Ein dynamisches, benutzerabhängiges Informationssystem im World Wide Web

Anita Behle
Lehrstuhl für Informatik III
RWTH Aachen, 52056 Aachen
behle@i3.informatik.rwth-aachen.de

Zusammenfassung

Im Rahmen des RSB-Projektes* wird ein Komponenteninformationssystem für das WWW entwickelt. Wir stellen in diesem Beitrag Konzepte dieses Informationssystems, auftretende Probleme und mögliche Lösungsansätze vor. Die dabei gewonnenen Erkenntnisse lassen sich auf beliebige Informationssysteme im World Wide Web übertragen, die eine große Menge strukturierter Daten über eine benutzerfreundliche Oberfläche bereitstellen. Zum Entwurf der Datenbank wird zunächst ein Klassifikationsansatz vorgestellt. Anschließend wird die Notwendigkeit eines Metaschemas auf der Basis dieses Klassifikationsansatzes erläutert, durch das einerseits verschiedene Dialogkonzepte zum Zugriff auf die Daten ermöglicht werden, andererseits das System leicht modifizierbar ist. Weiterhin wird ein Konzept für eine dynamische, benutzerabhängige Bedieneroberfläche erläutert, welches über die eingeschränkte, zustandslose Interaktion auf Basis statischer HTML-Seiten hinausgeht.

1 Einleitung

Im Rahmen des Projektes RSB (REGINA-Software-Bibliothek), an dem mehrere Mitglieder der REGINA (Regionaler Industrie-Club Informatik Aachen e.V.) teilnehmen, wollen wir ein Komponenteninformationssystem im World Wide Web (WWW) erstellen. Es soll registrierten Benutzern Informationen über kommerzielle oder frei erhältliche Komponenten liefern und so die Wiederverwendung bestehender Software unterstützen. Den Anbietern von Komponenten wollen wir weiterhin die Möglichkeit geben, ihre Komponenten für das Informationssystem vorzuschlagen. Unter einer Komponente verstehen wir dabei einen wiederverwendbaren Software-Baustein, beispielsweise eine C++-Klassenbibliothek im Quellcode, eine statische oder eine dynamische Library.

Im Gegensatz zu vielen Wiederverwendungsansätzen verwalten wir Informationen zu Komponenten, nicht die Komponenten selbst. Bei den Daten zur Beschreibung einer Komponente handelt es sich um eine große Menge unterschiedlichster Informationen (Name, Funktionalität, unterstützte Plattformen und Compiler, Bezugsquellen, Anwendungsbereiche, Umfang der Dokumentation, ...). Diese Informationen wollen wir nicht als Text ablegen. Wir wollen die Daten strukturieren und so ein Klassifikationsschema erstellen, durch das die Komponenten eingeordnet und verglichen werden können. Zur Ablage der strukturierten Informationen setzen wir eine Datenbank ein und können so die Darstellung der Informationen für den Kunden zur Laufzeit auf der Basis des aktuellen

* Dieses Projekt wird gefördert durch das Ministerium für Wissenschaft und Forschung des Landes Nordrhein-Westfalen und durch das Ministerium für Wirtschaft, Mittelstand und Technik des Landes Nordrhein-Westfalen.

Datenbankzustandes erzeugen. Die Anforderungen an dieses Informationssystem lassen sich daher auf beliebige Informationssysteme im World Wide Web übertragen, die eine große, durch ein Klassifikationsschema strukturierte Datenmenge bereitstellen und vielfältige Anfragemöglichkeiten dazu bieten wollen.

Die Konzepte des Komponenteninformationssystems, sollen nun zusammen mit einigen Problemen und möglichen Lösungsansätzen vorgestellt werden. Dabei behandeln wir zwei Schwerpunkte.

Erstens wird ein Klassifikationsschema zusammen mit den Anforderungen an ein Metamodell zum Entwurf des Datenbankschemas auf der Basis des Klassifikationsschemas vorgestellt. Dadurch werden verschiedene Dialogkonzepte für den Zugriff auf die Informationen ermöglicht. Bei diesen Dialogkonzepten handelt es sich um Möglichkeiten zur konkreten Suche sowie zur Navigation in der Datenbank auf Basis des Klassifikationsschemas. Zusätzlich gewährleistet das Metaschema die Erweiterbarkeit und leichte Modifizierbarkeit des Klassifikationsschemas.

Zweitens wird ein Konzept für eine dynamische, benutzerabhängige Bedieneroberfläche erläutert, deren Layout der Benutzer selbst bestimmen kann. In diesem Zusammenhang wird zum einen ein Benutzerprofil erstellt, welches bei dem dynamischen Aufbau der HTML-Seiten ausgewertet wird und so dazu beiträgt, daß die HTML-Seiten nach den Präferenzen des Benutzers gestaltet werden. Zum anderen wird eine alternative Benutzeroberfläche auf der Basis von Java Applets erläutert, welche die Nachteile der eingeschränkten, zustandslosen Interaktion zwischen Client und Server auf der Basis von HTML-Seiten umgeht.

2 Konzepte des Komponenteninformationssystems

2.1 Datenspeicherung

Der Zugriff auf Daten erfolgt im WWW [Ram 95] prinzipiell durch die Übertragung von Textseiten vom Server zum Client auf der Basis des *Hypertext Transfer Protocols* (http). Diese Textseiten müssen entsprechend der *Hypertext Markup Language* (HTML) [KS 95], [Mor 95] aufgebaut sein, man nennt sie deswegen auch HTML-Seiten. Für die Speicherung der Daten bestehen nun zwei Möglichkeiten:

1. **Speicherung in statischen HTML-Seiten**: Hier wird jede HTML-Seite physikalisch als Textdatei auf dem WWW-Server abgelegt. Beim Zugriff wird die vorhandene Datei zum Client kopiert. Die Erzeugung dieser Textdateien kann ggf. programmgesteuert erfolgen.
2. **Speicherung in einer Datenbank und dynamische Erzeugung der HTML-Seiten**: Über das Common Gateway Interface (CGI) besteht die Möglichkeit, daß ein Client auf dem Server die Ausführung einer Applikation oder eines Skripts initiiert. Eine solche CGI-Applikation kann als Ausgabe einen HTML-Text erzeugen, welcher zum Client übertragen und dort angezeigt wird.

   So ist es zum Beispiel möglich, daß der Benutzer in einer statischen HTML-Seite ein Formular mit einer Datenbankabfrage (z.B. Stichwortsuche) ausfüllt und das Auswahlfeld *Suchen* aktiviert. Das gesuchte Stichwort wird zum Server übertragen, welcher eine CGI-Applikation mit einer Datenbankabfrage startet. Das Ergebnis der Anfrage wird dann als HTML-Text aufbereitet und dem Client als Ergebnis zurückgesendet. Für den Client besteht in der Anzeige statischer oder dynamisch erzeugter HTML-Seiten kein Unterschied.

Wie wir bereits erläutert haben, wollen wir die Daten in einer Datenbank ablegen. Somit muß der Zugriff auf die Daten über das CGI und entsprechende Datenbankschnittstellen erfolgen. Die Strukturierung der Daten erfolgt auf Basis eines Klassifikationsschemas. Im Software-

Wiederverwendungsbereich haben sich dazu verschiedene Varianten der facettenorientierten Klassifikation durchgesetzt [Kar 95], [Pri 91]. Bei der hier eingesetzten *merkmalsorientierten Klassifikation* [Bör 93a], [Bör 93b] handelt es sich um eine Erweiterung der facettenorientierten Klassifikation. Es gibt dabei prinzipiell vier Konzepte:

- Eine 'is'-Beziehung entspricht einer Verfeinerung (Spezialisierung).
- Eine 'view'-Beziehung entspricht einer Komposition.
- Ein '*' bedeutet eine beliebige Wiederholung (0..n).
- Zwischen Schemaeinträgen können weitere Beziehungen definiert werden, z.B. zur Formulierung von Ähnlichkeiten oder Querbezügen.

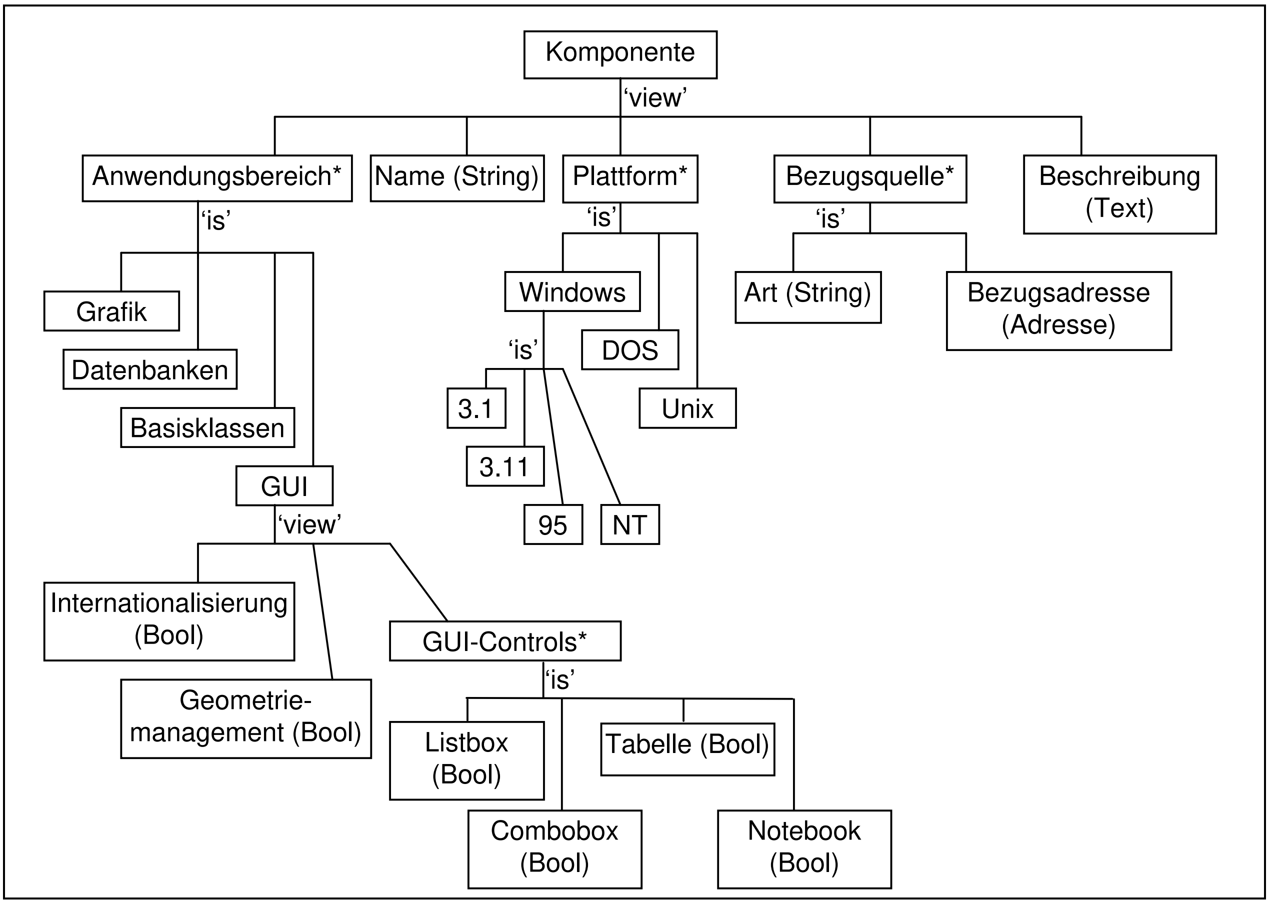

Abbildung 1: Ausschnitt aus dem Klassifikationsschema für Komponenten

Abbildung 1 zeigt einen kleinen Ausschnitt aus einem Klassifikationsschema für Software-Bausteine. Eine Komponente besitzt einen Namen, eine Beschreibung, mehrere Bezugsquellen, kann für mehrere Plattformen erhältlich sein und kann zu mehreren Anwendungsbereichen gehören. Eine Bezugsquellenangabe besteht aus der Angabe der Art (z.B. Hersteller, Vertrieb, Public Domain) und einer Adresse. Die Art der Bezugsquelle wird durch einen String angegeben, die Bezugsquellenadresse durch eine Adresse. Es werden drei Plattformen unterschieden, wobei hier außerdem mehrere Windows-Versionen aufgeführt werden. Die Anwendungsbereiche Grafik, Datenbanken und Basisklassen und die verschiedenen Plattformen sind nicht vollständig spezifiziert.

Ein solches Klassifikationsschema muß erweiterbar sein, wenn beispielsweise ein weiterer Anwendungsbereich oder eine zusätzliche Plattform aufgenommen werden soll. Speziell eine vollständige Analyse aller Anwendungsgebiete können wir bei der Installation des Informationssystems nicht voraussetzen, weil (a) das Know-how zu verschiedenen Anwendungsgebieten ständig wächst, (b) die Analyse aller bekannten Anwendungsgebiete kostspielig ist und (c) während des Betriebs des Informationssystems neue Anwendungsgebiete entstehen werden. Daher müssen wir mit einem unvollständigen Klassifikationsschema beginnen und mögliche Erweiterungen und Änderungen von vornherein berücksichtigen.

Da alle Komponenten durch das Schema klassifiziert werden und das Schema die Struktur zur Beschreibung der Komponenten enthält, bildet es folglich die Basis für die Dialoge zur Suche und Navigation. Damit sich die Erweiterung des Klassifikationsschemas nicht wesentlich auf die Datenbank auswirkt, muß der Datenbank statt des konkreten Klassifikationsschemas, ein Metaschema zugrunde gelegt werden. Mit Hilfe des Metaschemas werden die grundsätzlichen Zusammenhänge modelliert, die in einem Klassifikationsschema auftreten können. Ein konkretes Klassifikationsschema, eine Instanz des Metaschemas, kann man dann abhängig vom Metaschema spezifizieren. Die Daten des Klassifikationsschemas, die Metadaten, bilden anschließend die Grundlage für den Aufbau der Dialoge für den Zugriff auf die Komponentendaten. Ändert sich nun das Klassifikationsschema, so ändern sich die Metadaten und mit ihnen die betroffenen Teile der Datenbank. Da die Formulare auf der Basis der Metadaten aufgebaut werden, ändern sich ihre Implementierung und Auswertung nicht, wenn sich das Klassifikationsschema ändert. Wenn hingegen kein Metaschema existiert, berücksichtigt die Implementierung zum Zugriff direkt die Datenbankstruktur und muß bei jeder Änderung des Klassifikationsschemas angepaßt werden.

Bei einer Schemaänderung müssen die Datenbestände ggf. angepaßt werden. Börstler [Bör 93a] hat gezeigt, daß dies für Standardtransformationen bei der merkmalsorientierten Klassifikation automatisiert werden kann.

Eine auf Basis des zu erstellenden Metaschemas entwickelte Datenbank ist somit für die Aufnahme klassifizierter Daten und deren Zugriff geeignet. Außerdem können sowohl das Klassifikationsschema als auch die Daten leicht ergänzt und gepflegt werden. Weiterhin kann aus den Metadaten das Klassifikationsschema ermittelt werden. Dies ist Voraussetzung für die folgenden Dialogkonzepte.

2.2 Zugriffsmöglichkeiten

Die Akzeptanz des Komponenteninformationssystems ist - neben der Korrektheit, Aktualität und Vollständigkeit der angebotenen Informationen - direkt abhängig von der Benutzerfreundlichkeit der Zugriffsmöglichkeiten. Es sollen verschiedene Zugriffsmöglichkeiten angeboten werden, mit welchen die Suche nach Daten durchgeführt werden kann. Je nach Bedarf und Wissensstand kann der Benutzer direkt nach konkreten Begriffen suchen, oder er kann sich darüber informieren, welche Komponenten gesuchte (Klassifikations-)Merkmale erfüllen. Er kann sich beispielsweise darüber informieren, welche C++-Klassenbibliotheken für Unix den Zugriff auf relationale Datenbanken ermöglichen oder welche Software-Bausteine für grafische Benutzeroberflächen portabel zwischen Windows, Unix und OS/2 sind. Um diese verschiedenen Zugriffsarten zuzulassen, sind zwei Dialogkonzepte notwendig: *Suche* und *Navigation.* Der Aufbau dieser Dialoge (HTML-Formulare) geschieht auf der Basis des Klassifikationsschemas bzw. der Metadaten.

2.2.1 Suche

Die Suche nach Komponenten erfolgt durch HTML-Formulare, in welche der Benutzer Suchbegriffe eingeben kann. Außerdem kann der Suchbereich eingeschränkt werden, beispielsweise auf einen Anwendungsbereich, bestimmte Plattformen oder Bezugsquellen aus der Public Domain. Die Einschränkungen können weiterhin sukzessive verfeinert werden. Durch das

Klassifikationsschema ist stets vorgegeben, aus welchen Teilen ein Formular bestehen muß und welche Kriterien eingeschränkt werden können.

2.2.2 Navigation

Durch *Navigation* kann sich der Benutzer verschiedene logische Bereiche der Komponentendatenbank anschauen. Anhand eines Navigationsschemas, welches hier durch das Klassifikationsschema vorgegeben ist, können sich die Benutzer einen Überblick über die Struktur der Komponentendaten verschaffen und Daten zu Komponenten aus interessanten Bereichen anzeigen lassen. Die Navigation selbst erfolgt durch das Verfolgen von sog. *Links.* Sie können als das charakteristische Merkmal des WWW zur logischen Verbindung von HTML-Dateien bezeichnet werden. Der Aufbau eines solchen Schemas besitzt zunächst Baumstruktur, wird jedoch, wenn man Verweise zwischen Komponenten hinzunimmt, zu einem Graphen.

2.2.3 Dynamische Benutzeroberfläche durch Java Applet

Im obigen Abschnitt haben wir Konzepte zum Zugriff auf die Datenbank erläutert, die von fast jedem WWW-Browser unterstützt werden und die auf den Konzepten des WWW aufsetzen. Eine weitere Möglichkeit, die den Zugriff des Benutzers auf die Daten erlaubt, bietet Java [JMC 96], [Küh 96]. Java ist eine objektorientierte Programmiersprache. Sie unterstützt das WWW einerseits durch eine *spezielle Netzwerk-Klassenbibliothek*, andererseits kann man in Java Applikationen (Applets) entwickeln und in portablem Bytecode auf dem Server ablegen. Die Ausführung erfolgt durch einen Java-Interpreter im WWW-Browser des Clients. Die beiden Möglichkeiten zur Unterstützung des WWW durch Java wollen wir als Alternative zur HTML-Benutzeroberfläche nutzen.

Bedieneroberfläche als Java Applet

Ein Java Applet kann eine kleine bewegte Animation oder eine vollständige grafische Benutzeroberfläche sein. So können Menüs, Dialoge und andere grafische Kontrollelemente wie Listen, Radio-Buttons, Check-Boxen und Combo-Boxen zum Aufbau der Benutzeroberfläche verwendet werden. Außerdem kann auf dem Client-Rechner zur Laufzeit eine Auswertung der Eingaben (z.B. Überprüfung auf Korrektheit) vorgenommen werden und die Oberfläche kann sich je nach Aktionen des Benutzers verändern. Man umgeht so die Einschränkungen des HTML-Formates, durch das zwar Formulare festgelegt werden können, mit welchem jedoch zur Laufzeit beim Benutzer weder Plausibilitätsuntersuchungen stattfinden können noch der Aufbau eines Formulars beim Client den Benutzeraktionen angepaßt werden kann. Durch ein Java Applet kann eine Benutzeroberfläche zusammen mit dem Wissen über die Struktur des Klassifikationsschemas zum Client übertragen werden. Dort kann die Angabe der gesuchten Komponente durch Stichwortsuche oder Navigation soweit verfeinert werden, daß Anfragen nahezu vollständig sind. Dann werden die Daten dem Server zugesandt, welcher die Anfrage bearbeitet.

Kommunikation durch Java Applet

Java Applets können weiterhin zur Kommunikation zwischen Client und Server außerhalb des http-Protokolls genutzt werden. Durch die Kommunikation eines Client-Applets mit dem Server kann die Datenbankabfrage im WWW als verteilte Applikation gestaltet werden, die nicht auf die Fähigkeiten von HTML beschränkt ist. azu bestehen zur Zeit zwei Möglichkeiten:

1. Java.net
   Die Netzwerkklassen der Standard-Java-Klassenbibliothek bieten die Möglichkeit zur

Kommunikation zwischen Client und Server auf Socket-Ebene. Der Anbieter eines Informationssystems muß dazu ein proprietäres Protokoll implementieren.

2. Java-CORBA-Anbindung
   Durch mittlerweile verfügbare Anbindungen von Java an CORBA (Common Object Request Broker Architecture) [MZ 95], [OMG 95], wie beispielsweise *BlackWidow* und *OrbixWeb*, kann die Kommunikation zwischen zwei Java-Objekten im Internet auf Corba-Implementierungen abgestützt werden. So kann ein Java-Objekt des Clients Methoden eines Server-Objektes transparent aufrufen. Ein aif dem Client laufendes Applet kann also unter Umgehung des http-Protokolls mit einem Java-Objekt auf dem Server kommunizieren und beispielsweise fehlende Daten zu einer Komponenten oder Teile des Klassifikationsschemas ermittelt und sofort anzeigen.

2.3 Registrierte Benutzer und Benutzerprofil

Der Zugang zu dem Informationssystem soll neben registrierten Benutzern eingeschränkt auch Gästen, welche das System kennenlernen möchten, möglich sein. Da die Kommunikation im WWW zustandslos ist und außerdem kein Standard-Mechanismus für einen Login-Vorgang vorgesehen ist, ist dies bei der Entwicklung des Informationssystems explizit vorzusehen. Einstellungen, die der Benutzer einmal vorgenommen hat, sollen gespeichert werden und so beim nächsten Login bekannt sein. Einstellbare Optionen sind beispielsweise Sprache, Währung, Layout für Formulare und Layout der Ergebnisanzeige. Da es mittlerweile viele verschiedene WWW-Browser gibt, welche auf unterschiedlichen Plattformen unterschiedliche Darstellungsarten besitzen können, ist die Möglichkeit zur Beeinflussung des Layouts und des Seitenaufbaus besonders wichtig. Der Benutzer soll so beispielsweise den Aufbau der Formulare durch Angabe der gewünschten Einstellungen beeinflussen können, da die Formulare, ebenso wie die Anfrageergebnisse, dynamisch generiert werden.

Neben den expliziten Einstellungen tragen Präferenzen und die zugehörige Benutzergruppe zum *Benutzerprofil* bei. Durch die Präferenzen können Interessengebiete des Benutzers beim Aufbau der Formulare und Navigationsstrukturen berücksichtigt werden. Ein ähnlicher Ansatz wird in [KBA 95] für eine Tageszeitung im WWW vorgestellt. Weiterhin ist eine *Sitzungsverwaltung* notwendig, damit der Zugang zu dem System gesichert ist, sich der Benutzer aber innerhalb einer Sitzung nicht bei jedem Server-Zugriff neu identifizieren muß.

2.4 Aufbau

Das Szenario des Komponenteninformationssystems im WWW wird in Abbildung 2 dargestellt.

Auf der Client-Seite gibt es zwei Arten von Benutzern, welche über beliebige WWW-Browser auf das Komponenteninformationssystem zugreifen können: *Kunden* und *Anbieter*. Die Kunden können sich über vorhandene Komponenten informieren, die Anbieter (Hersteller, Vertriebsfirmen, ...) können Komponenten für die Aufnahme in das Informationssystem vorschlagen.

Die Server-Seite besteht aus einem WWW-Server, welcher den Zugriff auf die statischen HTML-Seiten und auf die Datenbank steuert. Zu den statischen Seiten zählen beispielsweise Hilfsseiten, Bedienungsanleitung und Begrüßungsseite.

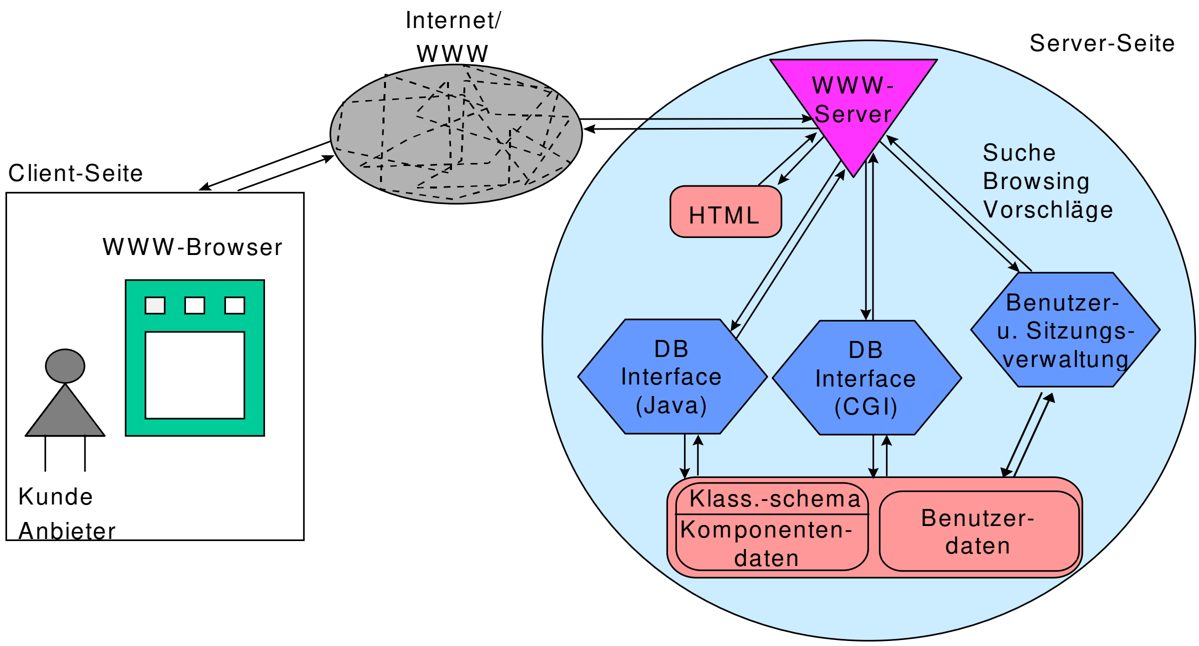

Abbildung 2: Aufbau des Komponenteninformationssystems im World Wide Web

Der Zugriff auf die Komponentendatenbank kann einerseits über CGI, andererseits über eine Java Applikation erfolgen. In jedem Fall wird das Klassifikationsschema berücksichtigt. Über die Benutzer- und Sitzungsverwaltung werden außer-dem Benutzerdaten und Benutzerprofile gepflegt. Das Benutzerprofil wird beim Datenbankzugriff ausgewertet, um den Aufbau der Formulare und Ergebnisse entsprechend den Einstellungen zu gestalten.

3 Ausblick

Realisierung

Das Komponenteninformationssystem soll auf der Basis des ORACLE WebSystems 2.0 auf Windows NT entwickelt werden. Es besteht aus dem ORACLE Web Request Broker, einem WWW-Server, der das Sicherheitsprotokoll Secure Socket Layer (SSL) 2.0 unterstützt, sowie einer ORACLE7 Datenbank und dem ORACLE WebServer Software Development Kit, welches neben einer PL/SQL- und einer LiveHTML-Schnittstelle auch eine Java-Anbindung an die ORACLE7-Datenbank unterstützt.

Entwicklungsstand

Bisher wurde ein Prototyp auf der Basis der Produkte mSQL und W3-mSQL [Hug] zur Untersuchung der verschiedenen Dialogkonzepte und Layout-Einstellungen erstellt.

Weiterhin haben wir mit der Entwicklung eines Prototyps auf Oracle-Basis begonnen. Zunächst beschäftigten wir uns mit der Implementierung des Datenbankschemas und der dynamischen Generierung der Benutzeroberfläche. Untersuchungen, ob Java Applets als grafische Benutzeroberfläche für das Komponenteninformationssystem geeignet sind und in welcher Form die Kommunikation zwischen Java-Objekten auf Client- und Server-Seite sinnvoll ist, werden folgen. Außerdem werden wir das oben beschriebene Benutzerprofil erstellen und bei der Generierung der Benutzeroberfläche berücksichtigen.

Referenzen

[Bör 93a] Börstler, Jürgen: *Programmieren im Großen: Sprachen, Werkzeuge, Wiederverwendung*, Dissertation, RWTH Aachen, 1993.

[Bör 93b] Börstler, Jürgen: *FOCS: A Classification System for Software Reuse*, Proceedings of the 11^{th} Pacific Northwest Software Quality Conference, 1993.

[Hug] http://Hughes.com.au/product/msql, http://Hughes.com.au/product/w3-msql

[JMC 96] Jackson, Jerry R.; McClellan, Alan L.: *Java by example*, SunSoft Press, 1996.

[Kar 95] Karlsson, Even-André: *Software Reuse - A Holistic Approach*, Wiley, 1995.

[KBA 95] Kamba, Tomorani; Bharat, Krishna; Albers, Michael C.: *The Krakatoa Chronicle - An Interactive, Personalized, Newspaper on the Web*, Fourth International World Wide Web Conference, Boston, Massachusetts, USA, http://www.w3.org/pub/Conferences/WWW4/Papers, Dezember 1995.

[KS 95] Koch-Steinheimer, Peter: *HTML: Veröffentlichen im Internet*, Deutsch, 1995.

[Küh 96] Kühnel, Ralf: *Die Java-Fibel*, Addison-Wesley, 1996.

[Mor 95] Morris, Mary: *HTML : WWW effektiv nutzen*, Heise, 1995.

[MZ 95] Mowbray, Thomas J.; Zahavi, Ron: *The Essential CORBA: Systems Integration Using Distributed Objects*, Object Management Group, Wiley, 1995.

[OMG 95] *The Common Object Request Broker: Architecture and Specification*, Revision 2.0, Object Management Group, July 1995.

[Pri 91] Prieto-Díaz, Rubén: *Implementing Faceted Classification for Software Reuse*, Communication of the ACM, Vol. 34, No. 5, May 1991.

[Ram 95] Ramm, Frederik: *Recherchieren und Publizieren im World Wide Web*, Vieweg, 1995.