Reihe: Telekommunikation und Mediendienste · Band 2

Herausgegeben von Norbert Szyperski, Udo Winand, Dietrich Seibt und Rainer Kuhlen

Martin Engelien/Kai Bender (Hrsg.)

GeNeMe98

Gemeinschaften in Neuen Medien

TU Dresden, 1./2.10.1998

JOSEF EUL VERLAG

Lohmar · Köln

Reihe: Telekommunikation und Mediendienste · Band 2

Herausgegeben von Prof. Dr. Dr. h. c. Norbert Szyperski, Köln, Prof. Dr. Udo Winand, Kassel, Prof. Dr. Dietrich Seibt, Köln, und Prof. Dr. Rainer Kuhlen, Konstanz

Doz. Dr.-Ing. habil. Martin Engelien
Dipl.-Inf. (FH) Kai Bender (Hrsg.)

GeNeMe98

Gemeinschaften in Neuen Medien

TU Dresden, 1./2.10.1998

JOSEF EUL VERLAG
Lohmar · Köln

Die Deutsche Bibliothek – CIP-Einheitsaufnahme

GeNeMe <1998, Dresden>:
GeNeMe 98 : Gemeinschaften in neuen Medien / Technische Universität Dresden, Fakultät Informatik, Institut für Informationssysteme, Dozentur „Entwurfmethoden und Werkzeuge für Anwendugssysteme". Martin Engelien; Kai Bender (Hrsg.). – Lohmar ; Köln : Eul, 1998.
(Reihe: Telekommunikation und Mediendienste ; Bd. 2)
ISBN 3-89012-632-4

Josef Eul Verlag GmbH
Brandsberg 6
53797 Lohmar
Tel.: 0 22 05 / 91 08 91
Fax: 0 22 05 / 91 08 92
e-mail: eul.verlag.gmbh@t-online.de

Printed in Germany
Druck: Rosch-Buch, Scheßlitz

Gedruckt auf säurefreiem und 100% chlorfrei gebleichtem Papier

Technische Universität Dresden
Fakultät Informatik • Institut für Informationssysteme
Dozentur „Entwurfsmethoden und Werkzeuge für Anwendungssysteme"

Doz. Dr.-Ing. habil. Martin Engelien
Dipl.-Inf. (FH) Kai Bender
(Hrsg.)

Dresden, 1./2.10.1998

GeNeMe98

Gemeinschaften in Neuen Medien

Workshop zu Organisation, Kooperation und Kommunikation auf der Basis innovativer Technologien

Forum für den Dialog zwischen Wissenschaft und Praxis zur Inversion der Virtualität (Ubiquitous Computing)

unter der Schirmherrschaft von:

Dr. W. Vehse
Staatssekretär für Wirtschaft
des Landes Sachsen

Prof. Dr. A. Mehlhorn
Rektor der TU Dresden

sowie unter Mitwirkung der
GI-Regionalgruppe Dresden

und mit freundlicher Unterstützung folgender Partner:

IST priv. Institut für angewandte Software-Technologie GmbH, Dresden
eine Ausgründung der TU Dresden auf dem Gebiet der Technologien und Anwendungen in den Neuen Medien

Heyde AG,
Bad Nauheim/ Dresden
Beratung • Software • Integration

E.2. Ein Rahmenwerk für kooperativen Informationsaustausch

Dipl.-Inf. A. Behle
RWTH Aachen

Abstract

In der Region Aachen wollen eine Reihe von Softwareherstellern durch Austausch von Softwarekomponenten und durch Austausch von Erfahrungen beim Einsatz von kommerziellen Softwarekomponenten kooperieren. In Zusammenarbeit mit diesen Partnern aus dem Regionalen Industrieclub Informatik Aachen wurde daher ein Internet-basiertes Informationssystem für kooperative Softwarewiederverwendung entwickelt, das wir in diesem Beitrag vorstellen. Wir gehen auf die Aspekte der Kooperation, den Klassifikationsansatz und das zugrundeliegende SYNERGIE-Rahmenwerk ein und erläutern, warum mit diesem Rahmenwerk auch andere Informationssysteme zur Unterstützung unternehmensübergreifender Kooperation entwickelt werden können.

1 Einleitung

Im Internet aber auch in den firmenspezifischen Intranets wird eine große Menge an mehr oder weniger interessanten Informationen bereitgestellt. Das Problem besteht für Benutzer dieser Systeme darin, die für sie relevanten Informationen zu finden. Suchmaschinen bieten die Möglichkeit, nach beliebigen Begriffen im Intra- oder Internet zu suchen und relevante Ergebnisse zu finden. Sind die Begriffe jedoch relativ allgemein, können mehrere tausend Verweise als Ergebnis geliefert werden, deren Verfolgung und Überprüfung manuell nicht effizient möglich ist. Diese Vorgehensweise ist zudem wenig geeignet, wenn möglichst alle Informationen zu einem Themengebiet, beispielsweise zu einer Produktart, gesucht werden, um so einen Vergleich dieser Produkte hinsichtlich ihrer Merkmale anzustellen. Noch schwieriger ist es, auf diese Art eine objektive Beurteilung der Produkte zu finden.

Zur Ermittlung von Informationen, welche der *Entscheidungsunterstützung*, beispielsweise einer Kaufentscheidung, dienen sollen, werden also Dienste in Form von *themen- oder produktspezifischen Informationssystemen* benötigt, welche möglichst objektiv und umfassend über vorhandene Produkte einer Sparte oder spezielle Themen informieren. Wird durch das System weiterhin eine Beurteilung der Produkte ermöglicht, indem ein Erfahrungsaustausch zwischen interessierten Parteien unterstützt wird, so kann dies neben Informationen zu den unterstützten Merkmalen auch Hinweise

auf die Qualität der Produkte liefern. Neben Art und Umfang der geeigneten Informationen müssen auch die Möglichkeiten zu deren Auffinden den Ansprüchen der Anwender genügen und den Zugriff auf unterschiedliche Arten, abhängig von dem Wissen über die gesuchten Produkte oder Themen, ermöglichen.

Im Rahmen des *RSB-Projektes* (REGINA Software-Bibliothek) wurde ein *Internet-basiertes Informationssystem* für *kooperative Softwarewiederverwendung*, das Komponenteninformationssystem (KIS), entwickelt [Beh98]. Dieses System befindet sich im Internet zur Zeit unter http://www.findcomponents.com in der Testphase und ist frei zugänglich. Projektteilnehmer sind neben dem Lehrstuhl für Informatik III neun Softwarehersteller, die Mitglieder im Regionalen Industrieclub Informatik Aachen e.V. (REGINA) sind. Ziel des Vereins ist die Unterstützung des *Austauschs von Know-how* und das Erzielen von *Synergieeffekten* durch die *Förderung der Kooperation* zwischen Softwareherstellern in der Aachener Region. Durch das RSB-Projekt, das 1995 begann und Ende 1998 ausläuft, soll diese Kooperation im Bereich der Softwarewiederverwendung unterstützt werden.

Bei dem Informationssystem handelt es sich nicht um eine Ansammlung von statischen, unstrukturierten HTML-Seiten zur Beschreibung von Softwarekomponenten, sondern um ein *interaktives, erweiterbares System*, welches *strukturierte* Informationen anbietet und die bei den Industriepartnern vorhandene Infrastruktur nutzt.

In diesem Beitrag werden zunächst die Möglichkeiten zur Kooperation vorgestellt, die durch unseren Ansatz unterstützt werden. Anschließend erläutern wir den verwendeten Klassifikationsansatz und das darauf aufbauende Rahmenwerk. Dabei stellen wir einige Überlegungen zum Client/Server-Ansatz mit Java vor. Nachfolgend gehen wir auf verwandte Arbeiten ein und geben abschließend eine Zusammenfassung und einen Ausblick.

2 Kooperation

Eine Kooperation innerhalb des Konsortiums (Projekt- und REGINA-Mitglieder) kann im KIS auf *drei Ebenen* stattfinden. Zunächst werden über das Informationssystem zwischen den Partnern wiederverwendbare Softwarekomponenten *ausgetauscht*. Durch entsprechende Referenzeinträge findet ein *Austausch von Erfahrungen* hinsichtlich des Einsatzes von Komponenten statt. Schließlich kann durch Angabe von Anprechpartnern innerhalb des Konsortiums die Grundlage für persönliche Kontakte gelegt und der *Transfer von Know-how* hinsichtlich der Nutzung von Komponenten eingeleitet werden. Das Informationssystem liefert dabei Informationen sowohl über *wiederverwendbare Komponenten*, die von den Projektpartnern angeboten werden, als auch über kommerziell oder frei verfügbare Softwarebausteine.

Die *virtuelle Gemeinschaft* kommuniziert über einen *zentralen Web-Server*, der (später) nur berechtigten Personen Zugang erlaubt.

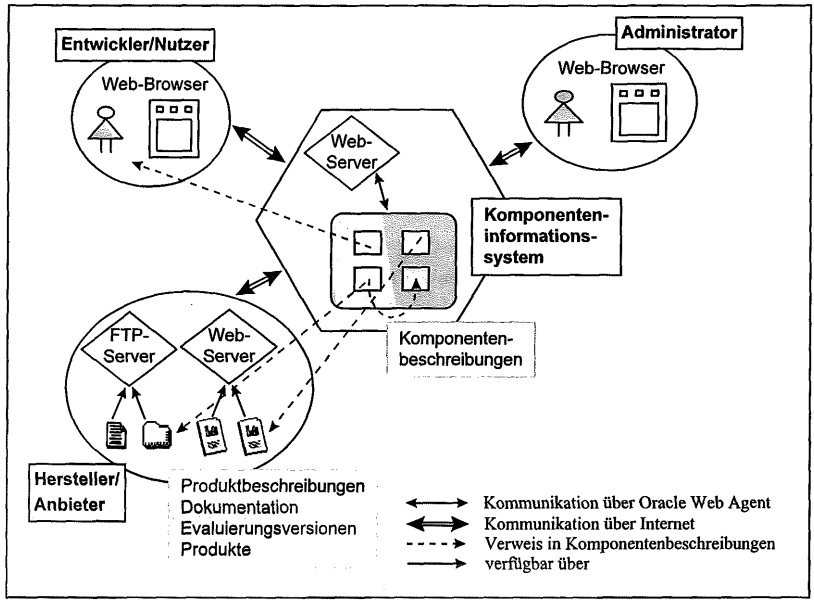

Abbildung 1: Kooperationsszenario im Internet

Abbildung 1 zeigt eine vereinfachte Darstellung des Kooperationsszenarios, in welches das KIS eingebettet ist. Wir können drei Rollen der Benutzer identifizieren: den Hersteller, den Entwickler und den Systemadministrator.

1. Der *Hersteller* bietet eine wiederverwendbare Softwarekomponente zur Nutzung durch Entwickler an. Es kann sich dabei einerseits um ein Unternehmen innerhalb des Konsortiums handeln, welches anderen Mitgliedern seine Komponenten zu besonderen Konditionen zur Verfügung stellt. Andererseits können auch Hersteller kommerzieller oder frei verfügbarer Produkte diese anbieten. Viele Hersteller von Softwarekomponenten besitzen heute einen Anschluß an das Internet und bieten beispielsweise über einen FTP-Server oder einen Web-Server Produktbeschreibung, Dokumentation, Evaluierungsversion oder das Produkt selber an.
2. Die *Entwickler* in Softwareunternehmen sind die eigentliche Zielgruppe des Systems. Sie können nach bestimmten wiederverwendbaren Produkten suchen, die bei der Erstellung eines eigenen neuen Produkts eingesetzt werden können, und ihre Erfahrungen darüber austauschen.

3. Der *Administrator* schließlich ist für die Pflege des Klassifikationsschemas (s.u.) und der Benutzerdaten sowie für die Pflege und Kontrolle aller anderen Produktdaten zuständig.

Über das KIS sollen nun einerseits Hersteller und Entwickler, andererseits Entwickler untereinander zusammengeführt werden. Wichtig ist dabei, daß die bei Entwicklern und Herstellern *vorhandene Infrastruktur genutzt* wird und keine neuen Softwareinstallationen erforderlich werden. Alle Benutzer des Systems benötigen lediglich Web-Browser, die heute bei fast allen Firmen installiert sind. Falls vorhanden, können Hersteller die bei Ihnen vorhandenen Internet-Dienste weiter nutzen, um umfangreiche Dokumente im Internet anzubieten. Über den Web-Browser können sie ihre Produkte im KIS registrieren und auf vorhandene Internet-Quellen durch entsprechende Referenzen in den Komponentenbeschreibungen hinweisen. Auch Beziehungen zwischen mehreren eingetragenen Komponenten können dargestellt werden. Der Schwerpunkt des KIS liegt in der *Bereitstellung von Informationen* für Entwickler, also die Endnutzer der Produkte, sowie der *Unterstützung der Kooperation* zwischen ihnen in den oben genannten Aspekten.

Im Gegensatz zu anderen Ansätzen zur Unterstützung der Wiederverwendung werden hier Informationen zu Komponenten bereitgestellt, nicht die Komponenten selbst. Dazu enthält das KIS neben *umfangreichen Beschreibungen* von wiederverwendbaren Softwarekomponenten, die vom Administrator oder den Herstellern eingegeben werden können, auch die *Ansprechpartner* zu Komponenten in der Region und *Erfahrungen* der Entwickler mit dem Einsatz der Komponenten. Die Komponenten selbst werden bei Anbietern lokal gepflegt und der (kostenpflichtige) Bezug wird ggf. über bei den Firmen vorhandene Internet-Dienste ermöglicht. So können Komponenten ausgetauscht werden, obwohl sie nicht direkt im KIS enthalten sind.

Die Beschreibungen der Softwarekomponenten werden jedoch nicht in unstrukturierter Form als Volltext gespeichert. Hierzu haben wir einen Klassifikationsansatz entwickelt.

3 Der Klassifikationsansatz

Die Vergleichbarkeit der Informationen ist eine wesentliche Voraussetzung für ein attraktives Informationssystem und eine bestmögliche Unterstützung bei Entscheidungen. Die Darstellung eines unstrukturierten Beschreibungstextes ist dazu nicht geeignet, denn bei den Beschreibungsdaten handelt es sich in der Regel um eine große Menge unterschiedlichster Informationen. Statt dessen werden im KIS strukturelle Informationen durch einen *Softwareklassifikationsansatz* bereitgestellt. Dieser dient als Grundlage sowohl für die Datenspeicherung als auch für die einheitliche Erstellung der Produktbeschreibungen. Wir nutzen einen *verallgemeinerten*

Klassifikationsansatz, auf welchen sich andere (facettierte, enumerierte) Klassifikationsansätze abbilden lassen.

Wir *kombinieren* so die *Vorteile* der Facettenklassifikation [Buc89] mit den Vorteilen einer hierarchischen Beschreibungsstruktur, welche im Gegensatz zu einer flachen Struktur Klarheit in die Zusammenhänge zwischen den Facetten und Merkmalen bringen kann. Die *hierarchische Struktur* des Klassifikationsschemas wird zusätzlich als Grundlage für die *Navigation* in dem Datenbestand des KIS genutzt.

Aus Gründen der *Änderungsfreundlichkeit* haben wir den Ansatz eines *Metasschemas* gewählt. Die Ursache für diese Vorgehensweise ist die Erkenntnis, daß gerade ein Schema zur Beschreibung von Softwarekomponenten häufigen Änderungen unterworfen ist, da ständig neue Anwendungsgebiete, Programmiersprachen und Plattformen hinzukommen und das Wissen über bereits bekannte Anwendungsgebiete permanent erweitert wird. Diese Erkenntnis wird bestätigt, wenn man die Entwicklung beispielsweise des bekannten *ACM Computing Classification Systems* betrachtet. Unsere Schlußfolgerung war, daß das System so entwickelt werden muß, daß das Klassifikationsschema angepaßt, erweitert und reorganisiert werden kann, ohne bestehende Implementierungen manuell ändern zu müssen. Zusätzlich muß die Konsistenz der Daten zu jedem Zeitpunkt gesichert sein.

Das Metaschema beschreibt das Klassifikationsschema und die Informationen zu Softwarekomponenten, die durch das Klassifikationsschema beschrieben werden können, und ist außerdem Basis für die verschiedenen Mechanismen zur Navigation und Suche (Volltextsuche, gefilterte Suche, Spezifikation der geforderten Merkmale) sowie den Möglichkeiten zum Wechsel zwischen Suche und Navigation. Dieses Metaschema stellt die Basis des Datenbankschemas und des im folgenden Abschnitts dargestellten Rahmenwerks dar. So können sowohl das Klassifikationsschema als auch die korrespondierenden Komponentendaten in der Datenbank gespeichert werden. Einzelne Klassifikationsschemaeinträge können geändert und das gesamte Schema kann leicht erweitert oder reorganisiert werden, ohne daß Daten verlorengehen. Dies wird durch ein Werkzeug garantiert, das in nicht eindeutigen Fällen auf Interaktion mit dem Administrator zurückgreift. Unsere Untersuchungen und praktischen Tests haben gezeigt, daß nur bei wenigen Schemaänderungen eine Interaktion mit dem Administrator des Systems notwendig ist.

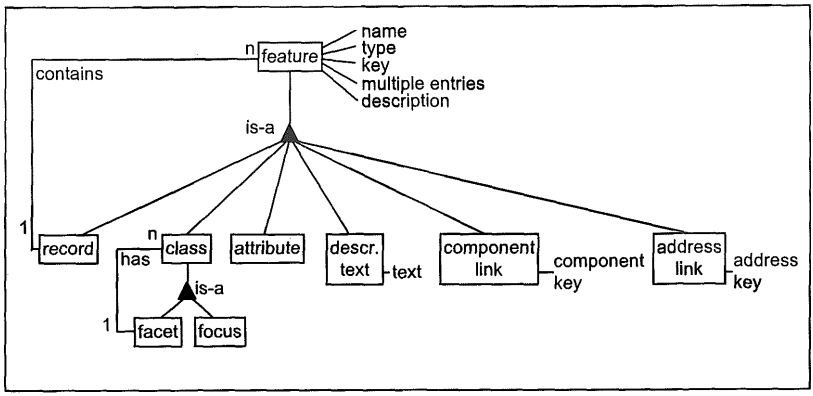

Abbildung 2: ER-Diagramm des Metaschemas

Unser Klassifikationsansatz erlaubt nun die verschachtelte Kombination von Datensätzen und Facetten, so daß für jedes Merkmal die geeignete Beschreibungsart gewählt werden kann. Beliebige Arten von Beziehungen können durch gerichtete Verweise zwischen Komponenten und Adreßeinträgen oder zwischen zwei Komponenten eingerichtet werden.

Das in Abbildung 2 dargestellte Metaschema beschreibt die generellen Konzepte, die zur Erstellung eines Klassifikationsschemas genutzt werden können. Ein *Merkmal* (feature) ist ein Term, der eine Softwarekomponente beschreibt. Es wird dargestellt durch seinen Bezeichner, einen eindeutigen Schlüssel und Informationen zu dem Typ des Merkmals. Ferner kann durch eine Flagge angegeben werden ob mehrere Einträge dieses Merkmals zugelassen sind. Die Beschreibung (description) wird in einer Online-Hilfe zur Erläuterung des Merkmals genutzt.

Die allgemeinen Konzepte zur Schaffung der Hierarchie (1-2) und zur Definition einfacher Merkmale (3-7) sind:

1. *Verbund* (Record): Dieses Merkmal wird durch eine Menge von Merkmalen beschrieben. Diese Menge ist nicht leer.
2. *Facette* (Facet): Alle zu beschreibenden Objekte können bestimmten Klassen zugeordnet werden. Eine Facette beschreibt einen wesentlichen Aspekt, den mehrere Klassen gemeinsam haben. Sie wird durch mindestens einen Fokus oder eine Subfacette beschrieben. Subfacetten sind Facetten, die anderen Facetten untergeordnet sind (Beispiel: Facette `application domain` mit Subfacette `graphical user interface (gui)`).
3. *Fokus* (Focus): Ein Fokus ist die Bezeichnung einer Klasse, die einer Facette zugeordnet ist. Eine Facette besteht aus all den Fokussen, die Gegenstände derselben Art repräsentieren [Buc89] (Beispiel: Windows `3.11`).

4. *Komponentenverknüpfung* (Component link): Dieses Merkmal stellt eine gerichtete Beziehung einer Softwarekomponente zu einer (anderen) Softwarekomponenten dar (Beispiele: `siehe-auch`, `besteht-aus`, `ist-Spezialisierung-von`).
5. *Adreßverknüpfung* (Address link): Dieses Merkmal stellt eine gerichtete Beziehung einer Softwarekomponente zu einem Adreßeintrag dar (Beispiele: `produced-by`, `distributed-by`)
6. *Beschreibender Text* (Descriptive text): Dieses Merkmal wird durch einen beliebigen Text dargestellt (Beispiele: `Identifier`, `WWW links`).
7. *Attribut*: Dieses Merkmal stellt eine spezielle Eigenschaft dar, die eine Komponente besitzen kann oder nicht. (Beispiel: ist eine `class library`).

Abbildung 3 zeigt einen kleinen und unvollständigen Ausschnitt des Klassifikationsschemas, welches zur Zeit bereits mehr als 250 Einträge enthält. Man erkennt, daß eine Komponentenbeschreibung durch einen Verbund von Merkmalen dargestellt wird. Dieser Verbund besteht in diesem Ausschnitt aus Informationen zu dem Bezeichner (identifier), den Plattformen (platform), der Beschreibung (description), dem Anwendungebiet (application domain) und den Bezugsquellen (supply sources). Hinzu kommen Verweise ins WWW (WWW links) und Verweise auf andere Komponenten (see also). Der Bezeichner wird durch einen Text dargestellt. Das Merkmal `Platform` ist eine Facette, die durch den Fokus `DOS` und die Subfacetten `Windows` und `Unix` beschrieben wird. `Windows` wird beschrieben durch die Subfacette `NT` und die Fokusse `3.1`, `3.11` und `95`. Da mehrere Windows-Plattformen unterstützt werden können, zeigt das `*`-Symbol an, daß dort mehrere Einträge erlaubt sind. Die Beschreibung besteht hier aus der Kurzbeschreibung als Text und einem Attribut, das angibt, ob es sich bei einer Komponente um eine Klassenbibliothek handelt. Die Facette `application domain` wird durch verschiedene Subfacetten dargestellt. Die grafischen Benutzeroberflächen (`gui`) werden dabei durch die Subfacetten `control elements` und `surfaces` beschrieben. Schließlich werden die Bezugsquellen durch Verweise auf Adressen zu Herstellern und Anbietern und einem Attribut, ob es sich um eine freie (public domain) Komponente handelt, beschrieben. Die Verweise auf Quellen im World Wide Web werden in einem Text gespeichert, der automatisch ausgewertet und in Hyperlinks umgewandelt werden kann. Die gestrichelt umrandeten Merkmale müssen weiter definiert werden.

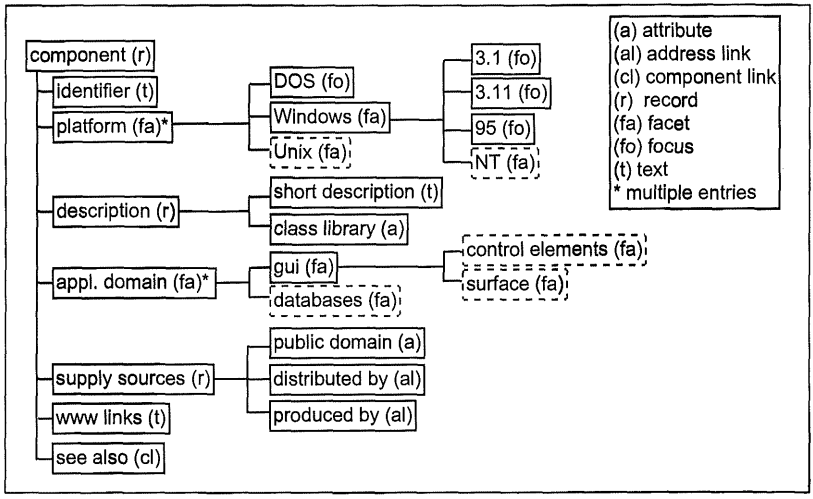

Abbildung 3: Ein Auszug aus dem Klassifikationsschema

Wichtig ist, daß eine dem Klassifikationsschema entsprechende Beschreibung *unvollständig* sein kann. Keine der Angaben (bis auf den Bezeichner) ist obligatorisch, jedoch sollten so viele Angaben wie möglich gemacht werden, um den Benutzern die Beurteilung der Komponente zu ermöglichen.

4 Das *SYNERGIE*-Rahmenwerk

Ein Rahmenwerk ist eine allgemeine Architektur für eine *spezielle* Klasse von *Anwendungssystemen*. Es besteht aus *unspezifischen* (Standard-) Komponenten, die für alle Anwendungen wiederverwendet werden können und *spezifischen* Komponenten, die ausgetauscht und in das Rahmenwerk je nach Bedarf passend eingefügt werden können. Sowohl der Entwurf als auch die Implementierung sind dabei nicht auf objektorientierte Ansätze beschränkt. Damit ist der Begriff des Rahmenwerks nicht mit den objektorientierten Frameworks zu verwechseln, obwohl einige Gemeinsamkeiten bestehen.

An unserem Lehrstuhl wurde in den letzten Jahren ein Rahmenwerk für integrierte Softwareentwicklungsumgebungen (Integrated Software Project Support Environment, IPSEN) entwickelt, das in [Nag96] ausführlich vorgestellt wird. Das IPSEN-Rahmenwerk besteht aus spezifischen und unspezifischen Komponenten, sowie Generatoren, die die spezifischen Teile auf der Basis einer erweiterten EBNF für AST-basierte Sprachen in das Rahmenwerk „hineingenerieren". Es basiert auf GRAS [KSW95], einem graphorientierten Datenbanksystem für (Software-)

Ingenieuranwendungen. Aus verschiedenen Gründen, insbesondere weil im Rahmen des RSB-Projektes die Übernahme der Konzepte und Implementierungen in kommerzielle Entwicklungskontexte zu gewährleisten war, sollte ein kommerzielles Entwicklungs- und Datenbanksystem eingesetzt werden. Da das IPSEN-Rahmenwerk und auch das damit entwickelte System FOCS (Feature-Oriented Classification System) [Bör95] nicht für kommerziellen Einsatz geeignet sind und auch deren Anschluß an das World Wide Web nicht vorgesehen ist, war es im RSB-Projekt nicht möglich, darauf zurückzugreifen.

Die Erfahrungen jedoch, die im Kontext von IPSEN gesammelt wurden, konnten in den Entwurf und die Realisierung des im Vergleich zu IPSEN kleinen SYNERGIE (*System für Internet-basierten gemeinsamen Aufbau von Informationen und Erfahrungen*)-Rahmenwerks einfließen. Die für die Architektur des SYNERGIE-Rahmenwerks relevanten Ergebnisse aus [Kle96] wollen wir nun zunächst kurz zusammenfassen, bevor wir seine Architektur und die durch Werkzeuge bereitgestellte Funktionalität vorstellen werden:

- Die Rahmenwerkarchitektur sollte stets so entworfen werden, daß durch sie eine Klasse von Problemen gelöst wird. So kann sie als wiederverwendbares Muster für ähnliche Umgebungen dienen.
- Die unspezifischen Komponenten des Rahmenwerks sollten auch in ähnlichen Umgebungen wiederverwendbar sein.
- Die spezifischen Komponenten sollten zusammengefaßt und als Block entfernt oder eingefügt werden können.
- Auch spezifische Komponenten können wiederverwendbare Bestandteile enthalten. Die Teile, die nicht wiederverwendbar sind, sollten durch Generatoren erzeugt werden können.
- Teile, die sich häufig ändern können, sollten nicht im Quelltext codiert sein. Diese Teile sollten vielmehr in Datenstrukturen, z.B. Tabellen, gespeichert und anschließend ausgewertet werden.
- Die Trennung von logischen Daten und (Re-)Präsentation der Daten ist dann sinnvoll, wenn es möglich sein soll, verschiedene Repräsentationen logischer Daten darzustellen.

Auf der Basis dieser Erkenntnisse und des oben beschriebenen Metaschemas wurde ein Rahmenwerk für Informationssysteme im Internet entwickelt, die speziell klassifizierte Informationen bereitstellen. Diese Informationen können beispielsweise Beschreibungen von *Produkten* oder im Intra- bzw. Internet vorhandenen *Dokumenten* sein. Wesentlich ist, daß ein Klassifikationsschema entsprechend des oben beschriebenen Metaschemas zur Klassifikation der Produkte/Dokumente entwickelt

werden kann. Die dort erläuterten verschiedenen Merkmalarten können, müssen aber nicht alle verwendet werden.

Abbildung 4 stellt nun die grobe Architektur des SYNERGIE-Rahmenwerks mit den spezifischen (grau hinterlegten) Komponenten und den Standardkomponenten entsprechend der in [Nag90] vorgestellten Notation dar. Die wesentlichen Bestandteile der Architektur sind i) Funktionsmodule (fm), die keinen internes „Gedächtnis" besitzen, ii) abstrakte Datenobjekte (ado), die Datenstrukturen mit „Gedächtnis" und sichtbarer Schnittstelle sowie verborgener Realisierung darstellen, iii) abstrakte Datentypen (adt), die Schablonen sind, mit deren Hilfe man abstrakte Datenobjekte erzeugt, iv) Teilsysteme (ts), die sich wieder aus mehreren Modulen zusammensetzen und v) allgemeine Benutzbarkeiten (Doppelpfeile), die darstellen, daß ein Modul ein anderes benutzt. Die Steuerung der gesamten Umgebung wird durch das Modul `Werkzeugsteuerung` übernommen. Der Zugriff auf die Datenbestände sowie deren Manipulation und Analyse erfolgt mit Hilfe der in der zweiten Schicht dargestellten Werkzeuge (`Anbieterverwaltung`, `Benutzerverwaltung`, `Komponentenverwaltung`, `Suche & Navigation`, `Schemaverwaltung`).

Neben dem Datenbestand müssen lediglich die grau hinterlegten Module im Quelltext angepaßt werden, wenn mit der Umgebung die Kooperation hinsichtlich anderer Produkte oder Dokumente ermöglicht werden soll. Diese Anpassung kann manuell erfolgen, man kann aber auch Generatoren erstellen, die die geänderten Module den Anforderungen entsprechend erzeugen. Die definierten Schnittstellen der Module/Teilsysteme bleiben dabei bestehen. So vermeidet man die Anpassung der Module, die auf geänderte Module aufsetzen. Momentan gibt es jedoch noch keine Generatoren für die spezifischen Komponenten des SYNERGIE-Rahmenwerks. Da keine sinnvolle Gelegenheit für den Einsatz des Rahmenwerks in einem anderen Kontext in Aussicht ist, haben wir dies zunächst zurückgestellt.

Wie man erkennen kann, sind lediglich die Layoutmodule (`Oberflächenlayout` und `Komponentenlayout`) spezifisch. Diese enthalten Vorschriften zum Aufbau der Benutzeroberfläche und zur Darstellung der Komponenten (oder allgemein: der Produkte/Dokumente). Zusammen mit dem gesamten Datenbestand (Komponente, Klassifikationsschema, Benutzer- und Anbieterdaten) werden sie durch den HTML-Generator genutzt, um die HTML-Oberfläche dynamisch zu generieren. Für die Suche und Komponentenverwaltung gibt es zusätzlich, für die Schemaverwaltung gibt es ausschließlich eine grafische Benutzeroberfläche, die am rechten Rand der Architektur als Teilsystem dargestellt ist. Diese grafische Benutzeroberfläche ist unabhängig von HTML und benutzt nur implizit definierte Layoutrichtlinien. Daher müßten auch dort

bei Erstellung eines neuen Informationssystems zur Förderung der Kooperation auf Basis des Rahmenwerkes Änderungen vorgenommen werden.

In der Architektur wurden mehrere Werkzeuge der Administratorumgebung aus Übersichtlichkeitsgründen nicht dargestellt. Hierzu gehören insbesondere Analyse- und Kontrollwerkzeuge.

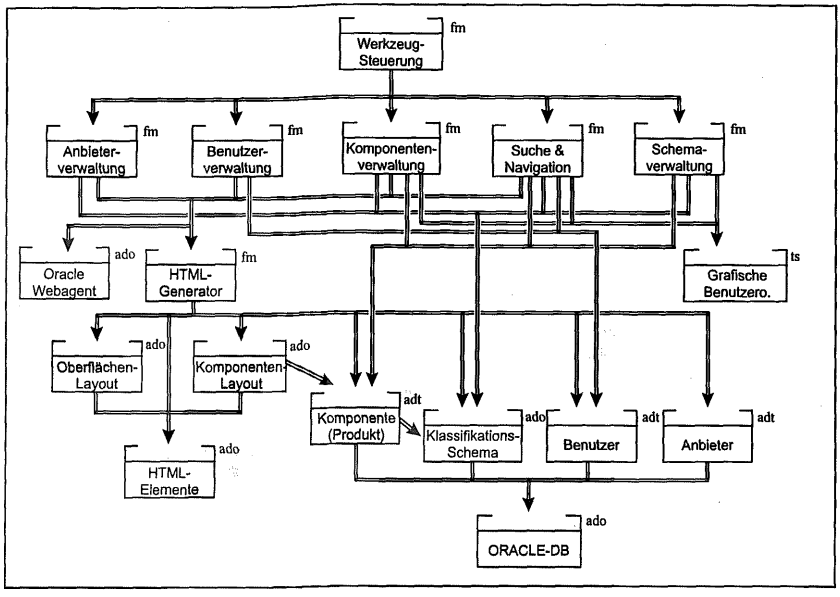

Abbildung 4: Architektur des SYNERGIE-Rahmenwerks

Die Realisierung dieser Architektur erfolgte durch Implementierungen in PL/SQL und Java. Der gesamten Implementierung liegt das oben vorgestellte Metaschema zugrunde. Das Klassifikationsschema und der restliche (auch) vom Klassifikationsschema abhängige Datenbestand wird in der ORACLE-Datenbank gespeichert. Diese „Tabellensteuerung" ist der Hauptgrund für die leichte Änderbarkeit des Schemas und für die Übertragbarkeit des Rahmenwerks in andere Produktkontexte. In dem Rahmenwerk wird der gesamte Datenbestand über benutzerabhängige, dynamisch generierte Bedieneroberflächen [Beh97] gepflegt. Dabei handelt es sich einerseits um HTML-Formulare, die mit Hilfe der PL/SQL-Routinen dynamisch generiert werden. Bei der Generierung der Sichten werden die Zugriffsrechte entsprechend der zugeordneten Benutzergruppen berücksichtigt. So wird beispielsweise garantiert, daß ein Hersteller nur die Daten der vom ihm angebotenen Produkte ändern kann, daß Administratoren ausgedehnte Rechte haben und daß neben den Projektpartnern später auch anderen Benutzern eine Registrierung ermöglicht werden kann. Dabei sollen aber

die Erfahrungen und die Ansprechpartner zu einzelnen Produkten nur innerhalb der Gruppe der Projektpartner und REGINA-Mitglieder sichtbar sein.
Andererseits werden komplexe grafische Benutzeroberflächen (z.B. der Arbeitsplatz zur Pflege des Klassifikationsschemas) in Form von Java Client/Server-Anwendungen angeboten, welche auf der Basis von JDBC und RMI entwickelt wurden [Als98, BC97]. Abbildung 5 zeigt eine vereinfachte Darstellung der Prozesse und Kommunikation einer dieser Anwendungen. Über JDBC kann ein Java-Prozeß auf dem Server sowohl auf die vorhandenen PL/SQL-Routinen als auch auf den Datenbestand zugreifen. Applets, die über den Web-Server in einen Web-Browser geladen werden, können dann mittels Remote Method Invocation (RMI) auf die Methoden des Java-Servers zugreifen.

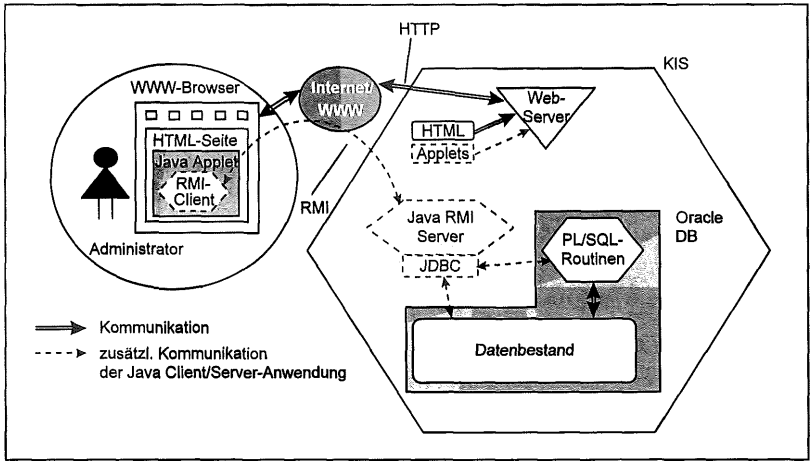

Abbildung 5: Java Client/Server-Anwendung mit RMI und JDBC

Alternativ kann auch die Verbindung zwischen Web-Browser und der Datenbank direkt über JDBC, also ohne zusätzlichen Java-Server, erfolgen. Dies ist die Lösung, die wir nach ausgiebiger Evaluierung aus Performanzgründen in unseren Werkzeugen zur Schemapflege, zur Komponentendatenpflege sowie zur Suche implementiert haben. Ein Java-Server wäre dann notwendig, wenn wir schreibende Zugriffe synchronisieren müßten. Dies können wir aber aufgrund des speziellen Szenarios ausschließen.

5 Verwandte Arbeiten

Die Mehrzahl der bestehenden Ansätze beschäftigt sich mit Softwarewiederverwendungsbibliotheken innerhalb *eines* Unternehmens. Zendler et al. haben in [ZG94, ZGH95] mehrere dieser Ansätze gegenübergestellt. Ansätze für Kooperation zwischen mehreren Unternehmen und auch für Erfahrungsaustausch von

Entwicklern im Bereich der Wiederverwendung von Komponenten sind uns jedoch nicht bekannt.

Eine detaillierte Übersicht zu prinzipiellen Möglichkeiten zur *Repräsentation* von wiederverwendbaren Softwarebausteinen liefert [GF90]. Ähnliche Ansätze zur *Softwareklassifikation* sind die facettierte Klassifikation [Pri91] und die merkmalsorientierte Klassifikation [Bör95]. Jedoch unterstützt die facettierte Klassifikation keine Beschreibungstexte. Weiterhin können Komponenten nicht auf verschiedenen Granularitätsstufen beschrieben werden, oft werden Komponenten beispielsweise lediglich auf Klassenebene betrachtet. Bei der merkmalsorientierten Klassifikation wird die Klassifikationshierarchie auf eine andere Art aufgebaut und ist insbesondere nicht auf verschiedene Zugriffsmechanismen (Suche/Navigation) ausgerichtet. Beide Ansätze unterstützen jedoch - im Gegensatz zu unserer derzeitigen Implementierung - Ähnlichkeiten zwischen Termen und Facetten, so daß wenn kein exaktes Ergebnis zu einer Anfrage gefunden wurde, ein ähnliches Ergebnis geliefert wird.

Andere Systeme, die sich mit Softwarewiederverwendung im Zusammenhang mit Internet-Technologien beschäftigen, sind ASSET [SAI97] und MOREplus [EMD94, Tru97]. ASSET beschäftigt sich mit wiederverwendbaren Dokumenten ganz allgemein (Standards, Werkzeuge, Dokumentationen, aber auch Klassenbibliotheken), ist aber nicht auf registrierte Benutzer ausgerichtet. Bei dem kommerziellen Produkt MOREplus handelt es sich um den insgesamt ähnlichsten Ansatz im Vergleich zu dem vorgestellten KIS, ebenfalls ohne Möglichkeiten zum Erfahrungsaustausch.

Das System JAVA REPOSITORY [BKR96] für Java-Klassen und Applets verwendet die gleiche Technologie für die Datenbank und den Web-Server wie der hier vorgestellte Ansatz. Im Gegensatz zu der Idee des KIS handelt es sich dabei jedoch um einen wenig kooperativen Ansatz, obwohl beliebige Internet-Benutzer Kommentare zu den Produkten angeben können. Die Beschreibung der Komponenten ist relativ kurz und basiert nicht auf einem hierarchischen Klassifikationsschema; es gibt verschiedene Kategorien und Schlüsselworte, denen eine Komponente zugeordnet werden kann. Technische Informationen (z.B. unterstütztes JDK, getestete Plattformen) werden nicht angeboten.

Das *Software Asset Library Management System* (SALMS) [Sod97] ist eine Client/Server-Anwendung mit grafischer Benutzeroberfläche zur Klassifikation, Beschreibung und Suche von einzelnen Softwarekomponenten sowie Kollektionen von Softwarekomponenten. Für die Repräsentation wird ein Klassifikationsschema verwendet, das in der in C++ implementierten Benutzeroberfläche fest codiert ist. Neben der Suche nach Stichworten ist auch die Navigation in dem Datenbestand

vorgesehen, welche entlang eines facettierten Klassifikationsschemas oder entlang Verweisen zwischen Komponenten (Spezialisierung_von, Weiterentwicklung_von, Teil_von) durchführbar ist. SALMS besitzt neben den reinen Bibliotheksfunktionen zwar keine Möglichkeiten zum Erfahrungsaustausch, es gibt jedoch Einträge für ein *Qualitätsmaß* und ein *Wiederverwendbarkeitsmaß*, welche durch eine zentrale Zertifizierungsstelle festgelegt werden. Durch die Wiedergabe des Klassifikationsschemas in der C++-Oberfläche wirkt sich jede Änderung des Klassifikationsschemas auch in der Implementierung der Benutzeroberfläche aus.
Der Schwerpunkt des *Federal Reuse Repository* (FFR) von Loral Federal Systems [PW95] liegt in der Bereitstellung einer benutzerfreundlichen Web-Oberfläche zur Suche nach beliebigen Softwarekomponenten und deren Eingabe. Realisiert wird die Web-Oberfläche durch *HTML-Formulare* und *strukturierte Zusammenfassungen* (engl. „Structured Abstracts"). Hierbei werden HTML-Formulare nicht tabellarisch oder stichwortartig, sondern in *ganzen Sätzen* aufgebaut, wobei dieselben Informationen immer in der *gleichen Reihenfolge* dargestellt werden. Der Entwickler muß dann lediglich bestimmte Teile der Sätze durch Selektion entsprechender Auswahllisten vervollständigen. Auf diese Art können Komponentenbeschreibungen erstellt oder gesucht werden. Ziel von Poulin und Werkman [PW95] ist es, durch die Darstellung vollständiger Sätze sowohl die Probleme der Klassifikation als auch des Programmverstehens zu lösen. Neben diesem Klassifikationsansatz werden vorhandene Dokumente mit dem frei verfügbaren Werkzeug *Wide Area Information Services* (WAIS) indexiert, um so zusätzlich eine Stichwortsuche zu ermöglichen. Der Ansatz geht davon aus, daß das Klassifikationsschema unveränderlich ist. Das Schema diente als Grundlage für den Entwurf des Datenbankschemas und die Implementierung der Benutzeroberfläche, wodurch spätere Änderungen nur mit größerem Aufwand möglich sind.

6 Zusammenfassung und Ausblick

Wir haben in diesem Beitrag den Ansatz des im RSB-Projekt entwickelten Komponenteninformationssystems vorgestellt. Wir sind dabei auf Aspekte der Kooperation, den Klassifikationsansatz und das zugrundeliegende SYNERGIE-Rahmenwerk eingegangen. Wir haben erläutert, daß mit diesem Rahmenwerk auch andere Informationssysteme zur Unterstützung unternehmensübergreifender Kooperation entwickelt werden können und daß aufgrund des tabellengesteuerten Ansatzes dazu lediglich Anpassungen im Layoutbereich notwendig sind.
Wir haben bereits erwähnt, daß bisher in der Implementierung keine Ähnlichkeiten zwischen Komponenten berechnet werden. Die Anzeige ähnlicher Komponenten in dem

Falle, daß keine Komponente gefunden werden konnte, die den Anforderungen exakt genügt, ist aber notwendig. Wir entwickeln und implementieren daher zur Zeit ein *Ähnlichkeitsmaß*, das nicht auf explizite Werte zur Formulierung von Ähnlichkeiten zwischen Merkmalen zurückgreift, sondern auf die *implizit* durch das Klassifikationsschema definierten Zusammenhänge.

Ziel des KIS ist, neben den Komponenten der Kooperationspartner auch über kommerziell oder frei verfügbare Komponenten zu informieren. Dazu ist eine intensive Recherche nach Komponenten aller Art notwendig. Obwohl das KIS zur Zeit fast 600 Komponenten enthält, gibt es gerade im Bereich der JavaBeans und ActiveX-Controls viele weitere Produkte. Fast alle Anbieter informieren über ihre Produkte im Internet, so daß man große Teile der Recherchen im Internet durchführen kann. Dies ist jedoch mühsam und kann durch Werkzeuge unterstützt werden. Gegenwärtig schließen wir die Entwicklung eines Werkzeugs ab, das die im Internet *vorhandenen Suchmaschinen nutzt*, um Komponentenbeschreibungen ausfindig zu machen und (semi-) *automatisch Beschreibungen* (Klassifikationen) der Komponenten für das KIS zu *erzeugen*.

Das System ist im Internet frei verwendbar, so daß wir in der Lage sind, Erfahrungen im Hinblick auf die Akzeptanz des Inhalts, die Präsentation und die Mechanismen zur Suche und Navigation und Registrierung zu sammeln. Der Nutzen dieses Systems allgemein, die Synergieeffekte, der Gewinn durch die Wiederverwendung von Softwarekomponenten innerhalb des Konsortiums und die Nützlichkeit der Erfahrungen, die andere Entwickler beim Einsatz von Komponenten gemacht haben, müssen zudem evaluiert werden.

Danksagung

Das RSB-Projekt wird von dem Ministerium für Wirtschaft und Mittelstand, Technologie und Verkehr und dem Ministerium für Forschung und Wissenschaft des Landes NRW sowie von dem Regionalen Industrieclub Informatik Aachen e.V. (REGINA) gefördert. Danksagen möchten wir vor allem den Projektleitern und allen Projektpartnern, die teilweise auch an der Implementierung beteiligt waren. Die Forschungsergebnisse, über die in diesem Beitrag berichtet wurde, konnten nur durch die gute Kooperation mit den Projektpartnern gewonnen werden.

Literatur

[AlS98] M.A. Al-Sayed Ali: *Einsatz von Java zur Entwicklung von Client/Server-Systemen am Beispiel einer Datenbankanwendung im World Wide Web,* Diplomarbeit des Lehrstuhls für Informatik III an der RWTH Aachen, Februar 1998.

[BC97] A. Behle, K. Cremer: Einsatz und Lehre von Java zur Entwicklung verteilter Applikationen, In *Tagungsband Smalltalk und Java in Industrie und Ausbildung*, Erfurt, September 1997, Ilmenau, 1997: TU Ilmenau, 108-113.

[Beh97] A. Behle: Ein dynamisches, benutzerabhängiges Informationssystem im World Wide Web, EMISA-Fachgruppentreffen Okt. 1996 „Informationsserver für das Internet - Anforderungen, Konzepte, Methoden", Aachen, *EMISA Forum* 1/1997, 75-79.

[Beh98] A. Behle: An Internet-based Information System for Cooperative Software Reuse, *Proc.of the 5th International Conference on Software Reuse*, Victoria, British Columbia, Canada, June 2-5 1998, IEEE Computer Society Press, 236-245.

[BKR96] P. Buxmann, W. König, F. Rose: The Java Repository - An Electronic Intermediary for Java Resources, In *Proc. 7th Annual Conference of the International Information Management Association* (IIMA), Estes Park, Colorado, December 1996.

[Bör95] J. Börstler: Feature-Oriented Classification for Software Reuse, In *Proc. of the 7th Intern. Conference on Software Engineering and Knowledge* (SEKE '95), Rockville, ML, USA, June 1995, 204-211.

[Buc89] B. Buchanan: *Bibliothekarische Klassifikationstheorie*, München: Saur, 1989.

[EMD94] D. Eichmann, T. McGregor, D. Danley: Integrating Structured Databases into the Web: The MORE System, In *Proc. of the First International Conference on the World Wide Web*, Geneva, Switzerland, May 25-27, 1994.

[FG90] W.B. Frakes, P.B. Gandel: Representing Reusable Software, *Information and Software Technology* 32(10), 1990, 641-664.

[Kle96] P. Klein: The Framework Revisited: A More Detailed View on the IPSEN Architecture, In [Nag96], 380-396.

[KSW95] N. Kiesel, A. Schürr, B. Westfechtel: GRAS, A Graph-Oriented (Software) Engineering Database System, In *Information Systems*, Vol. 20, No. 1, Oxford: Pergamon Press, 1995, 21-52.

[Nag90] M. Nagl: *Softwaretechnik: Methodisches Programmieren im Großen*, Berlin: Springer-Verlag, 1990.

[Nag96] M. Nagl (Hrsg.): *Building Tightly Integrated Software Development Environments: The IPSEN Approach,* Lecture Notes in Computer Science, Vol. 1170, Berlin, Germany: Springer-Verlag, 1996.

[Pri91] R. Prieto-Díaz: Implementing Faceted Classification for Software Reuse, In *Communications of the ACM* 34(5), May 1991, 88-97.

[PW95] J.S. Poulin, K.J. Werkman: Melding Structured Abstracts and the WWW for Retrieval of Reusable Components, In *ACM SigSoft Software Engineering Notes*, August 1995, 160-168.

[SAI97] Science Applications International Corporation (SAIC): ASSET - Asset Source for Software Engineering Technology, Juni 1997.

[Sod97] Sodalia SpA: SALMS v5.1: A system for classifying, describing, and querying about reusable software assets produced throughout the software life-cycle, March 27, 1997.

[Tru97] D. Trump: Using the WWW and the Internet to Support Corporate Reuse, In *Proc. of the 8th Workshop on Institutionalizing Software Reuse* (WISR8), Columbus, Ohio, USA, March 23-26, 1997.

[ZG94] A. Zendler, S. Gastinger: Werkzeuge zum Aufbau und Einsatz von Bibliotheken zur Software-Wiederverwendung: Kriterien und Anforderungen, *FAST-Bericht* Nr. 94-08, Forschungsinstitut für Angewandte Software-Technologie, München, Dezember 1994.

[ZGH95] A. Zendler, S. Gastinger, R. Haggenmüller: Vergleichende Analyse von Werkzeugen zum Aufbau und Einsatz von Bibliotheken für wiederverwendbare Software-Dokumente, *FAST-Bericht* Nr. 95-04, Forschungsinstitut für Angewandte Software-Technologie, München, März 1995.