprogrammen und Arbeitspapieren zur Verfügung gestellt werden. Dabei dürfen die Benutzer nur auf die Arbeitspapiere zugreifen, für die sie eine Berechtigung haben. In [Frühwacht 1995] sind Lösungen zum Zugriffsschutz von Dokumenten beschrieben.
Abbildung 4.7 zeigt die notwendigen Agenten für die verschiedenen Informationsaufbereitungen einiger Bereiche von *Baukado*, wobei die Felder die Agenten darstellen, die die beschriebene Funktion erfüllen.. Beispielsweise ist Agent *Abgang / Zugang* dafür zuständig, die benötigten Attribute für die Buchungen aus dem Informationssystem zu filtern, wobei die Attribute *Teilename* und *Teile-Nr.* im Nur-Lese-Modus und das Attribut *Zugang* bzw. *Abgang* im Schreib-/Lese-Modus ausgegeben werden (vgl. Abbildung 4.6).

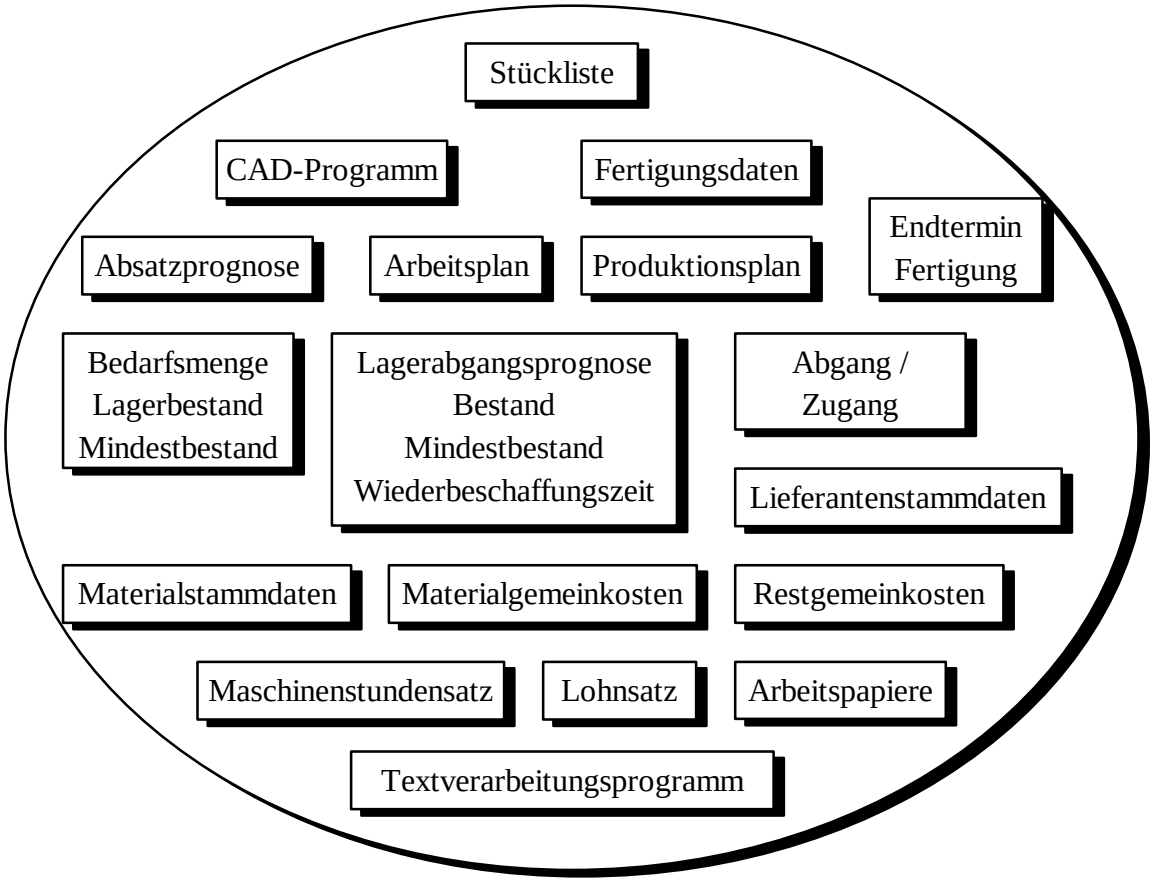


Abbildung 4.7: *Benötigte Agenten für die Informationsaufbereitungen einiger Bereiche von Baukado*

Im folgenden wird die Schnittstelle zwischen Benutzermodell und Agenten beschrieben. Sie ist dafür zuständig, bei Benutzeranfragen auf der Basis der Einträge im Benutzermodell die entsprechenden Agenten aufzufordern, gewünschte Informationen bereitzustellen und aufzubereiten.

4.3 Schnittstelle zwischen Benutzermodell und Agenten

Die Schnittstelle zwischen Benutzermodell und Agenten besteht aus mehreren Programmen (intelligente Agenten), deren Aufgaben darin bestehen, aufgrund der Einträge im Benutzermodell und der spezifischen Anfrage des Anwenders zu entscheiden, welcher Agent zur Beschaffung und Aufbereitung der Information anzusprechen ist. Dies bedeutet, daß zu jeder möglichen Anfrage, die aus den verschiedenen Aufgabenbereichen der Benutzer resultiert, eine bestimmte Informationsaufbereitung existiert. Im folgenden wird der Begriff "Schnittstelle" gleichbedeutend mit Schnittstellenprogramm verwendet.

Ebenso wie das Benutzermodell und die benötigten Agenten hängt die Ausprägung der Schnittstelle von der Art des Betriebes ab. Die Anfragen der Benutzer und die benötigten Informationsaufbereitungen sind die Basis für die Schnittstelle. Die Zuordnungen der verschiedenen Informationsaufbereitungen zu konkreten Anfragen der Anwender sind in der Schnittstelle gespeichert.
Im folgenden werden nun diese Zuordnungen konkret für *Baukado* dargestellt.

4.3.1 Schnittstelle zwischen Benutzermodell und Agenten für *Baukado*

In der Schnittstelle für *Baukado* ist festgelegt, welche Funktionen bzw. Anfragemöglichkeiten jedem Anwender aufgrund seiner Funktionsgruppenzugehörigkeit zur Verfügung gestellt werden. Im folgenden werden einige Anfragemöglichkeiten der verschiedenen Funktionsgruppen von *Baukado* vorgestellt.

Konstruktion

- Konstruktionszeichnungen
- Stücklisten

Arbeitsplanung

- Arbeitspläne
- Primärbedarf
- Starttermine der Arbeitsgänge

Bestellwesen

- Bedarfsmenge
- Bestellmenge
- Bestellzeitpunkt
- Lagerbestandsführung

Kostenträgerrechnung

- Materialeinzelkosten
- Materialgemeinkosten
- Fertigungslöhne
- Maschinenkosten
- Restgemeinkosten

In der Eingangsmaske für *Baukado* gibt der Benutzer seine im System eindeutige Benutzerkennung ein. In der Schnittstelle wird anhand der Einträge im Benutzermodell des jeweiligen Anwenders die Funktionsgruppenzugehörigkeit überprüft.

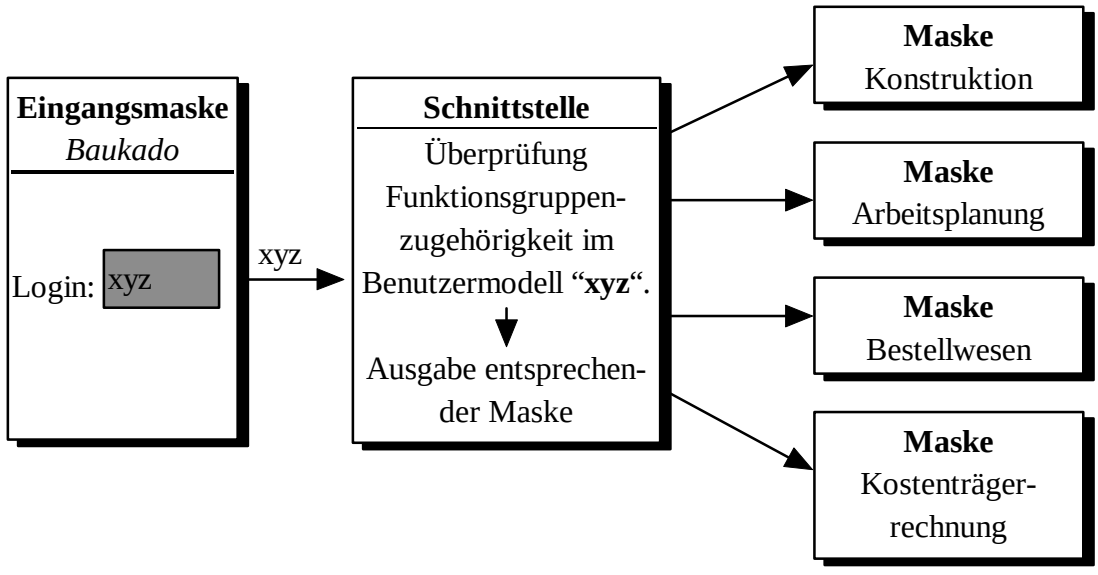


Abbildung 4.8: *Bereitstellung spezifischer Funktionen aufgrund der Funktions-gruppenzugehörigkeit*

Abhängig davon wird dann dem Benutzer die entsprechende Maske mit den für ihn relevanten Funktionen und Anfragemöglichkeiten ausgegeben (siehe Abbildung 4.8). Allgemeine Funktionen, wie Textverarbeitung, Zugang zu Arbeitspapieren etc. werden hier der Einfachheit halber vernachlässigt.

Bei Auswahl einer Anfragefunktion muß der Benutzer zunächst angeben, worüber er Informationen wünscht, z.B. Name des *Baukastens* oder *Einzelteils*, *Artikel-Nr.*, *Zeitraum* etc. Aufgrund der übrigen Attribute (*Qualifikation*, *Kenntnisstand*) des Benutzermodells wird dann die entsprechende Information aufbereitet. Abbildungen 4.9 bis 4.12 zeigen die unterschiedlichen Informationsaufbereitungen der einzelnen Funktionsgruppen. Abbildung 4.9 demonstriert zusätzlich die verschiedenen Aufbereitungen innerhalb einer Abteilung aufgrund unterschiedlicher *Qualifikation* und *Kenntnisstand*.

Informationsaufbereitung für Konstruktion			
Anfrage	**Qualifikation**	**Kenntnisstand**	**Informationsaufbereitung**
Konstruktions-zeichnung	sehr gut	Experte	Konstruktionszeichnung veränderbar
Konstruktions-zeichnung	Auszubil-dender	Anfänger	Test-Konstruktionszeichnung
Stückliste	sehr gut	Experte	Stückliste veränderbar
Stückliste	Auszubil-dender	Anfänger	Test-Stückliste

Abbildung 4.9: *Informationsaufbereitungen für den Bereich "Konstruktion"*

Abbildung 4.9 zeigt, daß ein Mitarbeiter mit der Qualifikation *sehr gut* und dem Kenntnisstand *Experte* auf Anfrage eine Konstruktionszeichnung erhält und verändern darf. Hingegen erhält ein Mitarbeiter mit der Qualifikation *Auszubildender* und Kenntnisstand *Anfänger* lediglich eine Test-Konstruktionszeichnung zum Erlernen der Funktionen des CAD-Programms. Dies gilt analog für Stücklisten.

Informationsaufbereitung für Arbeitsplanung			
Anfrage	**Qualifikation**	**Kenntnisstand**	**Informationsaufbereitung**
Arbeitsplan	gut	Fortgeschrittener	Stückliste im Nur-Lese-Modus Arbeitsplan veränderbar
Primärbedarf	gut	Fortgeschrittener	Absatzprognosen im Nur-Lese-Modus Primärbedarf veränderbar
Starttermine der Arbeits-gänge	gut	Fortgeschrittener	Endtermine der Baukästen im Nur-Lese-Modus Starttermine veränderbar

Abbildung 4.10: *Informationsaufbereitungen für den Bereich "Arbeitsplanung"*

Abbildung 4.10 zeigt spezielle Informationsaufbereitungsarten bei unterschiedlichen Anfragen eines Mitarbeiters des Bereichs *Arbeitsplanung*.
Bei Anfrage eines *Arbeitsplanes* wird die dafür benötigte Stückliste im Nur-Lese-Modus und der entsprechende Arbeitsplan veränderbar ausgegeben.
Bei Auswahl des Punktes *Primärbedarf* werden die für dessen Berechnung benötigten Absatzprognosen und der Primärbedarf aus dem Informationssystem gefiltert und im Nur-Lese-Modus bzw. veränderbar ausgegeben.
Zur Ermittlung der Starttermine der einzelnen Arbeitsgänge werden aus den vorgegebenen Endtermine die Starttermine berechnet und im Informationssystem abgespeichert.

Informationsaufbereitung für Bestellwesen			
Anfrage	**Qualifikation**	**Kenntnisstand**	**Informationsaufbereitung**
Bedarfsmenge	überdurch-schnittlich	Experte	Stückliste im Nur-Lese-Modus Materialbedarf veränderbar
Bestellmenge	überdurch-schnittlich	Experte	Bedarfsmenge, Bestand und Mindestbestand im Nur-Lese-Modus Bestellmenge veränderbar
Bestellzeit-punkt	überdurch-schnittlich	Experte	Lagerabgangsprognose, Bestand, Mindestbestand und Wiederbeschaffungszeit im Nur-Lese-Modus Bestellzeitpunkt veränderbar
Lagerbestands-führung	überdurch-schnittlich	Experte	Abfrage Zugang oder Abgang

Abbildung 4.11: *Informationsaufbereitungen für den Bereich "Bestellwesen"*

In Abbildung 4.11 werden die unterschiedlichen Informationsaufbereitungen für einen Mitarbeiter der Abteilung *Bestellwesen* mit der Qualifikation *überdurchschnittlich* und dem Kenntnisstand *Experte* dargestellt.
Bei Anfrage der *Bedarfsmenge* erhält der Anwender als Ausgabe die entsprechende Stückliste im Nur-Lese-Modus, berechnet den Materialbedarf und schreibt diesen Wert in die Datenbank des Informationssystems.
Zur Bestimmung der *Bestellmenge* werden *Bedarfsmenge*, *Bestand* und *Mindestbestand* im Nur-Lese-Modus ausgegeben.
Bei Auswahl des Punktes *Bestellzeitpunkt* werden die dafür benötigten Daten - *Lagerabgangsprognose, Bestand, Mindestbestand* und *Wiederbeschaffungszeit* - im Nur-Lese-Modus und der *Bestellzeitpunkt* veränderbar ausgegeben.
Wird der Punkt *Lagerbestandsführung* ausgewählt, erfolgt die Nachfrage, ob *Abgang* oder *Zugang* gewünscht wird.

Informationsaufbereitung für Kostenträgerrechnung			
Anfrage	**Qualifikation**	**Kenntnisstand**	**Informationsaufbereitung**
Materialeinzel-kosten	gut	Experte	Bedarfsmenge und Wert im Nur-Lese-Modus Materialeinzelkosten veränderbar
Materialge-meinkosten	gut	Experte	Materialgemeinkosten im Nur-Lese-Modus
Fertigungs-löhne	gut	Experte	Minuten aus den Arbeitsplänen und Lohnsatz im Nur-Lese-Modus Fertigungslohn veränderbar
Maschinen-kosten	gut	Experte	Menge, Stückzeit, Rüstzeit und Maschinenstundensatz im Nur-Lese-Modus Maschinenkosten veränderbar
Restgemein-kosten	gut	Experte	Restgemeinkosten im Nur-Lese-Modus

Abbildung 4.12: *Informationsaufbereitungen für den Bereich "Kostenträgerrechnung"*

Abbildung 4.12 zeigt die Informationsaufbereitungen für einen Mitarbeiter der Abteilung *Kostenträgerrechnung*.
Bei Auswahl der verschiedenen Punkte werden die zugehörigen Attribute aus dem Informationssystem gefiltert und ausgegeben.

In den Abbildungen wurden lediglich einige Informationsaufbereitungsarten der fünf Beispielanwender dargestellt. Bei der Realisierung aller Funktionsgruppen mit allen notwendigen Funktionen und Anfragemöglichkeiten wird die Aufgabe der Schnittstelle sehr komplex. Daher besteht die Notwendigkeit, Schnittstellenprogramme übersichtlich und modular zu gestalten, so daß Erweiterungen problemlos erfolgen können.

4.4 Zusammenfassung

In diesem Kapitel wurde ein Konzept eines Architekturmodells für die adaptive Informationsaufbereitung im betrieblichen Umfeld vorgestellt. Aufgaben und Zusammenwirken der drei maßgeblichen Komponenten des Modells - *Benutzermodell*, *Agenten* und *Schnittstelle zwischen Benutzermodell und Agenten* - wurden zunächst theoretisch erklärt und anhand der Beispielfirma *Baukado* konkretisiert.

Im Benutzermodell sind

- Funktionsgruppenzugehörigkeit
- Qualifikation
- Kenntnisstand
- Präferenzen

jedes Anwenders beschrieben.

Die Agenten stellen Funktionen zur Informationsbeschaffung und -aufbereitung zur Verfügung.

In der Schnittstelle sind die Zuweisungen der entsprechenden Informationsaufbereitungen zu den Anwendern gespeichert.

Aufgrund der Funktionsgruppenzugehörigkeit werden jedem Benutzer die für ihn relevanten Funktionen und Abfragemöglichkeiten zur Verfügung gestellt. Nachdem ein Benutzer eine Anfrage ausgewählt hat, muß er zunächst angeben, welche Information gewünscht wird. Diese Information und die Benutzerkennung werden an ein Schnittstellenprogramm übergeben. Anhand der Eingaben werden aus dem Benutzermodell die Attribute *Qualifikation* und *Kenntnisstand* des Anwenders eingelesen, woraufhin das Schnittstellenprogramm aufgrund der individuellen Kombination der beiden Attribute die entsprechenden Agenten auffordert, die gewünschte Information zu liefern.

Im folgenden Kapitel wird die Implementierung einiger Informationsaufbereitungsarten für die Beispielfirma *Baukado* vorgestellt.

5 Implementierung

In diesem Kapitel wird das erarbeitete Konzept eines Architekturmodells für die adaptive Informationsaufbereitung anhand einiger Bereiche der Beispielfirma *Baukado* implementiert.
Die Benutzungsoberfläche wird mit Hilfe von HTML-Dokumenten und eines Hypertext-Browsers realisiert. Die Implementierung der Schnittstelle erfolgt durch CGI-Programme mit der Skriptsprache *Perl*. Das Benutzermodell wird für jeden Anwender in einer Textdatei abgespeichert, wobei die Präferenzen des Benutzers nicht berücksichtigt werden. Die intelligenten Agenten zur Informationsbeschaffung und -aufbereitung werden durch HTML-Dokumente simuliert.

Zunächst wird ein allgemeiner Überblick über die berücksichtigten Module gegeben. Der Benutzer muß sich durch ein Login eindeutig im System identifizieren. Der Benutzername muß stets verfügbar sein, da bei jeder Anfrage auf die individuellen Einträge des Benutzermodell zugegriffen werden muß. Die Attribute des Benutzermodells sind für jeden Benutzer in einer Textdatei abgespeichert. Der Name der Textdatei entspricht dem Namen des Benutzers.

Benutzermodell <Benutzername>:

Textdatei <Benutzername>:

```
<Attribut 1>
<Attribut 2>
...
<Attribut n>
```

Nachdem sich ein Benutzer eingelogt hat, wird sein Name an ein CGI-Programm übergeben, welches das Benutzermodell einliest und aufgrund der Funktionsgruppenzugehörigkeit dynamisch die Homepage dieser Abteilung mit den entsprechenden Anfragemöglichkeiten generiert.
Bei Auswahl einer Anfrage erscheint ein Eingabeformular, in dem der Benutzer angibt, welche Information er wünscht. Diese Anfrage und der Benutzername werden an ein CGI-Programm übergeben, welches dann wiederum das Benutzermodell einliest und

aufgrund der Qualifikation und des Kenntnisstandes die Information in einer für den Anwender geeigneten Form aufbereitet. Den Verlauf einer Sitzung vom Einlogvorgang über die Anfrage bis zur Darstellung der Information wird in Abbildung 5.1 dargestellt.

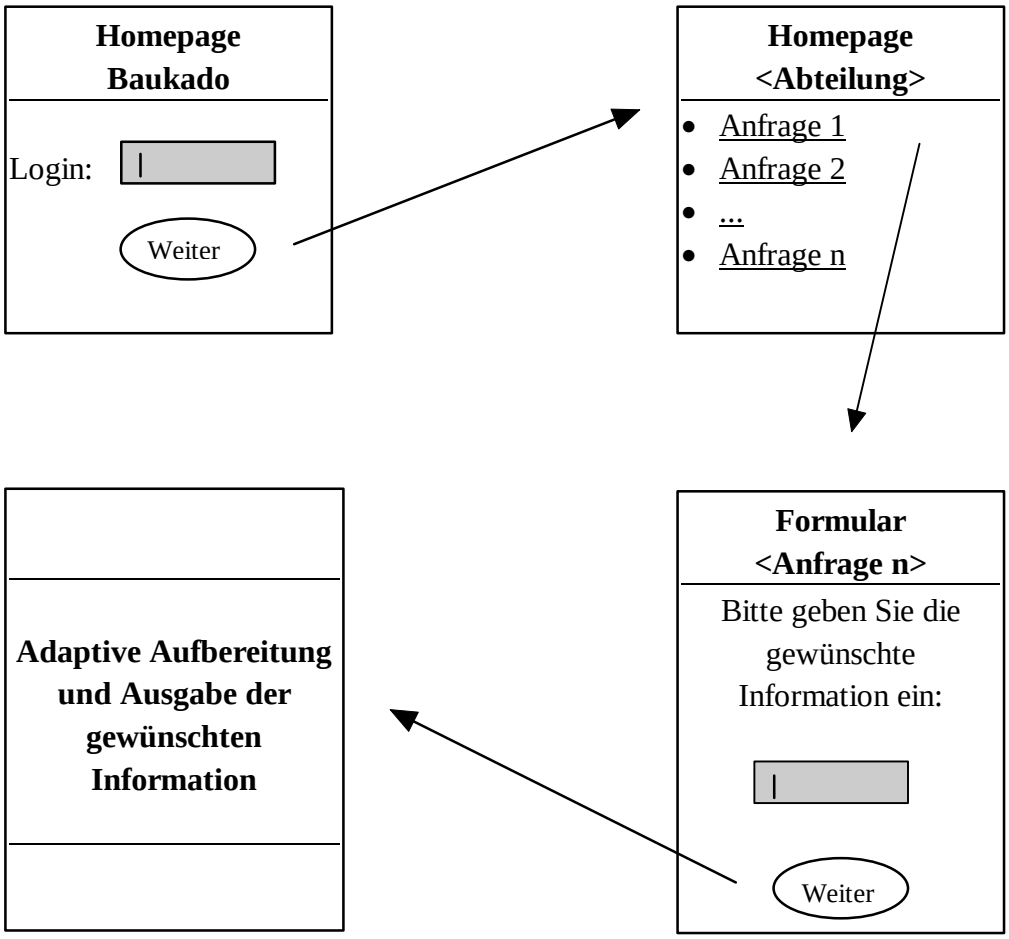


Abbildung 5.1: *Allgemeiner Verlauf einer Anfrage*

Im folgenden werden das Benutzermodell und die Programme anhand eines Anfragebeispiels für *Baukado* vorgestellt. Benutzer *Bach* aus der Abteilung *Konstruktion* mit der Qualifikation *sehr gut* und dem Kenntnisstand *Experte* fragt die Konstruktionszeichnung des Teils *Quader1* an.

5.1 Benutzermodell

Im Benutzermodell sind *Funktionsgruppenzugehörigkeit*, *Qualifikation* und *Kenntnisstand* des Anwenders abgespeichert. Das Benutzermodell von Mitarbeiter *Bach* sieht wie folgt aus:

Benutzermodell Bach:

Textdatei *bach:*

```
Konstruktion
sehr gut
Experte
```

Bei jeder möglichen Auswahl einer Anfrage durch den Benutzer, wird sein Benutzermodell eingelesen und aufgrund der individuellen Einträge die gewünschte Information entsprechend aufbereitet.

Die Routine zum Einlesen des Benutzermodells sieht in *Perl* wie folgt aus:

```
# Aufruf der Sub-Routine bm_einlesen mit dem Parameter
# bm_file (z.B. bach)

&bm_einlesen($bm_file);

sub bm_einlesen {
     $i=0;
     # Öffnen der Datei $bm_file (z.B. bach)
     open (BM,"/lasagne/users/maile/baukado/bm/$bm_file");
     # Zeilenweise lesen aus einer Textdatei bis EOF
     while ($name = <BM>) {
          # entfernt das letzte Zeichen von $name (CR)
          chop($name);
          # Schreiben der einzelnen Zeilen aus der
          # Textdatei in einen Array
          $bm[$i] = $name;
          $i++;
```

```
          }
close(BM);
}
```

Auf die Attribute des Modells kann folgendermaßen zugegriffen werden:

$bm[i]

wobei *i* einen Wert von 0 bis 2 annehmen kann, da die Textdatei aus drei Zeilen besteht. Im Falle des Benutzers *Bach* sieht der Inhalt des Arrays *bm* wie folgt aus:

```
$bm[0]: Konstruktion
$bm[1]: sehr gut
$bm[2]: Experte
```

Um dem Benutzer beispielsweise die für ihn relevanten Anfragemöglichkeiten aufgrund seiner Funktionsgruppenzugehörigkeit zur Verfügung zu stellen, wird der Inhalt des ersten Feldes des Arrays *bm* abgefragt. Die Abfrage lautet in *Perl*:

```
if ($bm[0] eq “Konstruktion“) {
     ...
}
```

Im folgenden werden einige HTML-Dokumente und CGI-Programme zur Realisierung des Architekturmodells für die Beispielfirma *Baukado* vorgestellt.

5.2 HTML-Dokumente und CGI-Programme

Die Startseite für *Baukado* enthält zur Veranschaulichung die Namen aller Mitarbeiter von Baukado. Die Hervorhebung der Einträge ist ein Link auf das CGI-Programm *benutzermodell.pl* mit Übergabe des Parameters *name=<Benutzername>*. Dadurch wird der Einlogvorgang simuliert. Zu Demonstrationszwecken wird zunächst das Benutzermodell des jeweiligen Anwenders gezeigt. Der Name des Benutzers muß an

jedes aufgerufene CGI-Programm geschickt werden, damit Zugriff auf die Einträge des Benutzermodells möglich ist.

In der Praxis kann jedoch direkt auf den Benutzernamen zugegriffen werden. Die entsprechende *Perl*-Routine sieht wie folgt aus [Schwartz 1993 S. 25]:

```
@password = getpwuid($<);
$name = $password[5];
$name =~ s/,.*//;
```

In der ersten Zeile werden die Paßwortdaten in den Array *@password* eingelesen. Die Variable *$* enthält automatisch die UNIX *User-ID number*. Der Befehl *getpwuid()* hat als Eingabe eine *User-ID number* und gibt eine Liste mit den Informationen der Paßwortdatei zurück.
Meist steht der Benutzername an sechster Stelle in der Liste. Da auf das erste Feld des Arrays mit *$password[0]* zugegriffen wird, erhält man den Inhalt des sechsten Feldes mit *$password[5]*.
Dem Benutzernamen folgen üblicherweise weitere Daten, die durch Kommata voneinander getrennt sind. Daher wird in der dritten Zeile alles nach dem ersten Komma entfernt.
In *$name* steht schließlich der Name des Benutzers.

Von der HTML-Seite mit der Anzeige des Benutzermodells wird wiederum der Name des Benutzers an ein CGI-Programm - *homepage.pl* - übergeben, welches aufgrund der Funktionsgruppenzugehörigkeit dynamisch die Homepage der entsprechenden Abteilung mit den relevanten Anfragemöglichkeiten erzeugt.
Bei Auswahl einer Anfrage wird der *Benutzername* an das dazugehörige CGI-Programm versendet, welches dynamisch ein Eingabeformular erzeugt, in dem der Benutzer angibt, über welches Teil oder welchen Baukasten Informationen gewünscht werden.
Von diesem Formular aus werden die Parameter *Benutzername* und *Teilename* wiederum an ein CGI-Programm übergeben, welches aufgrund der *Qualifikation* und des *Kenntnisstandes* die gewünschte Information entsprechend aufbereitet.
Abbildung 5.2 zeigt den Verlauf einer Anfrage durch Benutzer *Bach*. Nach Anwahl von "*Bach*" (Simulation des Login) in der Eingangsseite von *Baukado*, wird dessen Benutzermodell angezeigt.

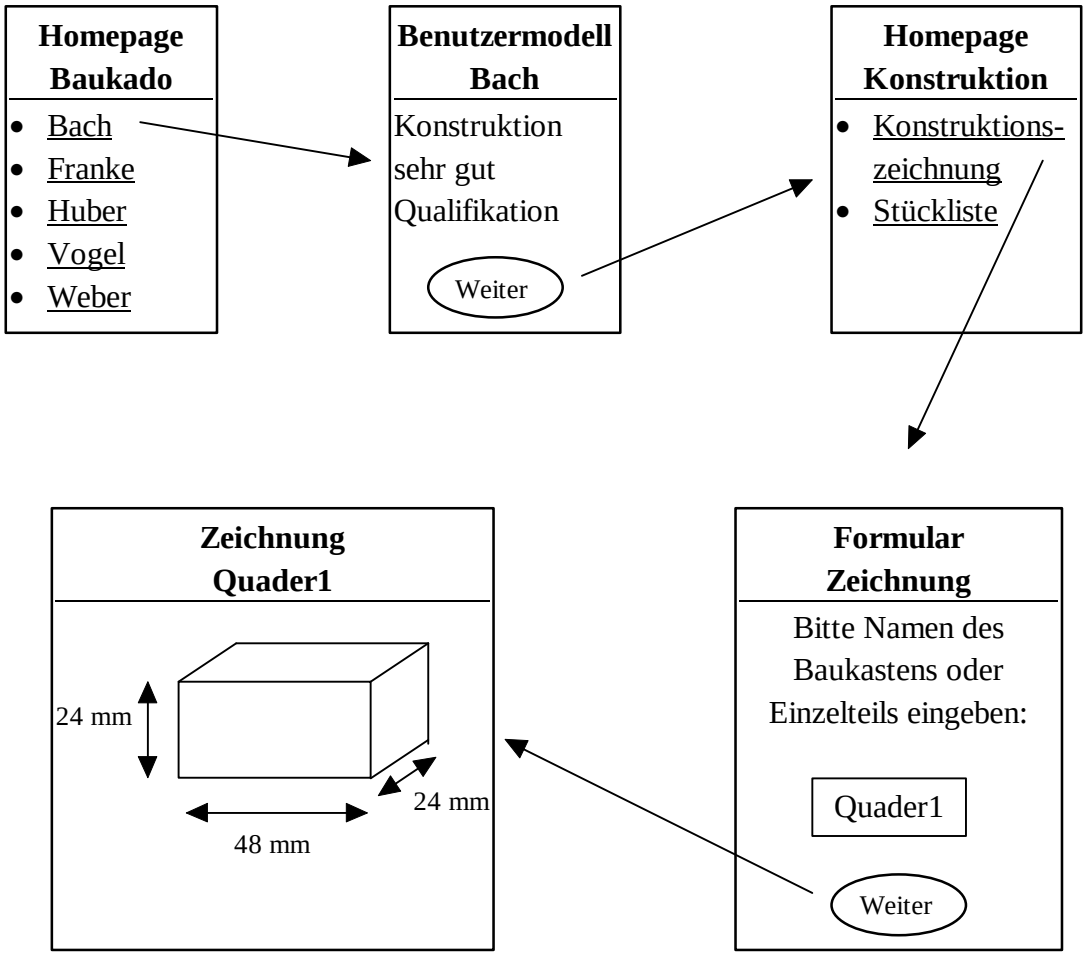


Abbildung 5.2: *Verlauf der Anfrage zur Konstruktionszeichnung des Teils "Quader1" durch Benutzer "Bach"*

Auf Anklicken des Buttons "*Weiter*" erscheint die Homepage *Konstruktion*. Nachdem "*Konstruktionszeichnung*" ausgewählt wird, erfolgt die Darstellung des HTML-Formulars *Formular Zeichnung*. Nach Eingabe von "*Quader1*" wird die Konstruktionszeichnung des gewünschten Teils ausgegeben.

Im folgenden werden die benötigten HTML-Dokumente und die CGI-Programme zur dynamischen Erzeugung der Dokumente für *Baukado* beschrieben und am Beispiel von Benutzer *Bach* erklärt.

Eingangsseite für Baukado:

baukado.html:

```
<HTML>
<HEAD>
<TITLE>Homepage Baukado</TITLE>
</HEAD>
<HR><H1>Homepage Baukado</H1><HR>
<BODY>
<p>
<p>Mitarbeiter:
<p>
<p><a href="http://farfalle.informatik.uni-stuttgart.de/
     cgi-bin/maile/benutzermodell.pl?name=Bach">Bach </a>
<p><a href="http://farfalle.informatik.uni-stuttgart.de/
     cgi-bin/maile/benutzermodell.pl?name=Franke">Franke
     </a>
<p><a href="http://farfalle.informatik.uni-stuttgart.de/
     cgi-bin/maile/benutzermodell.pl?name=Huber">Huber </a>
<p><a href="http://farfalle.informatik.uni-stuttgart.de/
     cgi-bin/maile/benutzermodell.pl?name=Vogel">Vogel </a>
<p><a href="http://farfalle.informatik.uni-stuttgart.de/
     cgi-bin/maile/benutzermodell.pl?name=Weber">Weber </a>
</BODY>
</HTML>
```

Abbildung 5.3 stellt die Homepage für *Baukado* dar. Wird "*Bach*" angeklickt, so wird die CGI-Umgebungsvariable QUERY-STRING mit dem Inhalt *name=Bach* an das CGI-Programm *benutzermodell.pl* geschickt und dort weiterverarbeitet.

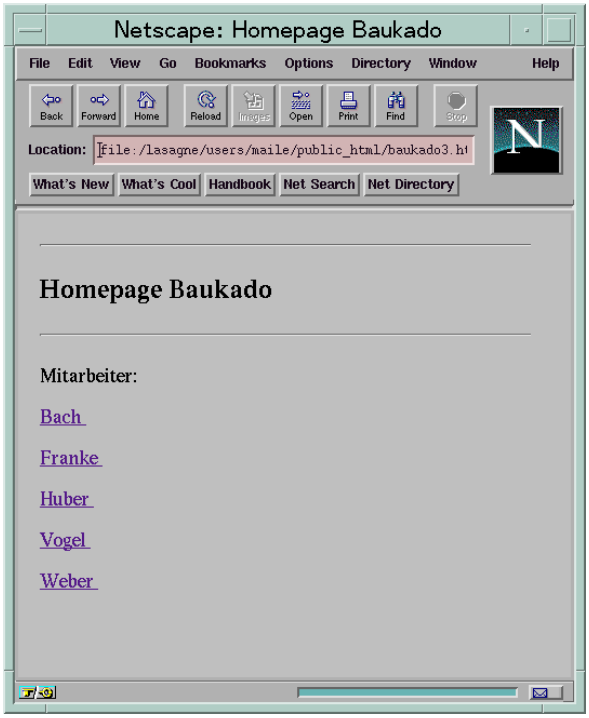


Abbildung 5.3: *Homepage von Baukado (Login-Simulation)*

CGI-Programm zur Anzeige des Benutzermodells:

benutzermodell.pl

```
01   #!/usr/remote/bin/perl
02   # Übergabe des vorhandenen Parameters, z.B.
03   # Benutzer an die Variable $in
04   if ($ENV{'REQUEST_METHOD'} eq "GET") {
05       $in = $ENV{'QUERY_STRING'};
06   } elsif ($ENV{'REQUEST_METHOD'} eq "POST") {
07       for ($i = 0; $i < $ENV{'CONTENT_LENGTH'};
08    $i++) {
09       $in .= getc;   }
10   }
11   # Das Name-Wert-Paar wird zerlegt und kodierte
12   # Sonderzeichen werden dekodiert.
```

```
13   ($name,$bm_file) = split(/=/,$in);
14   $bm_file =~ tr/+/ /;
15   $bm_file =~ s/%([a-fA-F0-9])/pack("C",hex($1))/eg;
16   # Grossbuchstaben durch Kleinbuchstaben ersetzen
17   $bm_file =~ tr/A-Z/a-z/;
18   # Der Benutzername ist Inhalt der Variablen $bm_file.
19
20   print <<“Ende“;
21   Content-type:text/html
22   # Leerzeile nach dem Header unbedingt notwendig
23   \n\n
24   <HTML>
25   <HEAD>
26   <TITLE>Benutzermodell</TITLE>
27   </HEAD>
28   <BODY>
29   <HR><H1>Benutzermodell $bm_file</H1><HR>
30   Ende
31
32   &bm_einlesen($bm_file);
33
34   print <<“Ende“;
35   Funktionsgruppe:$bm[0]
36   <p>
37   Qualifikation  :$bm[1]
38   <p>
39   Kenntnisstand  :$bm[2]
40   <p>
41   <p>
42   <FORM> METHOD = POST ACTION="http://farfalle.
     informatik.uni-stuttgart.de/cgi-bin/maile/homepage.pl“
43   <INPUT NAME=user TYPE=hidden VALUE=“$bm_file“>
44   <INPUT TYPE=submit VALUE=“Weiter“
45   </FORM>
46   </BODY>
47   </HTML>
48   \n\n
48   Ende
50
```

```
51   sub bm_einlesen {
52        $i=0;
53        open (BM,"/lasagne/users/maile/baukado/
               bm/$bm_file");
54        while($attribut = <BM>){
55             chop($attribut);
56             $bm[$i] = $attribut;
57             $i++;
58             }
59        close (BM);
60        }
61
```

Die Zeilennumerierung dient lediglich zur besseren Referenzierung.

In Zeile 1 wird angegeben, daß es sich um ein Perl-Programm handelt und wo sich Perl befindet.

Von Zeile 4 bis 10 wird der Inhalt der Umgebungsvariablen QUERY-STRING in die Variable *$in* geschrieben. Dies kann auf zwei Arten erfolgen, je nachdem welche Methode des Versendens - *GET* oder *POST* - gewählt wurde. Wenn keine Methode angegeben wird, so ist *GET* die Default-Methode.

Von Zeile 13 bis 17 wird der kodierte Inhalt der Umgebungsvariablen dekodiert.

Der Befehl in Zeile 20

```
print <<“Ende“;
```

gibt alles bis zum Schlüsselwort “*Ende*“ auf dem Ausgabegerät aus. Dies erspart einige Schreibarbeit, denn jede Zeile müßte entsprechend dem Beispiel:

```
Content-type: text/html
```

in

```
print “Content-type: text/html“;
```

geändert werden.

Die Zeilen 21 bis 30 erzeugen dynamisch den Anfang des HTML-Dokuments zur Darstellung des Benutzermodells.

Durch Zeile 32 wird das Unterprogramm *bm_einlesen* aufgerufen, um das Benutzermodell einzulesen.

Die Befehle in den Zeilen 34 bis 39 erzeugen die Ausgabe von Funktionsgruppenzugehörigkeit, Qualifikation und Kenntnisstand des Benutzers auf dem Bildschirm.

In den Zeilen 42 bis 46 wird ein ein HTML-Formular mit einem nicht sichtbaren Feld *user* definiert. Bei Anklicken des Submit-Buttons "*Weiter*" wird die Variable *user=<bm_file>* (im Beispiel *user=bach*) an das CGI-Programm *homepage.pl* geschickt. Im Programm *homepage.pl* werden wiederum die Daten des Benutzermodells des Anwenders, der in *bm_file* eingetragen ist, eingelesen. Aufgrund der Funktionsgruppenzugehörigkeit wird dann dynamisch die Homepage der Abteilung mit den relevanten Anfragemöglichkeiten erzeugt.

Abbildung 5.4: *Benutzermodell "Bach"*

Abbildung 5.4 zeigt die Attribute des Benutzermodells von Mitarbeiter *Bach*. Das Programm *benutzermodell.pl* dient nur zu Demonstrationszwecken und entfällt in der Praxis, da das Benutzermodell normalerweise nicht angezeigt werden muß. Zudem wird in der Realität die Eingangsseite allgemeine Funktionen, wie Textverarbeitungsprogramme, Zugriff auf Arbeitspapiere etc. und einen Link auf das Programm *homepage.pl* enthalten.

Im folgenden werden lediglich neue Programmteile erklärt. Diese sind **fett** hervorgehoben. Die in jedem Programm vorkommende Dekodierung des Inhalts der Umgebungsvariablen QUERY-STRING und das Einlesen der Daten des Benutzermodells werden nicht aufgeführt. Der vollständige Source-Code der Programme kann im Anhang eingesehen werden.

CGI-Programm zur dynamischen Generierung der Homepages für die einzelnen Funktionsgruppen:

homepage.pl:

```
    ...
14  $bm_file =~ tr/A-Z/a-z/;
15  # Der Benutzername ist Inhalt der Variablen $bm_file.
16
17  &homepage_anfang;
18
19  &bm_einlesen($bm_file);
20
21  print "<HR><H1>Homepage $bm[0]</H1><HR>";
22  if ($bm[0] eq "Konstruktion"){
23  print <<“Ende“;
24  <UL>
25  <LI> ‘<a href="http://farfalle.informatik.uni-
         stuttgart.de/cgi-bin/maile/formular_zeichnung.pl
         ?name=$bm_file">Konstruktionszeichnung</a>‘
26  <LI> ‘<a href="http://farfalle.informatik.uni-
         stuttgart.de/cgi-in/maile/formular_stueckliste.pl
         ?name=$bm_file">St&uuml;ckliste</a>‘
27  </UL>
```

```
28    Ende
29    }
30
31    &homepage_ende;
      ...
```

Der Befehl in Zeile 21 gibt die Funktionsgruppe des Anwenders aus. In Zeile 22 wird abgefragt, ob der Benutzer zur Funktionsgruppe K*onstruktion* gehört. Ist dies der Fall, so wird durch die Zeilen 23 bis 28 dynamisch die Homepage *Konstruktion* erzeugt. Der Anwender hat die Mö glichkeit, eine Konstruktionszeichnung oder eine Stückliste abzufragen.

Abbildung 5.5: *Homepage der Funktionsgruppe "Konstruktion"*

Abbildung 5.5 stellt die Homepage der Abteilung *Konstruktion* dar. Bei Anklicken des Links *Konstruktionszeichnung* wird die Variable *name=$bm_file* (im Beispiel *name=bach*) an das CGI-Programm *formular_zeichnung.pl* übergeben. Dieses Programm generiert ein HTML-Formular, in dem der Benutzer den Namen des Teils angibt, dessen Konstruktionszeichnung er wünscht.

Bei einem Klick auf den Link *Stückliste* wird die Variable *name=$bm_file* an das CGI-Programm *formular_stückliste.pl* übergeben. Dieses Programm generiert ein HTML-Formular, in dem der Benutzer den Namen des Teils angibt, dessen Stückliste er wünscht.
Das reale Programm enthält ab Zeile 30 die Abfragebefehle für die übrigen Funktionsgruppen und deren dazugehörigen Abfragemöglichkeiten. Dies wird hier der Übersichtlickeit halber nicht dargestellt.

CGI-Programm zur dynamischen Generierung des HTML-Formulars zur Eingabe der gewünschten Konstruktionszeichnung:

formular_zeichnung.pl:

```
    ...
14  $bm_file =~ tr/A-Z/a-z/;
15  # Der Benutzername ist Inhalt der Variablen $bm_file.
16
17  &homepage_anfang;
18
19  &bm_einlesen($bm_file);
20
21  print <<“Ende“;
22  FORM METHOD=POST ACTION=“http://farfalle.informatik.
        uni-stuttgart.de/cgi-bin/maile/zeichnung.pl"
23  INPUT NAME=user TYPE=hidden VALUE=“$bm_file“
24  <p>Bitte geben Sie den Namen des gewünschten Teils an
25  <p>
26  INPUT NAME=teil SIZE=“30“ MAXLENGTH=“30“
27  INPUT TYPE=submit VALUE=“Weiter“
28  INPUT TYPE=reset
29  </FORM>
30  Ende
31
32  &homepage_ende;
    ...
```

In den Zeilen 22 bis 29 wird ein HTML-Formular definiert, das ein nicht sichtbares Feld *user* mit dem Namen des Benutzers und ein Eingabefeld *teil* enthält. Bei der beispielhaften Eingabe des Namens des gewünschten Teils *Quader1* durch den Benutzer *Bach* wird in die CGI-Umgebungsvariable QUERY-STRING folgendes geschrieben:

user=bach&teil=Quader1

Abbildung 5.6 zeigt das HTML-Formular *Konstruktionszeichnung*.

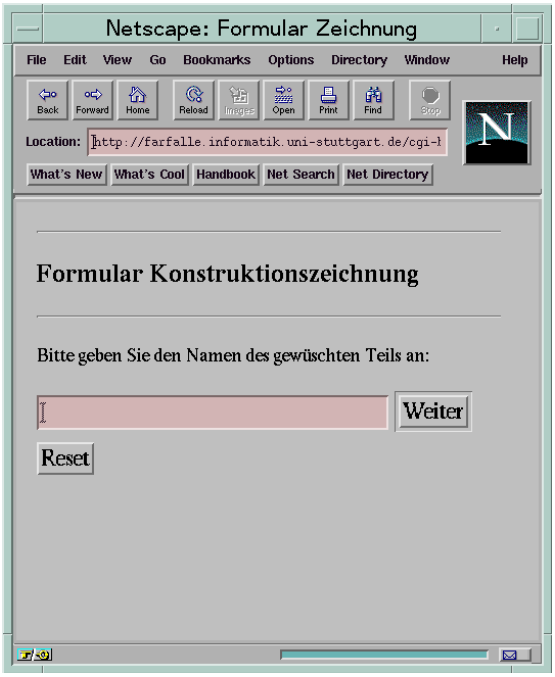


Abbildung 5.6: *Formular "Konstruktionszeichnung"*

In Zeile 22 wird definiert, daß die Variable QUERY-STRING an das Programm *zeichnung.pl* übergeben wird. Dieses Programm ist dafür zuständig, daß die entsprechende Konstruktionszeichnung auf dem Bildschirm des Anwenders ausgegeben wird.

CGI-Programm zur Ausgabe der gewünschten Konstruktionszeichnung:

zeichnung.pl:

```
01   #!/usr/remote/bin/perl
02   # Übergabe der vorhandenen Parameter, z.B.
03   # Benutzername und Teilename an die Variable $in
04   if ($ENV{'REQUEST_METHOD'} eq "GET") {
05       $in = $ENV{'QUERY_STRING'};
06   } elsif ($ENV{'REQUEST_METHOD'} eq "POST") {
07       for ($i = 0; $i < $ENV{'CONTENT_LENGTH'};
08    $i++) {
09       $in .= getc;   }
10   }
11   # Die Name-Wert-Paare werden getrennt und in das Array
12   # @paare geschrieben
13
14   @paare = split(/&/,$in)
15   # Die Name-Werte-Paare werden zerlegt und kodierte
16   # Sonderzeichen werden dekodiert.
17   foreach $paare(@paare){
18        ($name,$inhalt) = split(/=/,$paare);
19        $inhalt =~ tr/+/ /;
20        $inhalt =~ s/%([a-fA-F0-9])/pack("C",hex($1))/eg;
21        $inhalt =~ tr/A-Z/a-z/;
22        $felder{$name} = $inhalt;
23   }
24   $teil = $felder{$teil};
25   $bm_file = $felder{user}
26   # Der Benutzername ist Inhalt der Variablen $bm_file.
27   # Der Teilename ist Inhalt der Variablen $teile.
28   &bm_einlesen($bm_file);
29
30   if ($bm[1] eq “sehr gut“){
31        if ($bm[2] eq “Experte“){
32            print “Location:/maile/zeichnung-$teil.html“;
33        }
34   }
     ...
```

Das Programm *zeichnung.pl* erhält im Gegensatz zu den vorherigen Programmen durch die Umgebungsvariable QUERY-STRING zwei Variablen, *user* und *teil*. Deshalb werden durch den Befehl in Zeile 14 die beiden Variablen zunächst getrennt und in das Array *@paare* geschrieben.

Aufgrund der Einträge der Zeilen 17 bis 21 werden die Namen und zugehörigen Inhalte der Variablen getrennt und dekodiert.
In Zeile 22 wird der Inhalt der Variablen in das assoziative Array *felder* in Abhängigkeit vom Variablennamen geschrieben. Beispielsweise ist bei Inhalt von QUERY-STRING: *user=bach&teil=Quader1* der Inhalt des Arrays *felder:*

$felder{user] = bach

$felder{teil} = quader1

In Zeile 24 und 25 wird der Teilename der Variablen *$teil* und der Benutzername der Variablen *$bm_file* zugewiesen.

Durch die Abfragen in den Zeilen 30 und 31 werden Qualifikation und Kenntnisstand verifiziert. Ist die Qualifikation *sehr gut* und der Kenntnisstand *Experte*, so wird die HTML-Seite mit der Zeichnung des gewünschten Teils ausgegeben.

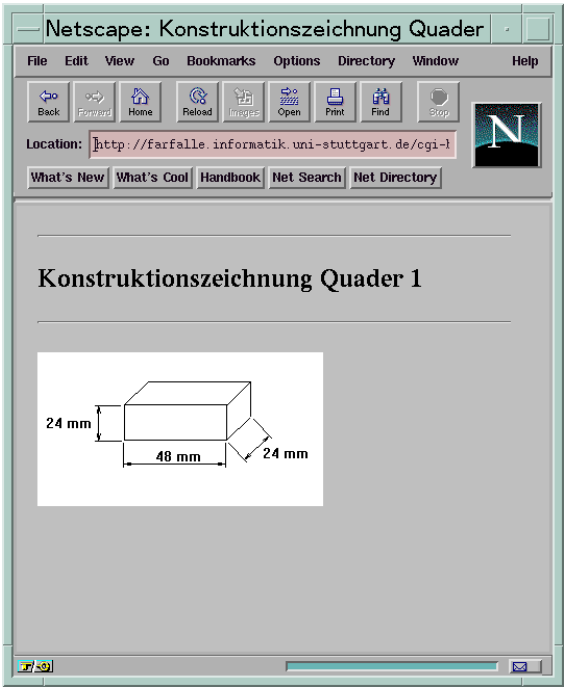


Abbildung 5.7: *Konstruktionszeichnung des Quaders1*

Wenn beispielsweise die Variable *$teil* den Inhalt *quader1* hat, so wird die HTML-Seite *zeichnung-quader1.html* ausgegeben. Abbildung 5.7 zeigt diese HTML-Seite mit der Konstruktionszeichnung von *Quader1*. Das Wort *Location* in Zeile 32 steht für den Pfad .../htdocs (z.B. *farfalle.informatik.uni-stuttgart.de/htdocs*).
In der Praxis wird an dieser Stelle ein CAD-Programm mit der entsprechenden Datei geladen.

5.3 Zusammenfassung

In diesem Kapitel wurden Teile des in Kapitel 4 entwickelten Architekturmodells prototypisch implementiert. Die Programme wurden am Beispiel des Benutzers *Bach*, der eine Konstruktionszeichnung des Teils *Quader1* anfordert, erklärt. Der vollständige Source-Code ist im Anhang zu finden.

Für die Benutzungsoberfläche wurden HTML-Dokumente in Verbindung mit einem Hypertext-Browser ausgewählt. Dabei sind lediglich die Eingangsseite von *Baukado* und die Informationsaufbereitung statische Dokumente. Alle übrigen Dokumente wurden in Abhängigkeit von den Inhalten des Benutzermodells dynamisch erzeugt.
Das Benutzermodell wurde für jeden Anwender in Form einer Textdatei realisiert. Diese Dateien können jederzeit verändert oder um weitere Attribute ergänzt werden. Außerdem kann eine Textdatei einfach von den in *Perl* erstellten CGI-Programmen eingelesen werden.

Die Schnittstelle wurde durch CGI-Programme mit Hilfe der Skriptsprache *Perl* verwirklicht. Die Sprache *Perl* wurde deshalb ausgewählt, da eine Stärke dieser Sprache in der Verarbeitung von Zeichenketten liegt. Es existieren zahlreiche Funktionen, die das Dekodieren der versendeten Variablen mühelos ermöglichen. Ebenso problemlos ist die Konkatenation von Zeichenketten. Programm *zeichnung.pl* zeigt dies in Zeile 28:

```
28   print "Location:/maile/zeichnung-$teil.html";
```

Eine Zeichenkette kann somit auf einfache Art und Weise mit dem Inhalt einer Variablen gekoppelt werden.

Die Implementierung der Agenten zur Beschaffung und Aufbereitung der gewünschten Informationen wäre zu umfangreich für den Rahmen dieser Diplomarbeit. Daher wurden die Ausgaben der Agenten durch HTML-Dokumente simuliert.

6 Fazit

"...there has been a tendency to drop the use of the phrase "data-processing-system" in favour of "information system", without actually changing the nature of the system... The distinction is that data only becomes information when meaning is attached through interpretation." [Wilson 1995, S. 2]

Die Daten eines Informationssystems werden erst dann zu Informationen, wenn sie interpretiert werden können. Daher besteht die Notwendigkeit, die Daten den Anwendern nicht in der Form zu präsentieren, wie sie abgespeichert sind, sondern sie in einer für den jeweiligen Mitarbeiter angepaßten Form aufzubereiten. Dies ist die Aufgabe der adaptiven Informationsaufbereitung. Dabei hängt die Art der Informationsaufbereitung von der Funktionsgruppenzugehörigkeit und den damit verbundenen Aufgabenbereichen des Benutzers ab. Beispielsweise wird in der Funktionsgruppe *Konstruktion* eine Konstruktionszeichnung des Produkts benötigt, während die Darstellungsform für die Marketingabteilung eine werbewirksame, dreidimensionale, farbige Graphik ist. Auch sind unterschiedliche Zugriffsrechte auf die Daten zu berücksichtigen. Z.B. werden in der Konstruktionsabteilung Stücklisten erstellt und verändert. In der Funktionsgruppe *Arbeitsplanung* dagegen werden die Stücklisten zwar zur Erstellung von Arbeitsplänen benötigt, jedoch haben die Arbeitsplaner nicht die Berechtigung, diese zu verändern. Somit müssen für diesen Bereich die Stücklisten im Nur-Lese-Modus ausgegeben werden.
Neben den unterschiedlichen Informationsaufbereitungen bei verschiedenen Funktionsgruppen, variieren die Aufbereitungsformen auch innerhalb einer Abteilung aufgrund unterschiedlicher Qualifikation und Kenntnis der Mitarbeiter. So hat beispielsweise ein Auszubildender lediglich Zugriff auf Test-Daten zum Erlernen der Funktionen eines Programms, während einem im Fachgebiet ausgebildeten Mitarbeiter Zugriff auf aktuelle Daten gewährt wird.

Eine weitere Aufgabe der adaptiven Informationsaufbereitung besteht darin, unterschiedliche Detaillierungsgrade der Informationen zu berücksichtigen. So werden beispielsweise Konstruktionszeichnungen des gesamten Produkts, einer Baugruppe oder auch nur eines Einzelteils benötigt. Zudem muß die Möglichkeit bestehen, eine Konstruktionszeichnung in einer Gesamtansicht oder einen Ausschnitt auf Pixelebene bearbeiten zu können. Ein weiteres Beispiel für den Detaillierungsgrad der Informationen ist eine Umsatzübersicht über den Zeitraum des letzten Jahres mit den Gesamtsummen der einzelnen Monate, oder einer detaillierte Aufstellung der Umsätze des letzten Monats mit allen Eingängen.

Um eine anwenderzentrierte, adaptive Informationsaufbereitung zu ermöglichen, müssen die Informationsbedürfnisse, Qualifikationen, Kenntnisse und Präferenzen der Benutzer beschrieben werden. Dies erfolgt mit Hilfe eines Benutzermodells. Das Benutzermodell wird bei Änderung von Aufgabenbereich, Qualifikation etc. an den Anwender angepaßt und somit eine benutzergerechte Darstellung der Informationen gewährleistet. Dabei hat der Benutzer die Möglichkeit, die Parameter seiner Präferenzen und gegebenenfalls seines Kenntnisstandes anzupassen. Die Möglichkeit der Änderung der Attribute, die die Funktionsgruppe und die Qualifikation betreffen, liegen bei dem Vorgesetzten oder einem dafür vorgesehenen Administrator.

Um den einzelnen Benutzern die entsprechende Informationsaufbereitung auf Anfrage zukommen zu lassen, wurde im Rahmen diese Diplomarbeit ein Konzept eines Architekturmodell für die adaptive Informationsaufbereitung erarbeitet. Die Benutzer werden in einem Benutzermodell mit Hilfe der Attribute *Funktionsgruppenzugehörigkeit, Qualifikation* und *Kenntnisstand* beschrieben. Die unterschiedlichen Informationsaufbereitungsarten werden von intelligenten Agenten generiert, die aufgrund übergebener Parameter die gewünschte Information aus dem bereichsübergreifenden, betrieblichen Informationssystem filtern und entsprechend aufbereiten. In der Schnittstelle zwischen dem Benutzermodell und den intelligenten Agenten sind die Zuordnungen der unterschiedlichen Informationsaufbereitungsarten zu den entsprechenden Benutzerprofilen abgespeichert. Die Schnittstellenprogramme sind ebenfalls intelligente Agenten, die auf der Basis der Einträge im Benutzermodell entscheiden, welcher Agent zur Informationsbeschaffung und -aufbereitung anzusprechen ist.
Für die Realisierung adaptiver Informationsaufbereitung eignet sich besonders *Hypermedia,* da unterschiedliche Medien, wie Text, Graphik, Audio und Video, unterstützt werden.

Der Aufwand für die Entwicklung eines adaptiven Systems liegt deutlich höher als für weniger flexible Informationssysteme, jedoch wird die Akzeptanz und Effizienz adaptiver Systeme wesentlich gesteigert.

An dieser Stelle möchte ich mich bei meiner Betreuerin Dipl.-Inform. Monika Bihler für ihre sehr engagierte Unterstützung und Betreuung meiner Diplomarbeit bedanken.

Mein weiterer Dank gilt Dipl.-Inform. Markus Stolpmann für seine ergänzenden Kommentare und Anregungen.

Meinem Lebensgefährten Peter Meurisch danke ich dafür, daß er mir stets hilfreich zur Seite stand.

Besonderer Dank gilt auch meinen Eltern, die mir dieses Studium ermöglicht haben.

Anhang

Source-Code der HTML-Dokumente und CGI-Programme der vorgestellten Beispiele in Kapitel 5 “Implementierung“.

baukado.html

```
<HTML>
<HEAD>
<TITLE>Homepage Baukado</TITLE>
</HEAD>
<BODY>
<HR><H1>Homepage Baukado</H1><HR>
<p>
<p>Mitarbeiter:
<p>
<p><a href="http://farfalle.informatik.uni-stuttgart.de/cgi-bin/maile/benutzermodell.pl
        ?name=Bach">Bach </a>
<p><a href="http://farfalle.informatik.uni-stuttgart.de/cgi-bin/maile/benutzermodell.pl
        ?name=Franke">Franke </a>
<p><a href="http://farfalle.informatik.uni-stuttgart.de/cgi-bin/maile/benutzermodell.pl
        ?name=Huber">Huber </a>
<p><a href="http://farfalle.informatik.uni-stuttgart.de/cgi-bin/maile/benutzermodell.pl
        ?name=Vogel">Vogel </a>
<p><a href="http://farfalle.informatik.uni-stuttgart.de/cgi-bin/maile/benutzermodell.pl
        ?name=Weber">Weber </a>
</BODY>
</HTML>
```

---

benutzermodell.pl

```
#!/usr/remote/bin/perl
#Übergabe des vorhandenen Parameters, z.B. Benutzer an die Variable $in
if ($ENV{'REQUEST_METHOD'} eq "GET") {
   $in = $ENV{'QUERY_STRING'};
} elsif ($ENV{'REQUEST_METHOD'} eq "POST") {
        for ($i = 0; $i < $ENV{'CONTENT_LENGTH'};
        $i++) {
        $in .= getc;   }
        }
 # Das Name-Wert-Paar wird zerlegt und kodierte Sonderzeichen werden dekodiert.
($name,$bm_file) = split(/=/,$in);
$bm_file =~ tr/+/ /;
$bm_file =~ s/%([a-fA-F0-9])/pack("C",hex($1))/eg;
# Grossbuchstaben durch Kleinbuchstaben ersetzen
$bm_file =~ tr/A-Z/a-z/;
# Der Benutzername ist Inhalt der Variablen $bm_file.

print <<"Ende";
Content-type:text/html
\n\n
<HTML>
<HEAD>
<TITLE>Benutzermodell</TITLE>
</HEAD>
<BODY>
<HR><H1>Benutzermodell $bm_file</H1><HR>
Ende

&bm_einlesen($bm_file);

print <<"Ende";
Funktionsgruppe: $bm1[0]\n
<p>
Qualifikation  : $bm1[1]\n
<p>
Kenntnisstand  : $bm1[2]\n
```

```
<p>
<p>
<FORM METHOD=POST ACTION="http://farfalle.informatik.uni-stuttgart.de/cgi-bin/maile/homepage.pl">
<INPUT NAME=user TYPE=hidden VALUE="$bm_file">
<INPUT TYPE=submit VALUE="Weiter">
</FORM>
</BODY>
</HTML>
\n\n
Ende

sub bm_einlesen {
        $i=0;
        open (BM,"/lasagne/users/maile/baukado/bm/$bm_file");
        while($attribut = <BM>){
                chop($attribut);
                $bm1[$i] = $attribut;
                $i++;
                }
        close (BM);
        }
```

homepage.pl

```
#!/usr/remote/bin/perl
#Übergabe des vorhandenen Parameters, z.B. Benutzer an die Variable $in
if ($ENV{'REQUEST_METHOD'} eq "GET") {
   $in = $ENV{'QUERY_STRING'};
} elsif ($ENV{'REQUEST_METHOD'} eq "POST") {
        for ($i = 0; $i < $ENV{'CONTENT_LENGTH'};
        $i++) {
        $in .= getc;   }
        }
```

```
# Das Name-Wert-Paar wird zerlegt und kodierte Sonderzeichen werden dekodiert.
($user,$bm_file) = split(/=/,$in);
$bm_file =~ tr/+/ /;
$bm_file =~ s/%([a-fA-F0-9])/pack("C",hex($1))/eg;
# Grossbuchstaben durch Kleinbuchstaben ersetzen
$bm_file =~ tr/A-Z/a-z/;
# Der Benutzername ist Inhalt der Variablen $bm_file.

print <<"Ende";
Content-type:text/html
\n\n
<HTML>
<HEAD>
<TITLE>Homepage</TITLE>
</HEAD>
<BODY>
Ende

&bm_einlesen($bm_file);

print "<HR><H1>Homepage $bm[0]</H1><HR>";
print "<p>";

if ($bm[0] eq "Konstruktion"){
print <<"Ende";
<UL>
<LI> <a href='http://farfalle.informatik.uni-stuttgart.de/cgi-bin/maile/
        formular_zeichnung.pl?name=$bm_file'>Konstruktionszeichnung</a>
<LI> <a href='http://farfalle.informatik.uni-stuttgart.de/cgi-bin/maile/
        formular_stueckliste.pl?name="$bm_file"'>St&uuml;ckliste</a>
</UL>
Ende
}

print <<"Ende";
</BODY>
</HTML>
\n\n
Ende
```

```
sub bm_einlesen {
        $i=0;
        open (BM,"/lasagne/users/maile/baukado/bm/$bm_file");
        while($attribut = <BM>){
                chop($attribut);
                $bm[$i] = $attribut;
                $i++;
                }
        close (BM);
        }
```

formular_zeichnung.pl

```
#!/usr/remote/bin/perl
#Übergabe des vorhandenen Parameters, z.B. Benutzer an die Variable $in
if ($ENV{'REQUEST_METHOD'} eq "GET") {
   $in = $ENV{'QUERY_STRING'};
} elsif ($ENV{'REQUEST_METHOD'} eq "POST") {
        for ($i = 0; $i < $ENV{'CONTENT_LENGTH'};
        $i++) {
        $in .= getc;   }
        }
# Das Name-Wert-Paar wird zerlegt und kodierte Sonderzeichen werden dekodiert.
($name,$bm_file) = split(/=/,$in);
$bm_file =~ tr/+/ /;
$bm_file =~ s/%([a-fA-F0-9])/pack("C",hex($1))/eg;
# Grossbuchstaben durch Kleinbuchstaben ersetzen
$bm_file =~ tr/A-Z/a-z/;
# Der Benutzername ist Inhalt der Variablen $bm_file.

print <<"Ende";
Content-type:text/html
\n\n
<HTML>
```

```
<HEAD>
<TITLE>Formular Zeichnung</TITLE>
</HEAD>
<BODY>
<HR><H1>Formular Konstruktionszeichnung</H1><HR>
<p>
Ende

&bm_einlesen($bm_file);

print <<"Ende";
<p>
<FORM METHOD=POST ACTION="http://farfalle.informatik.uni-stuttgart.de/cgi-
bin/maile/zeichnung.pl">
<INPUT NAME=user TYPE=hidden VALUE="$bm_file">
<p>Bitte geben Sie den Namen des gew&uuml;nschten Teils an
<p>
<INPUT NAME=teil SIZE="30" MAXLENGTH="30">
<INPUT TYPE=submit VALUE="Weiter">
<INPUT TYPE=reset>
</FORM>
Ende

print "</BODY>";
print "</HTML>";
print "\n\n";

sub bm_einlesen {
        $i=0;
        open (BM,"/lasagne/users/maile/baukado/bm/bach");
        while($attribut = <BM>){
                chop($attribut);
                $bm1[$i] = $attribut;
                $i++;
                }
        close (BM);
        }
```

zeichnung.pl

```
#!/usr/remote/bin/perl
#Übergabe der vorhandenen Parameter, z.B.
#Benutzername und Teilename an die Variable $in
if ($ENV{'REQUEST_METHOD'} eq "GET") {
   $in = $ENV{'QUERY_STRING'};
} elsif ($ENV{'REQUEST_METHOD'} eq "POST") {
        for ($i = 0; $i < $ENV{'CONTENT_LENGTH'};
        $i++) {
        $in .= getc;    }
        }
# Die Name-Wert-Paare werden getrennt und in das Array @paare geschrieben
@paare = split(/&/,$in);
# Die Name-Werte-Paare werden zerlegt und kodierte Sonderzeichen werden dekodiert.
foreach $paare (@paare){
        ($name,$inhalt) = split(/=/,$paare);
        $inhalt =~ tr/+/ /;
        $inhalt =~ s/%([a-fA-F0-9])/pack("C",hex($1))/eg;
        $inhalt =~ tr/A-Z/a-z/;
        $felder{$name} = $inhalt;
}
$teil=$felder{teil};
$bm_file=$felder{user};
#Der Benutzername ist Inhalt der Variablen $bm_file.Der Teilename ist Inhalt der
#Variablen $teile.

&bm_einlesen($bm_file);

if ($bm[1] eq "sehr gut"){
        if ($bm[2] eq "Experte"){
                print "Location:/maile/zeichnung-$teil.html";
        }
}
print "\n\n";
```

```
sub bm_einlesen {
        $i=0;
        open (BM,"/lasagne/users/maile/baukado/bm/$bm_file");
        while($name = <BM>){
        chop($name);
        $bm[$i]=$name;
        $i++;
        }
close (BM);
}
```

zeichnung-quader1.html

```
<HTML>
<HEAD>
<TITLE>Konstruktionszeichnung Quader 1</TITLE>
</HEAD>
<BODY>
<HR><H1>Konstruktionszeichnung Quader 1 </H1><HR>
<p><img align=middle src="file://localhost/usr/local/etc/httpd/cgi-bin/maile/graphik/
        quader1.gif">
</BODY>
</HTML>
```

Literaturverzeichnis

[Agents FAQ 1996]

"The Software Agents Mailing List FAQ", 1996, http://www.ee.mcgill.ca/~belmarc/agent_faq.html

[Benyon 1994]

D. Benyon: *"Adaptive Systems: A Solution to Usability Problems"*, in *"User Modeling and User-Adapted Interaction 3: An International Journal"*, S. 65 - 87, Kluwer Academic Publishers, Dordrecht, 1994

[Bihler et al. 1996]

M. Bihler, D. Roller: *"Adaptive Information Processing for Technical Applications"*, Paper number 96ME052, Universität Stuttgart, 1996

[Böcker et al. 1993]

D. Böcker, W. Glatthaar, T. Strothotte (Hrsg.): *"Mensch-Computer-Kommunikation: Benutzergerechte Systeme auf dem Weg in die Praxis,* [unserem Lehrer Rul Gunzenhäuser zum 60. Geburtstag gewidmet], Springer-Verlag, Berlin, Heidelberg, New York, London, Paris, Tokyo, Hong Kong, Barcelona, Budapest, 1993

[Boyle et al. 1995]

C. Boyle, A. O. Encarnacion: *"Metadoc: An Adaptive Hypertext Reading System"*, in *"User Modeling and User-Adapted Interaction 4: An International Journal"*, S. 1 - 19, Kluwer Academic Publishers, Dordrecht, 1995,

[CACM Juli 1994]

Communications of the ACM: *"Intelligent Agents"*, Juli 1994

[Duden Informatik 1993]

"Duden Informatik", Dudenverlag, Mannheim, Leipzig, Wien, Zürich, 1993

[Frühwacht 1995]

S. Frühwacht: *"Nutzung von Shell-Skripten aus Mosaic, adaptive Informationsaufbereitung über Zugriffsberechtigung"*, Studienarbeit, Universität Stuttgart, 1995

[Genesereth et al. 1994]

M. R. Genesereth, S. P. Ketchpel: *"Software Agents"*, in [CACM Juli 1994], S. 48 - 53

[Girard et al. 1992]

J. Girard, G. Gauthier, S. Levesque: *"Une architecture multiagent"*, in C. Frasson, G. Gauthier, G. I. McCalla (Hrsg.): *"Intelligent Tutoring Systems"*, S. 172 - 182, Springer-Verlag, Berlin, Heidelberg, 1992

[Haaks 1992]

D. Haaks: *"Anpaßbare Informationssysteme - Auf dem Weg zu aufgaben- und benutzerorientierter Systemgestaltung und Funktionalität"*, Verlag für angewandte Psychologie, Göttingen, Stuttgart, 1992

[Herrmann 1996]

E. Herrmann: *"Teach yourself CGI Programming with Perl in a week"*, Sams.net Publishing, 1996

[Hohl 1995]

H. Hohl: *"Entwurf und Einsatz wissensbasierter Werkzeuge zur computergestützten Exploration von Informationsräumen"*, Dissertation, Universität Stuttgart, 1995

[Hughes 1993]

K. Hughes: "*Entering the World-Wide-Web: A Guide to Cyberspace*", Honolulu Community College, 1993

[Kobsa 1994]

A. Kobsa: "*Adaptivität und Benutzermodellierung in interaktiven Softwaresystemen*", Bericht Nr. 64/94 (WIS-Bericht 8), Universität Konstanz, 1994

[Kuhlen 1991]

R. Kuhlen: "*Hypertext: Ein nichtlineares Medium zwischen Buch und Wissensbank*", Springer-Verlag, Berlin, Heidelberg, New York, London, Paris,Tokyo, Hong-Kong, Barcelona, Budapest, 1991 (Edition SEL-Stiftung)

[Maes 1994]

P. Maes: "*Agents that Reduce Work and Information Overload*", in [CACM Juli 1994], S. 30 - 40

[Mayer 1996]

E. A. Mayer: "*Introduction to HTML*", Case Western Reserve University, 1996, http://www.cwru.edu/help/introHTML/toc.html

[Mertens et al. 1992]

P. Mertens, F. Bodendorf, W. König, A. Picot, M. Schumann: "*Grundzüge der Wirtschaftsinformatik*", Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, 1992

[Meyerhoff 1994]

D. B. Meyerhoff: "*Hypertext und tutorielle Lernumgebungen: Ein Ansatz zur Integration*", Oldenbourg Verlag, München, Wien, 1994

[Minsky et al. 1994]

M. Minsky, D. Riecken: *"A Conversation with Marvin Minsky About Agents"*, in [CACM Juli 1994], S. 22 - 29

[Parsaye et al. 1989]

K. Parsaye, M. Chignell, S. Khoshafian, H. Wong: *"Intelligent Databases - Object-Oriented, deductive Hypermedia Technologies"*, John Wilea & Sons, Inc., 1989

[Reiter et al. 1993]

D. Reiter, D. Roller: *"Erschließung von Information und Wissen: Verfahren zur dynamischen Aggregation von Hypertrails"*, in H. P. Frei, P. Schäuble (Hrsg.): *"Hypermedia"*, S. 81 - 92, Springer Verlag, Berlin, Heidelberg, 1993

[Roller et al. 1995]

D. Roller, M. Bihler, M. Stolpmann: *"Adaptive hypermediale Informationsaufbereitung in betrieblichen Informationssystemen"*, in R. Kuhlen, M. Rittenberger (Hrsg.): *"Hypertext - Information Retrieval - Multimedia, Synergieeffekte elektronischer Informationssysteme"*, S. 319 - 331, Universitätsverlag Konstanz UVK, 1995

[Schoop et al. 1992]

E. Schoop, U. Glowalla: *"Computer in der Aus- und Weiterbildung: Potentiale, Probleme und Perspektiven"*, in U. Glowalla, E. Schoop (Hrsg.): *"Hypertext und Multimedia - Neue Wege in der computerunterstützten Aus- und Weiterbildung"*, S. 4 - 20, Springer-Verlag, Berlin, Heidelberg, 1992,

[Schwab 1989]

T. Schwab: *"Methoden zur Dialog- und Benutzermodellierung in adaptiven Computersystemen"*, Dissertation, Universität Stuttgart, 1989

[Schwartz 1993]

R. L. Schwartz: *"Learning Perl"*, O'Reilly & Associates, Inc., 1993

[Schwarze 1994]

J. Schwarze: *"Einführung in die Wirtschaftsinformatik"*, Verlag Neue Wirtschafts-Briefe, Herne, Berlin, 1994

[Steinbuch 1991]

P. A. Steinbuch: *"Betriebliche Informatik"*, Friedrich Kiehl Verlag, Ludwigshafen (Rhein), 1991 (Kompendium der praktischen Betriebswirtschaft)

[Sundermeyer et al. 1993]

K. Sundermeyer, S. Albayrak: *"Workshop: Kommunikation, Koordination und Kooperation in Mehragenten Systemen"*, in O. Herzog, Th. Christaller, D. Schütt (Hrsg.): *"Grundlagen und Anwendungen der Künstlichen Intelligenz"*, S. 242 - 247, Springer-Verlag, Berlin, Heidelberg, 1993

[Wilson 1995]

T. Wilson: *"Modelling the information user: the wider perspective"*, Department of Information Studies, University of Sheffield, U.K., November 1995, http://www.shef.ac.uk/academic/I-M/is/lecturer/klpaper.html