Abschlussbericht zum Projekt Elektronischer Aufsatzdienst (ELEKTRA II)

Prof. Ph.D. Rudolf Bayer, Dr. Reiner Kallenborn, Rudolf Heinrich, Dirk Nitsche

Institut für Informatik Technische Universität München Orleansstr. 34 81667 München
Fax: (089) 48095-170 EMail: (bayer, kallenbo, heinrich, nitsche)@in.tum.de

1 Einleitung

Zusammenfassung

Es war das Ziel von Elektra, ein modernes digitales Bibliothekssystem zu entwickeln, das alle Phasen der Literaturversorgung im wissenschaftlichen Bereich umfasst, also

- die bibliographische Erfassung
- den Nachweis
- das Browsing und
- die Lieferung an den Bibliotheksbenutzer auf elektronischem Weg.

Da Projekt konnte sich dabei auf die Ergebnisse und Erfahrungen von Vorläuferprojekten abstützen, nämlich

- OMNIS (entwickelt in Eigenleistung am Lehrstuhl von Prof. Bayer)
- Öttingen Wallerstein (in Kooperation mit der UB Augsburg, gefördert durch die DFG)
- DIBWIN/Elektra I (Projekt *Bayern Online* der Bayerischen Staatsregierung)
- VD17: Verzeichnis der Drucke des 17 Jahrhunderts im Deutschen Sprachraum (in Kooperation mit sechs großen wissenschaftlichen Bibliotheken, gefördert durch die DFG)

Als Industriepartner sollte Swets&Zeitlinger sich an dem Projekt beteiligen und seine eigenen Zeitschriftenserver in Elektra integrieren. In Folge des Verkaufs von Swets&Zeitlinger an die Firma Blackwell und der damit verbundenen Restrukturierungen hat sich Swets&Zeitlinger schließlich doch nicht an Elektra beteiligt.

Elektra ist ein sehr komplexes System, das aus den folgenden Subsystemen besteht:

- Benutzerverwaltung (User Administration)
- Suche von Dokumenten, Bestellung und Lieferung (Search and Order)
- Archivierer

Elektra konnte vollständig realisiert werden und bis auf die fehlende Integration mit Swets&Zeitlinger alle gesteckten Ziele erreichen. Eine ausführliche technische Beschreibung des Systems findet sich in den folgenden Kapiteln.

Aufgrund der allgemeinen Situation auf dem IT Markt ergaben sich Schwierigkeiten und Engpässe im Personalbereich, die zu einer Verzögerung des Projektes führten, aber schließlich überwunden werden konnten, so dass das System jetzt einsatzfähig ist, siehe dazu insbesondere Kap. 8.

Um einen stabilen Einsatz von Elektra und vor allem auch eine längerfristige Systemunterstützung sicherzustellen, musste eine Basis für eine Kommerzialisierung gefunden werden. Daraus ergab sich schließlich eine Partnerschaft mit der Firma Sisis GmbH, die Mitte des Jahres 2000 den Elektra Broker übernommen hat, ihn weiterentwickelt und kommerziell vertreibt. Das zugehörige Vertragswerk wurde mit dem DFN abgestimmt. Derzeit gibt es mehrere Installationen und Testinstallationen, für Details siehe Kap. 8.

Schließlich sei noch erwähnt, dass Elektra eine wesentliche Rolle bei der Neustrukturierung des Bibliothekswesens an der Technischen Universität München spielen soll. Insbesondere sollen durch die Zentralisierung und gleichzeitige Beschleunigung der Zeitschriftenversorgung erhebliche Finanzmittel eingespart werden.

1.1 Projektbeschreibung

Realisierung eines integrierten intelligenten Informationssystems im Internet unter Verwendung des Kommunikationsprotokolls Z39.50. Die angestrebten Funktionalitäten umfassen Archivierung, parallele Recherche sowie Dokumentlieferung, Knowledge-Broker und Schnittstellen zu externen Informationssystemen.

1.2 Ziele von Elektra II

Ausgehend von den bereits existierenden Vorarbeiten des Projektes Elektra I war das Ziel des Nachfolgeprojektes die Entwicklung eines Brokers, der den Zugang zu Fachinformation über das WIN erleichtern und verbessern soll. Folgende Aufgaben waren dabei zu berücksichtigen:

Nachweis: Elektra I weist nur Aufsätze nach, die in Elektra-Servern der teilnehmenden Bibliotheken (Content-Provider) vorgehalten werden. Elektra II soll auch Aufsätze in anderen Servern nachweisen, sofern diese eine Z39.50- oder Harvest-Schnittstelle besitzen, insbesondere auch in dem Server der Firma Swets & Zeitlinger. So werden zwei Vorteile erzielt: Weniger Erfassungsarbeit bei den Partner-Bibliotheken für diejenigen Zeitschriften, die ausreichend in anderen Servern nachgewiesen werden, und gleichzeitig eine insgesamt größere Literaturmenge, die nachgewiesen werden kann.

Retrieval: Das Retrieval über heterogene Server erfolgt durch eine automatische Umsetzung der Retrieval-Syntax der jeweiligen Clients (Elektra, Z39.50,..) in die Retrieval Sprache des Ziel-Servers. Dies ist Aufgabe des zu entwickelnden Elektra-Brokers und erfolgt unter Verwendung einer einheitlichen, gemeinsamen Zwischendarstellung für Anfragen.

Browsing: Treffer-Mengen in Servern verschiedener Content-Provider werden unterschiedlich in Formaten und Umfang angezeigt. So zeigen z.B. die Verbundsysteme der Bibliotheken nur bibliographische Datensätze auf Heftebene an. Fachinformationssysteme dagegen zeigen bibliographische Datensätze in BibTex und möglicherweise den Abstract als ASCII Text. Elektra-Server zeigen zusätzlich die Inhaltsverzeichnisse von Heften und die ersten Seiten von Aufsätzen im Faksimile. Diese verschiedenen Anzeigen sind durch die inhaltlich sehr unterschiedlichen Erschließungstiefen einerseits, andererseits aber auch durch die ursprünglich zum Einsatz kommende Rechnertechnologie (ASCII-Terminals bei Verbundsystemen) bedingt. Der Benutzer erwartet aber, dass diese unterschiedlichen Informationen aus heterogenen Servern trotzdem einheitlich und verständlich angezeigt werden. Dies ist ebenfalls Aufgabe des Brokers . Hinzu kommt, dass Server auf Anfragen unterschiedlich schnell reagieren, der Benutzer aber nicht auf den letzten Server warten will, sondern ankommende Ergebnisse möglichst schnell sehen möchte. Deshalb muss der Broker die Ergebnisse zwischen Client und Server asynchron wandeln und vermitteln. Die Bestellung von Aufsätzen erfordert für die verschiedenen Content-Provider unterschiedliche Bestellvorgänge, die dem Benutzer aber nicht zuzumuten sind. Deshalb muss der Broker auch hier vermitteln, nach außen ein möglichst einheitliches Bestellverfahren anbieten und

intern auf die Erfordernisse der jeweiligen Server abbilden.
Die Lieferung von Dokumenten erfolgt bei Elektra I online im Postscript-Format, andere Server liefern z.B. HTML oder Latex. Subito liefert zusätzlich per Fax, FTP in Kombination mit E-Mail oder sogar gelber Post. Manche dieser Lieferungen können durch den Elektra-Broker weitergeleitet werden, bei anderen werden nur Statusmeldungen weitergegeben, manche gehen vollständig am Elektra-Broker vorbei. Der Elektra-Broker war so zu entwickeln, dass er mit diesen unterschiedlichen Lieferverfahren umgehen kann.

2 Systemaufbau von Elektra II

Elektra II ist ein sehr komplexes Software System, das aus den drei Subsystemen *User Administration, Search & Order, Archivierer* besteht. Jedes dieser Subsysteme ist selbst als Client/Server System konzipiert und intern sorgfältig modularisiert. Der Benutzerkreis dieser drei Subsysteme ist ebenfalls verschieden:

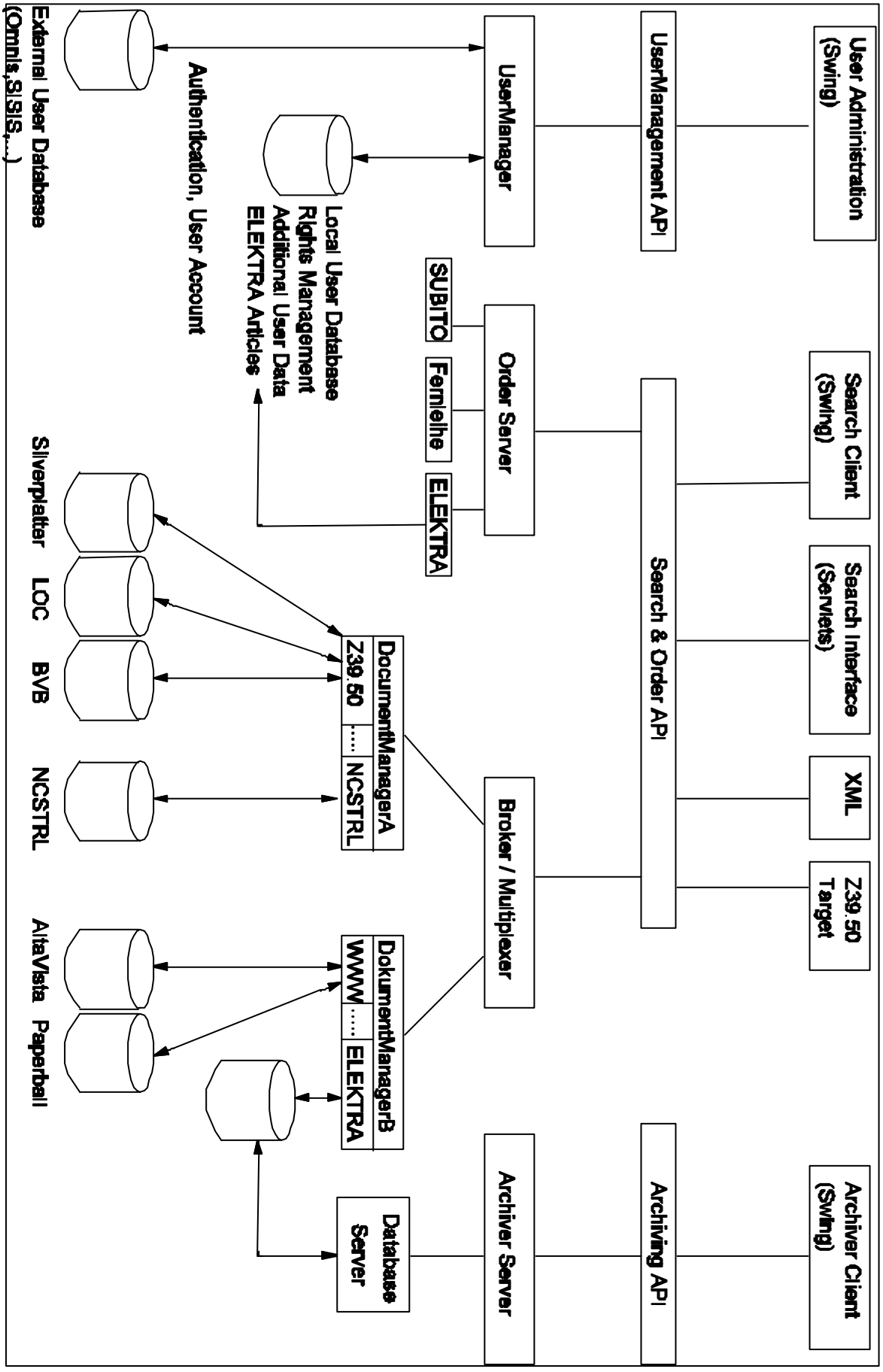
- Die User Administration wird hauptsächlich von der Bibliotheksleitung verwendet.
- Search & Order ist für den Bibliotheksbenutzer, also den Leser und Forscher.
- Der Archivierer wird vom Fachpersonal der Bibliothek verwendet, um bibliographische Informationen zu erfassen und Bestellungen abzuarbeiten und für die Lieferung bereitzustellen.

In den folgenden Kapiteln werden die Komponenten mit ihrer Architektur und ihren technischen Schnittstellen beschrieben.

2.1 Komponenten

Name	Aufgaben
Broker	• Recherche in einer Vielzahl von Datenquellen mit einer einheitlichen Anfragesprache • Konvertierung der Suchergebnisse in ein gemeinsames Metadatenformat (Dublin Core) • Zugriff auf spezielle Ergebnisdarstellungen eines Target-Systems (MARC,MAB,..)
Benutzerverwaltung	• Registrierung/Verwalten von Benutzerdaten in lokalen Datenbanken, Authentifizierung über externe Bibliothekssysteme (Sisis/Omnis) • Verwaltung eines lokalen und externen Benutzerkontos mit der Möglichkeit zur Verlängerung von Medien und Stornierung von Vormerkungen (Sisis/Omnis) • Speichern von Benutzerprofilen
Benutzerschnittstellen	• Servletbasierte Schnittstelle zu Broker und Benutzerverwaltung. • XML-Schnittstelle zur Brokerfunktionalität • Z39.50-Target Zugang für Broker
Bestellkomponenten	• Order Server: Verwaltet interne wie externe Bestellungen. Verschicken von Bestellungen an externe Systeme (derzeit nur die DOD-Station von Subito-lieferanten), parsen von Statusmeldungen und Benachrichtigen der Benutzer. • Archivierer: Katalogisieren von Dokumenten mit bibliographischen Daten und Bildern (Inhaltsverzeichnisse, Abstracts von Artikeln). • Batcharchivierung von Fremddaten (Swets&Zeitlinger TOC) • Bearbeiten von internen Elektra Bestellungen.

2.2 Systemarchitektur (Übersicht)



2.3 Kommunikation in Elektra II

Die Kommunikation der einzelnen Elektra-Komponenten erfolgt mittels serialisierter Objekte/Java-Klassen über das RMI-Protokoll. Die Datenübertragung erfolgt je nach Anforderung asynchron (durch einen Callback-Mechanismus) oder synchron. Für jede Client-Anfrage antwortet ein Server mit genau einem Resultat. Die Anfragen/Antworten von Broker und Benutzerverwaltung sind dabei alle von gemeinsamen Oberklassen abgeleitet, welche folgende Informationen beinhalten:

com.sisis.elektra.common.Request: (Basisklasse aller Anfragen)

Parameter	Bedeutung
RequestName	Lesbarer Name des Requests
Databases	Pfade zu den einzelnen Targets(Server), an die die Anfrage gerichtet werden soll. Ein Pfad hat dabei folgendes Format: »Multiplexer1«:...:«MultiplexerN«@«Kürzel des Servers«
SequenceNumber	Sequenznummer. Wird mit dem entsprechenden Result zurückgeliefert.
Timeout	Maximale Wartezeit (in Sekunden) für ein Result. Kann ein Server dieses Limit nicht einhalten wird der Request verworfen.
AuthEntry	Authentifizierungsdaten des Clients (IP-Adresse, Rechte/Kollektionen)
Language	Sprachidentifikator (»de«, »en«, ..)

com.si sis.elektra.common.Result: (Basisklasse aller Antworten)

Parameter	Bedeutung
ResultName	Lesbarer Name des Results
SequenceNumber	Gespiegelte Sequenznummer aus Request
Duration	Bearbeitungszeit innerhalb des Server (in ms)
Servername	Lesbarer Name des Servers
Serverpath	Adresse des Targets innerhalb des Brokers (s.a. Request.Databases)
Status	**Status<0**: Fehler, **Status>0**: Warnung, **Status=0**: erfolgreiche Bearbeitung
Warnings	Warnungstexte in Request.Language
Errors	Fehlertexte in Request.Language

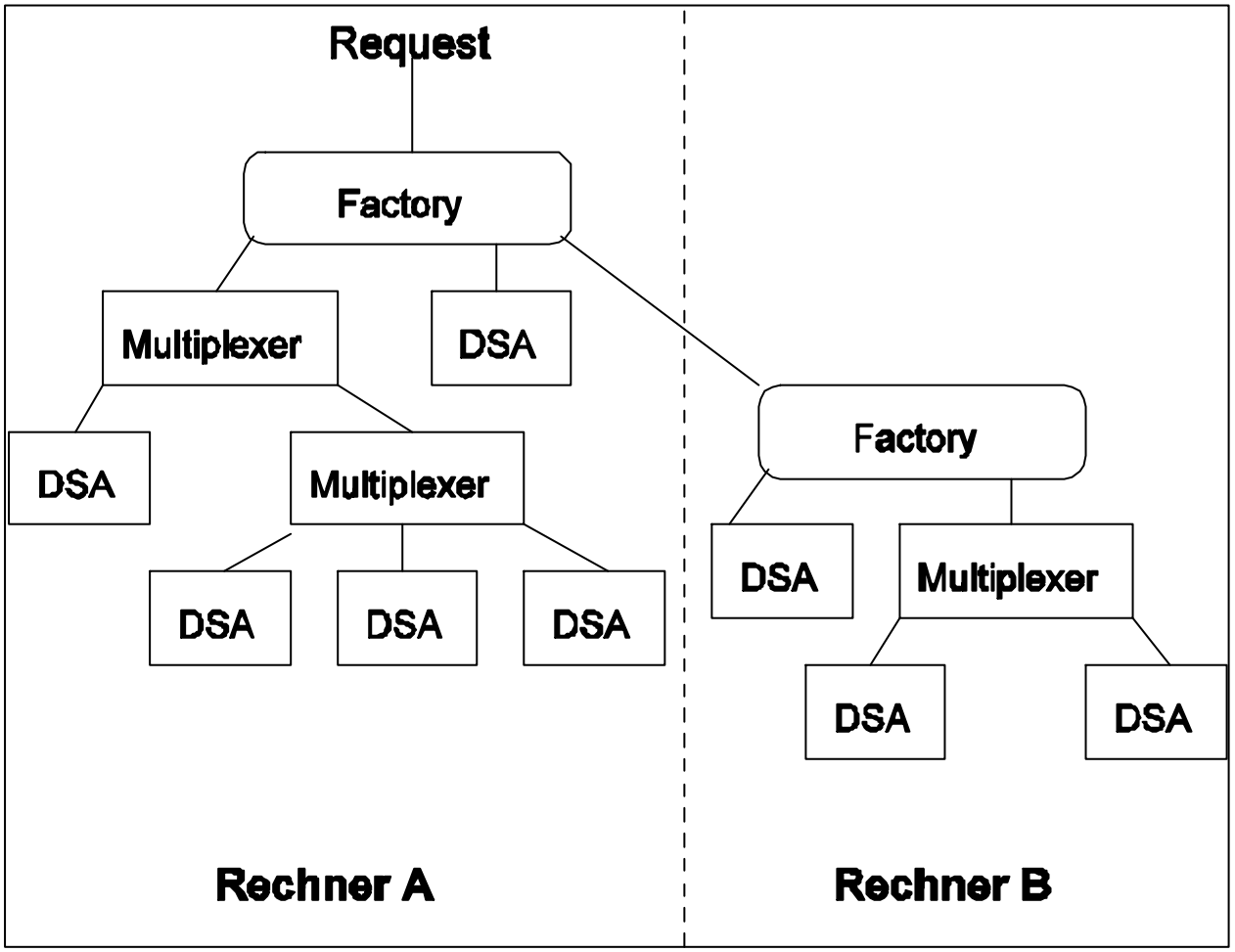
Kommunikationsknotenpunkte:

Protokoll-Beteiligte	Beschreibung
Server	DocumentServiceAgent (DSA) oder UserManager. Implementieren die Serverfunktionalität d.h. bearbeiten die eintreffenden Requests
Multiplexer	Verteilerprozesse. Ein Multiplexer verwaltet mehrere Server oder aber auch weitere Multiplexer, so dass eine hierarchische Verteilerstruktur entsteht. Eingehende Anfragen werden durch die verwalteten Multiplexer durchgereicht bis ein passender Server erreicht ist. Die Multiplexer können dabei auf verschiedenen Rechnern laufen um eine Lastverteilung zu ermöglichen.
LoadBalancer	Komponenten zum Lastausgleich für Server. Verwaltet mehrere Instanzen eines Servers (z.B.: LOC Z39.50-Target). Trifft ein Request ein, wird er an Hand eines modifizierten Round-Robin-Algorithmus an die am wenigsten beschäftigte Instanz weitergereicht.
Factory	Zentrale Komponente für Broker und Benutzerverwaltung. Wird beim Start als RMI-Objekt am Naming-Service (rmiregistry) angemeldet. Aufgabe: Starten/Verwalten von Servern, Multiplexern und LoadBalancer; Sammeln von Statistikinformationen. Vorteil: leichtere Administrierbarkeit, da für die Target-Systeme des Brokers nicht jeweils ein eigener Java-Server ausgeführt werden muß.

Beispiel einer Multiplexerhierarchie:

Ablauf einer asynchronen Dienstanforderung in Elektra:

1. Client sendet Request an zentrale Factory.
2. Die Pfadinformationen der Anfrage werden analysiert und über die entsprechenden Multiplexer an den gesuchten Server vermittelt.
3. Der Server bearbeitet die Anfrage und ruft eine Callback-Methode des Clients auf.



2.4 Logging

Für das Logging wird die vom Apache-Projekt verwaltete Log4j-API verwendet.
Lognachrichten werden per Default in eine Datei geschrieben (Das benutzen der Syslog-Facility ist alternativ möglich).
Das Verzeichnis der Log-Dateien liegt im Installationsverzeichnis (*$ELEKTRA_HOME*) unter *elektra/log/*. Folgende Ereignisse können festgehalten werden:

communication.log	Kommunikation mit externen Systemen (Usermanager: SLNP-Anfragen zu OPServer; Broker: Z39.50-Protokoll, SLNP-Protokoll)
elektra.log	Allgemeine Debug-Nachrichten
error.log	Alle aufgetretenen Java-Exceptions
protocol.log	Alle Elektra Protokollanfragen und Ergebnisse
servlets.log	Ereignisse aus Servlet/JSP GUI
session.log	Erzeugte und beendete Benutzersitzungen (mit statistischer Info)
sql.log	SQL-Statements an Lokale Benutzerdatenbank, OMNIS-Ausleihsystem und SQL-Recherchesysteme

2.5 Technische Daten

Programmiersprache	Java (100%) RMI (Remote Method Invocation) als Kommunikationsmechanismus
Umfang	ca. 150.000 LOC
Packages	59
Klassen	480
Platformen	Solaris, Linux
Externe Packages	ZedJava, Z39.50 (http://www.crossnet.uk) Log4j, Logging API (http://www.log4j.org) Java CUP, Parser (http://www.cs.princeton.edu/~appel/modern/java/CUP/)

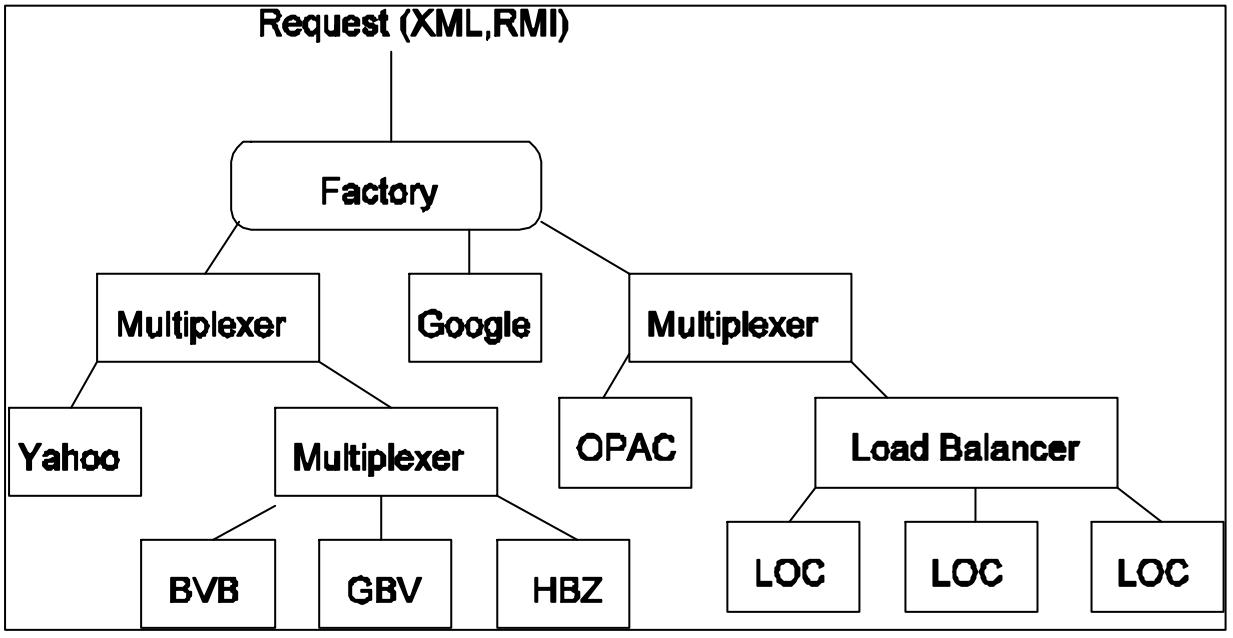
3 Detailliertere Beschreibung der Broker-Komponente

Wissenschaftliche Literatur wird heute über viele verschiedene Systeme zur Verfügung gestellt. Vor der Einführung des Internets arbeitete der Wissenschaftler überwiegend mit dem Angebot seiner lokalen Universitätsbibliothek, in Ausnahmefällen konnte er sich der Fernleihe bedienen und auf die Literatur anderer Bibliotheken zugreifen. Diese Arbeitsweise hat sich grundlegend verändert. Heute braucht man Zugang zu vielen, meist sogar sehr heterogenen Literaturquellen. Ziel von Elektra II war, diese Heterogenität der wissenschaftlichen Literatur zu überwinden und sie wesentlich einfacher und schneller zugänglich zu machen. Das zentrale Instrument dabei ist der Elektra Broker, der in den folgenden Abschnitten genauer beschrieben wird.

3.1 Aufgaben

Parallele Suche in den Zielsystemen. Anfragen werden von einer gemeinsamen Anfragesprache in die Syntax des Zielsystems überführt (soweit dies semantisch möglich ist). Die Kommunikation zwischen Client und Server kann sowohl synchron als auch asynchron erfolgen. D.h. Teilergebnisse werden sofort an den Client zurückgeliefert. Der Broker unterstützt multiple Protokolle (Z3950, SLNP, Dienst4 (NCSTRL),ODBC) Zusätzlich zu einer gemeinsamen Ergebnisdarstellung wird versucht, dem Client möglichst viele Originalformate zugänglich zu machen.

3.2 Architektur



3.2.1 Beteiligte Komponenten

DocumentServiceAgent(DSA)	Wrapper für die einzelnen Target-Systeme. Verantwortlich für die

	Konvertierung der Anfragesprache und der Ergebnisdarstellungen. Protokollabhängig; implementiert die Schnittstelle zu den Zielsystemen. Anfragen werden in eine Warteschlange eingereiht und nach dem FIFO-Prinzip abgearbeitet. Die Ergebnisse werden je nach Anforderung synchron oder asynchron an den Client zurückgegeben.
LoadBalancer	Objekt zum Lastausgleich. Verwaltet mehrere DSA's für ein Target. Eingehende Requests werden nach einem Scheduling-Mechanismus an wenig ausgelastete DSA's weitergeleitet. (Ausnahme: Eine Anforderung zur Anzeige von Einzeltreffern muss von dem DSA entgegengenommen werden, der die Ergebnismenge verwaltet.)
Multiplexer	Verteilerobjekt. Aufgabe: logische, hierarchische Verwaltung der DSA's, Verteilung des Brokers über mehrere Rechner. Anfragen an einen Multiplexer werden an die von diesem verwalteten DSA's bzw. weitere Multiplexer weitergereicht.
Factory (Broker)	Zentrale Komponente des Brokers. Startet bzw. beendet DSA, Multiplexer und LoadBalancer. Sammelt Statistikinformationen über die DSA's.

3.2.2 Funktionalität des Brokers

Request	Bedeutung
Database_Info	Information über die einzelnen Target-Systeme (Name, Bestand an Titeln, jährlicher Zuwachs, Zugangsbeschränkungen, Inhalt, Beschreibung, URL zum Target)
Search_Profile	Liefert unterstützte Operatoren und Suchfelder der Target-Systeme. Dient zur dynamischen Generierung einer Suchmaske zu einer Benutzerdefinierten Datenbankauswahl.
Search	Suche in Zieldatenbank --> Ergebnis Anzahl der Treffer.
Present	Liefert Ergebnisse einer vorhergehenden Suche in der geforderten Recordsyntax
CombinedSearch	Kombination aus Search- und Present-Request: holt die ersten x Records zu einer Anfrage
Scan	Liefert Auszug aus Wortliste eines suchbaren Attributes

3.3 Unterstützte Zielprotokolle

Derzeit können in Elektra Datenbanken folgender Zielprotokolle eingebunden werden:

Protokoll	Beschreibung	Verbreitung	Aufwand*
HTTP(GENERIC)	Suchmaschinen, Verzeichnisdienste, Verlage, Buchhändler	+++	++
Z39.50	Bibliothekssysteme, Forschungseinrichtungen	++	-
SLNP	Sisis Bibliothekssystem	+	-
HARVEST	Suchschnittstelle des Harvest Gatherers.	++	-
COMPASS	Netscape Compass Suchmaschine	+	-
NCSTRL	Sammlung von technischen Reports	-	-
OMNIS3, OMNIS4, TRANSBASE, SQL	Proprietäre Informationssysteme der TU-München (Omnis, Transbase). Anbindung über JDBC/SQL.	-	++

*Aufwand um neue Systeme zu integrieren

3.3.1 Hinzufügen neuer Datenquellen

Der Aufwand ein neues Target-System in Elektra zu integrieren kann in drei Klassen eingeteilt werden:

- **Protokoll wird bereits angeboten (Ausnahme HTTP):** -> nur Konfiguration notwendig (Technische Parameter, Abbildung der Attribute) Aufwand: gering

- **Protokoll wird nicht unterstützt** -> Neuen DSA implementieren (Parser für Anfragesprache, integration der Kommunikationsmechanismen,Bearbeitung der zur Verfügung stehenden Ergebnisformate) Aufwand (Beispiele): 1 Woche(SISIS) - 6 Wochen(Z39.50)

- **HTTP-Targets:** Wrapper + Konverter "HTML -> Dublin Core" muss implementiert werden (Java). Sofern es sich nicht um eine sehr exotische Quelle handelt ist die Übersetzung der Anfrage konfigurierbar. Der Aufwand ist sehr unterschiedlich (3 Stunden - 3 Tage), wobei etwa 80% der Quellen in den Bereich von etwa 3 Stunden fallen. Problematisch: HTML - Seiten können sich ändern
  ⇨ Konverter müssen laufend aktualisiert werden.

3.3.2 Bislang integrierte Datenbanken zu den einzelnen Protokollen

http	Allgemeine Suchmaschinen: Altavista, Google, Infoseek, Lycos, etc. Verlage: ACM, Springer Zeitungsartikel: Paperball Buchhändler: JF. Lehmanns, Amazon.com, Amazon.de Usenet: Dejanews Sonstige: IBM-Patent Research, ResearchIndex (wissenschaftliche Beiträge), ...
Z39.50	Bayerischer Verbundkatalog, Library of Congress, CD-ROM-Datenbanken(ERL), ...
SLNP	Bibliotheken: TU-München, Uni-München, Bayerische Staatsbibliothek
HARVEST	MathNet (www.mathnet.org,mathnet.preprints.org)
COMPASS	Compass Webserver der Universitäten Stuttgart und Köln
NCSTRL	www.ncstrl.org
SQL	Bibliotheken der TU-München mit Bibliographischen Daten, Volltexten und Bildern der Inhaltsverzeichnisse

3.4 Anfragesprache

Als Syntax zur broker-internen Anfragesprache wird eine CCL-konforme Grammatik verwendet. Diese entspricht von ihren Merkmalen im Wesentlichen der mit der Z39.50 darstellbaren RPN-Syntax und der Anfragesprache, wie sie bislang im Elektra I Projekt verwendet wurde.

Operatoren	Boolsche Operatoren: AND, OR, NOT
Proximity	Abstandsoperatoren für exakten (t1 !n t2) und maximalen (t1 %n t2) Abstand zweier Terme. Nur von einigen Zielsystemen bislang überhaupt angeboten.
Suchfelder	Use- Attribute des Z39.50 Attributesets BIB-1 (Erweitert um zusätzliche Attribute). Syntax: ATTR-<Nr. des Attributs>
Relationen	<, >, <=, >=, =, != erlaubt. Unterstützt wird von den Zielsystemen meistens nur »=«.
Resultset-Operator	RESULT-<Nr. des Resultsets>
Trunkierungszeichen	?
Klammerung	(,)
Phrasensuche	Aus mehreren Termen bestehende Anfragen werden automatisch als Phrase interpretiert.

Beispiele:

ATTR-4 java beans (Titel enthält die Phrase "java beans")
ATTR-1 bayer? (Personenname muss mit Bayer beginnen)
ATTR-4 corba %3 progr? (Die beiden Terme dürfen einen maximalen Abstand von 3 Worten haben)
ATTR-1 heine OR (ATTR-1 böll AND A TTR-4 clown?)

3.4.1 Anfragetransformationen

Kann eine Anfrage semantisch nicht korrekt in die Zielsprache übertragen werden, führt Elektra folgende Transformationen (deaktivierbar) durch:

Fehlendes Merkmal	Transformationen in ELEKTRA
Trunkierung	Suche nach Term ohne Trunkierung
Phrasensuche	Verknüpfung der Terme mit AND
Proximity-Operator	Konvertierung zu Phrase

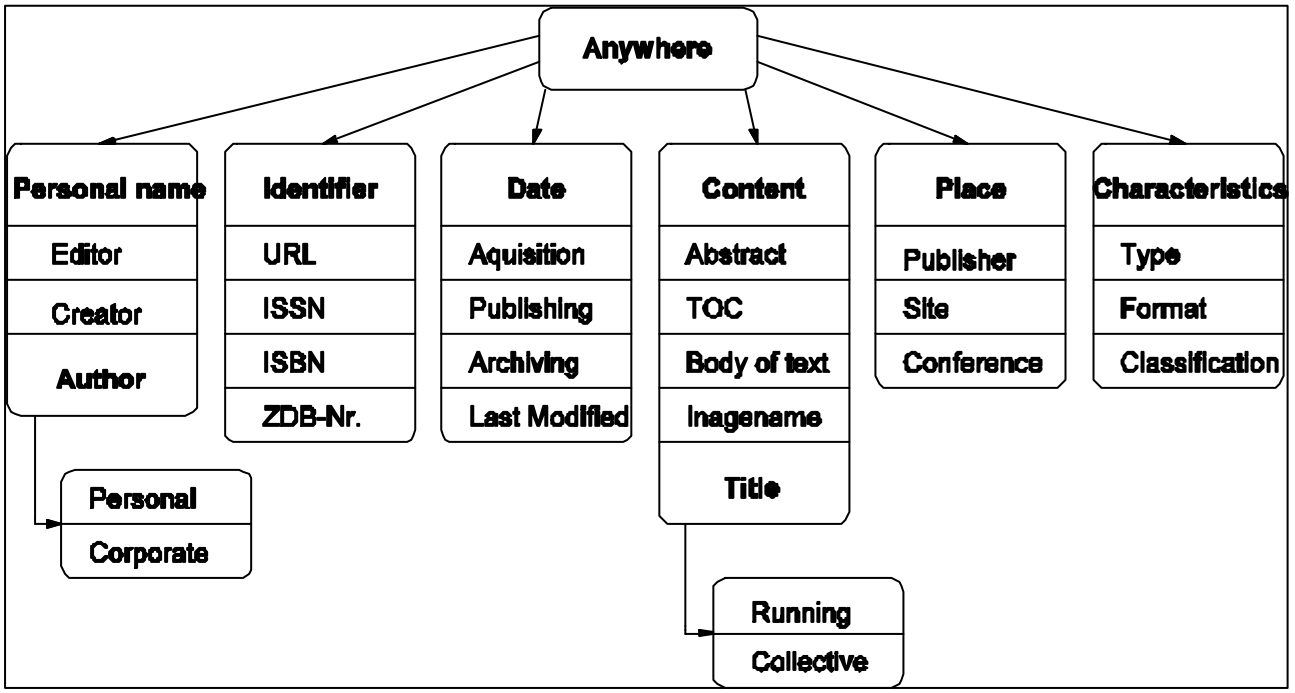
Wird ein Suchfeld vom Zielsystem nicht angeboten, wird die Anfrage zurückgewiesen.

3.4.2 Virtuelle Attribute

Z39-Bib-1 Attribute werden in der Regel 1:1 auf die Suchfelder des Retrievalsystems abgebildet. Zusätzlich ist auch eine hierarchische Strukturierung möglich.

Beispiel: Personal Name -> Editor
-> Author -> Author pers
-> Author conf

Besitzt ein Zielsystem nicht den Wurzelknoten als Suchfeld, wird mit einer Kombination der Subfelder gesucht. Diese Methode wird auch verwendet zum Kombinieren von Attributen (ISSN + ISBN --> Neues Feld).



3.4.3 Grammatik der gemeinsamen Anfragesprache

```
search_elements   ::=   search_elements bool_op element
                  |     element
                  ;

element                 ::=     CCL_OPEN search_elements CCL_CLOSE
                  |     search_terms
                  |     attributes relation value_term
                  |     attributes value_term
                  |     CCL_RESULTSET
                  ;
value_terms       ::=   value_terms bool_op value_term
                  |           value_term
                  ;
value_term              ::=     CCL_OPEN value_terms CCL_CLOSE
                  |           search_terms
                  |     range_term
                  ;
range_term        ::=   term range term
                  |           term range
                  |     range term
                  ;
search_terms      ::=   search_terms prox_op term
                  |           search_terms term
                  |           term
                  ;
attributes        ::=   attributes COL attribute
                  |           attribute
                  ;
range             ::=   CCL_RANGE
                  ;
relation          ::=   CCL_RELATION
                  ;
bool_op                 ::=     CCL_BOOLOP
                  ;
prox_op                 ::=     CCL_EPROXIMITY
                  |     CCL_PROXIMITY
                  ;
term              ::=   CCL_STRING
                  ;
attribute         ::=   CCL_ATTRIBUTE
                  ;
```

3.5 Ergebnisformate

Rechercheergebnisse werden in Elektra zunächst in ein gemeinsames Präsentationsformat (Dublin Core) zur Darstellung einer gemeinsamen Trefferliste umgewandelt. Da dieses Format nur den kleinsten gemeinsamen Nenner der von den

Target-Systemen angebotenen bibliographischen Daten darstellt, wird es dem Benutzer auch ermöglicht, ausgehend von der Einzeltrefferanzeige, zu den umfangreicheren Datensätzen zu gelangen. Derzeit werden von dem Elektra-Broker folgende Formate (Protokoll- und Targetabhängig) geliefert:

Von allen angeboten. Zum Aufbau einer gemeinsamen Trefferliste		
DUBLINCORE	Alle	15 Basiselemente (http://purl.oclc.org/dc/)
DESCRIPTION	Alle	Erweiterung von DC um Statusinfo (ausleihbar, vormerkbar), Vorhandensein weiterer Präsentationsformate, Multimediadaten(Bildern)
Teilweise unterstützte Formate		
UNIMARC, USMARC, MAB	Z39.50	Bibliographische Daten. Darstellung im Format: Identifikator-Inhalt
OPAC	Z39.50	Z39.50 – Exemplarinformationen
HOLDINGS	OMNIS3,SISIS	Exemplarinformationen
OMNIS	OMNIS3/4	Bibliographische Daten der Omnis-Systeme
SOIF	HARVEST, COMPASS	Metadaten im Format: Identifikator-Inhalt
HTML	GENERIC	HTML-Seite des Orginaldokuments
SISIS	SLNP	Katalogsatz aus Sisis OPAC
HIERARCHY	SLNP	Hierarchiesuche in Sisis (Bände oder übergeordnete Titel)
Binäre Formate (Blobs)		
1 (TEXT)	OMNIS, TRANSBASE	Inhaltsverzeichnisse
2 (IMAGE)	OMNIS3/4, TRANSBASE, GENERIC	GIF- und JPEG-Dateien. Die Bilder werden entweder direkt über das Elektra Protokoll übertragen oder nur als Referenz (URL) übergeben

4 Benutzerverwaltung

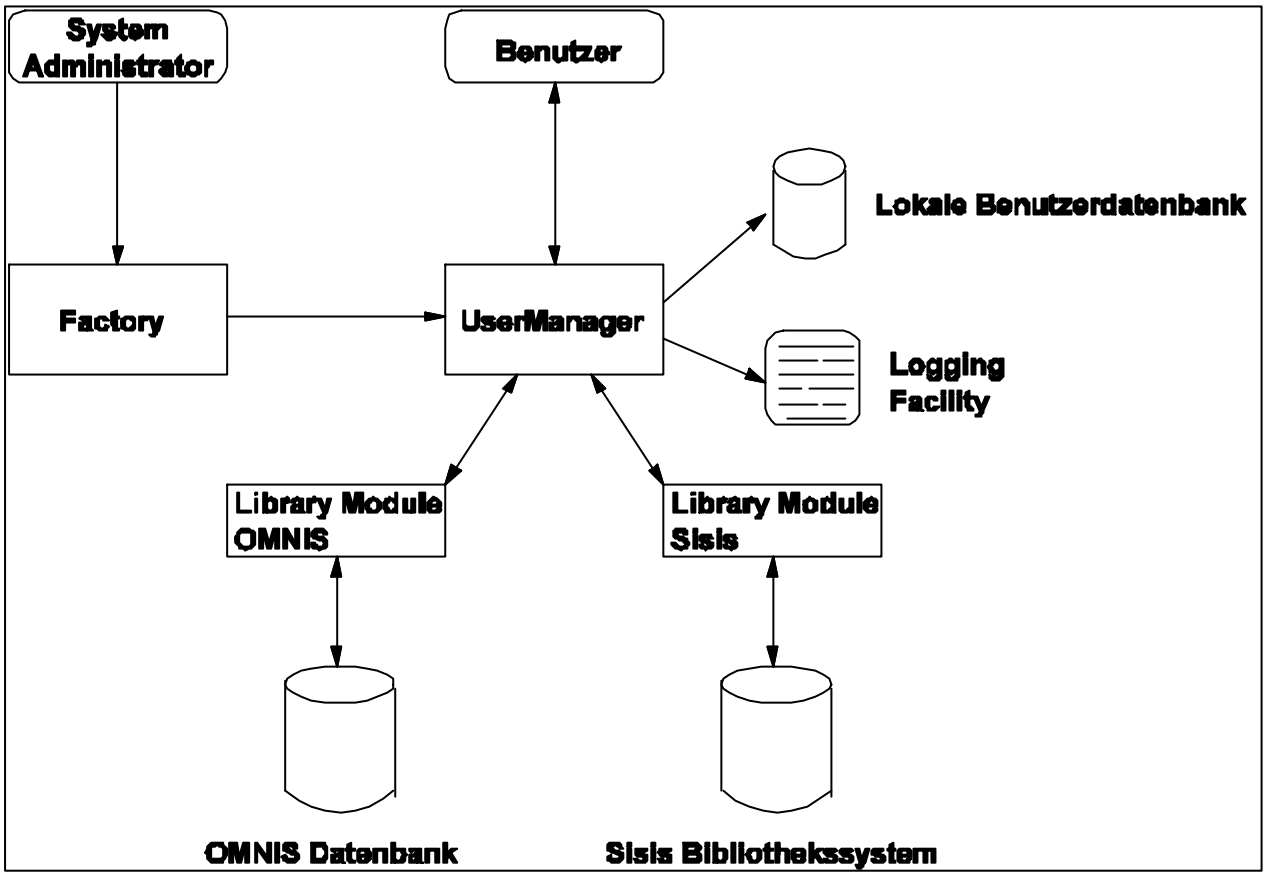
Ein modernes Bibliothekssystem muss viele Benutzerklassen unterscheiden – studentische Nutzer, Angestellte der Universität, Fachpersonal der Bibliothek etc. - und ihnen Zugriff mit unterschiedlichen Rechten gewähren können. Dazu ist eine ausgefeilte Benutzerverwaltung erforderlich, die sich nur partiell auf die in Betriebssystemen und Datenbanksystemen angebotene Benutzerverwaltung abstützen kann. In Elektra II wurde deshalb in enger Abstimmung mit den Anforderungen der Bibliotheksleitung eine differenzierte Benutzerverwaltung realisiert.

4.1 Features

Benutzerverwaltung	• Aufnahme von Benutzerdaten (durch den Administrator oder durch Selbstregistrierung der Benutzer) • Suche nach Benutzern • Ändern, Entfernen von registrierten Benutzern
Externe Authentifizierung	• Anbindung an Sisis/Omnis-Systeme. Die technische Anbindung ist dabei Modular, so dass weitere Authentifizierungssysteme ohne großen Aufwand einzubinden sind. • Datenübernahme aus Fremddatenbank (Zur Vermeidung von Inkonsistenzen bei jeder neuen Benutzersitzung überschreibbar)
Rechteverwaltung	• Zuweisen von Rechten (Kollektionen) an Benutzergruppen, die den Zugriff auf Ressourcen in Elektra gewähren
Kontoverwaltung	• Anzeige eines externen Benutzerkontos (Sisis/Omnis) • Gesamtkontoverlängerungen, Einzelverlängerungen (Sisis) • Stornierung von Vormerkungen (Sisis) • Registrieren von Bestellungen in lokalem Benutzerkonto
Benutzereingabemaske	• Konfigurierbares Layout (Neue Felder können integriert werden ohne die Programmlogik verändern zu müssen)

4.2 Systemaufbau

Architektur:



4.2.1 Beteiligte Komponenten / Benutzer:

Beteiligte	Aufgabe
Factory	Ausführbares Programm; Startet die Server (UserManager) und sammelt statistische Daten. Zur Lastverteilung kann die Factory mehrerer Instanzen der Server verwalten.
UserManager	Stellt die Funktionalität der Benutzerverwaltung über RMI zur Verfügung. Der Zugriff erfolgt entweder über das ELEKTRA-Protokoll oder eine RMI-Schnittstelle.
Library Module	Wrapper für externe Bibliothekssysteme. Aufgaben im Einzelnen: • Authentifizierung (Prüfen von Benutzernummer, Passwort, Ermitteln der Benutzergruppe) • Import von Benutzerdaten • Extraktion der Kontoinformation eines Benutzers • Gesamtkontoverlängerung • Einzelverlängerung • Stornierung einer Vormerkung
External Library System	Ausführen der Aktionen des jeweiligen Library Moduls
Lokale Benutzerdatenbank	Speichern von:

Beteiligte	Aufgabe
	Benutzerdaten Lokaler Kontoinformationen (Bestellungen) Benutzergruppen Zugriffsrechten Datenbankkonfigurationen einzelner Benutzer
Logging Facility	Loggingmechanismus (log4j) Kennt die Kategorien: SQL: Logging der SQL-Statements der lokalen Benutzerdatenbank ECOM: Logging der Kommunikation mit externem Bibliothekssystem (SLNP,OMNIS) ERROR: ELEKTRA Fehlermeldungen / Exceptions
System Administrator	Startet die Factory
Benutzer	Stellt Anfragen an den UserManager

4.3 Authentifizierung

Bei der Authentifizierung eines Benutzers werden zwei Benutzerklassen unterschieden:

Interne Benutzer: Benutzer,deren Daten sich nur in der lokalen Benutzerdatenbank befinden. D.h. die Benutzerdaten wurden vom Administrator aufgenommen oder der Benutzer hat sich Selbst registriert.

Externe Benutzer: Benutzerdaten liegen in einem Fremdsystem (Sisis oder Omnis). Teile der Benutzerdaten können in die lokale Elektra-Benutzerdatenbank übernommen werden (Caching für schnelleren Zugriff)

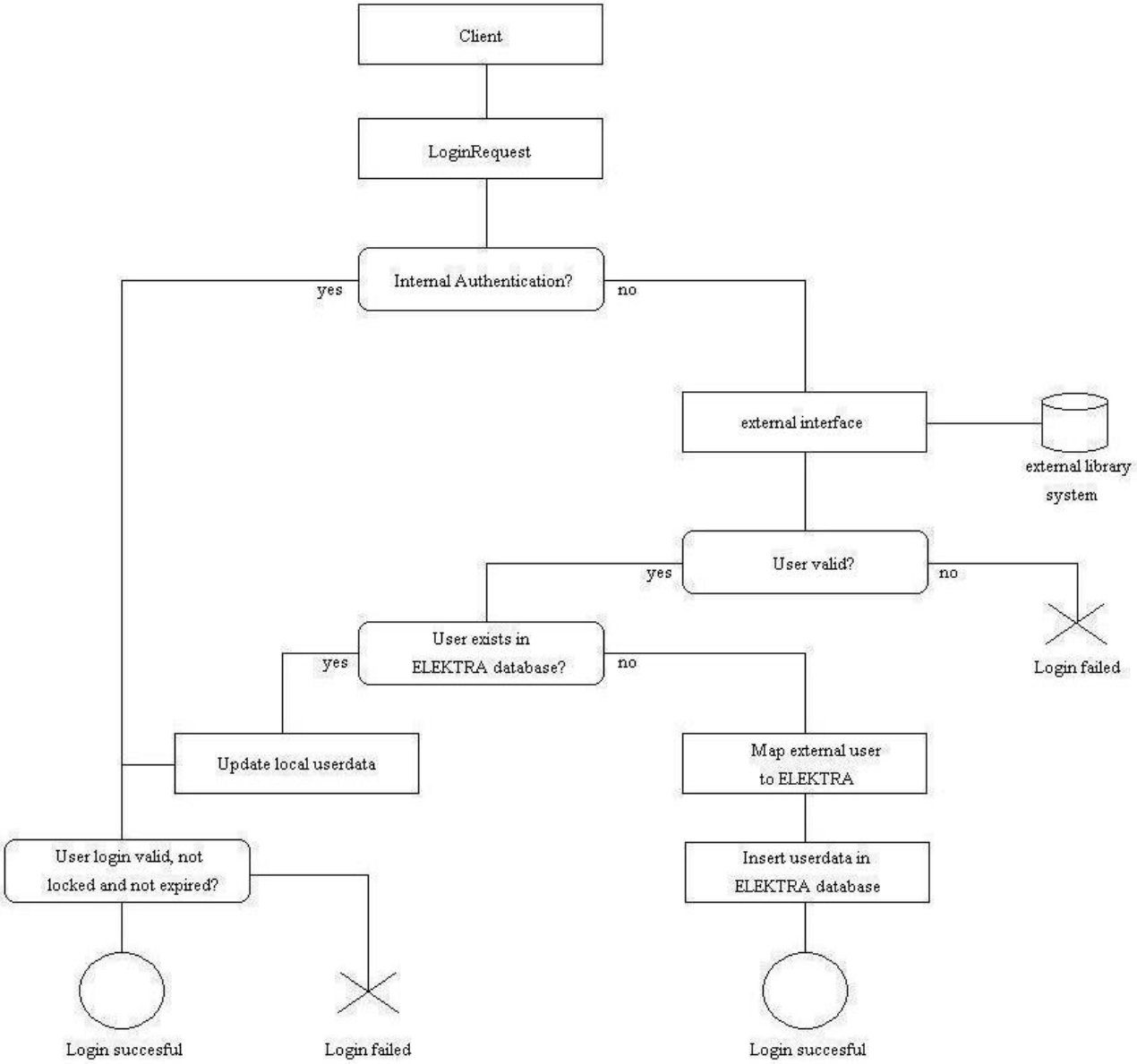
Authentifizierungsprozess für interne Benutzer:

- Prüfen von Benutzernummer (e_elektralogin) und Passwort (e_elektrapassword)
- Prüfen der Kontobefristung (e_expirationdate)
- Prüfen ob Kontosperre aktiviert (e_locked)
- Falls Tests erfolgreich:

* Setze e_lastlogindate und e_lastloginplace
* Ermittle Berechtigungen (Kollektionen) des Benutzers an Hand seiner Benutzergruppe

Authentifizierung für externe Benutzer:

- Prüfen der Gültigkeit des Benutzers an Fremdsystem
- Falls Tests erfolgreich: extrahiere Benutzerdaten
- Ist es der erste Loginversuch in Elektra:
  Füge die Benuterattribute in die lokale Benutzerdatenbank ein
- Ist ein Benutzer mit gleicher Kennung bereits in Elektra registriert:
  Ersetze alle Attribute in der lokalen Benutzerdatenbank, für die das »updatable-Flag« gesetzt ist


Client
LoginRequest
Internal Authentication?
yes
no
external interface
external library system
User valid?
yes
no
Login failed
User exists in ELEKTRA database?
yes
no
Update local userdata
Map external user to ELEKTRA
User login valid, not locked and not expired?
Insert userdata in ELEKTRA database
Login succesful
Login failed
Login succesful

4.4 Zugriffsberechtigungen IP-Adressen / Kollektionen

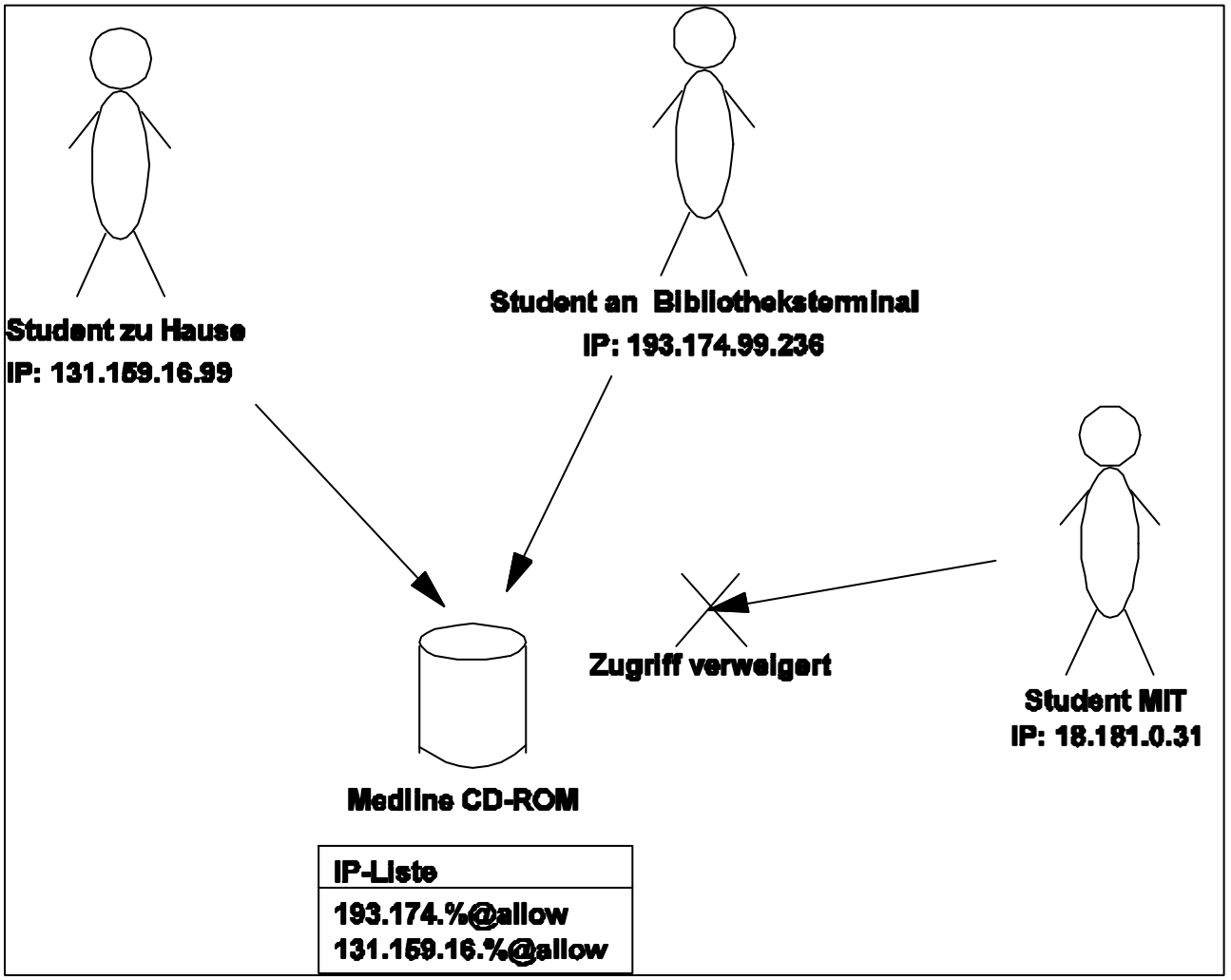
Der Zugriff eines Benutzers auf Target-Systeme kann mit zwei Methoden eingeschränkt werden.

4.4.1 Zugriffsbeschränkung durch IP-Adressen

Für jedes Target können in der Konfiguration IP-Schemata definiert werden, die folgender Syntax genügen müssen: IP-Schema@[allow | deny]. Als IP-Schema sind folgende Möglichkeiten erlaubt:

- IP-Netze der Klasse A : z.B.: 131.%
- IP-Netze der Klasse B : z.B.: 131.31.%
- IP-Netze der Klasse C : z.B.: 131.31.11.%
- Einzelne Rechner : z.B.: 131.31.11.200
- Alle Rechner : all

Mit »allow« und »deny« können die Berechtigungen noch weiter gesteuert werden. Wobei »deny« immer Priorität vor »allow« hat.



Szenarien:

Gültige Szenarien	IP-Definitionen	Bedeutung
Keine Prüfung der IP-Adresse	Keine Einträge oder all@allow	Sind in der Target-Konfiguration keine IP-Adressen definiert oder nur der Eintrag all@allow, ist eine Überprüfung der Berechtigung für einen Client immer erfolgreich
Zugriff nur für eine Domain	193.31.%@allow	Zugriff ist nur für die Domain 193.31 erlaubt anfragen von Rechnern mit unterschiedlichem IP-Präfix werden zurückgewiesen
Explizites verweigern für eine Domain	all@allow 193.%@deny	Zugriff ist für alle Rechner außerhalb der 193 Domain erlaubt
Einzelne Rechner einer Domain ausschließen	193.%@allow 193.31.11.200@deny	Nur Rechner der 193 Domain haben Zugriff. Mit Ausnahme von Rechner 193.31.11.200

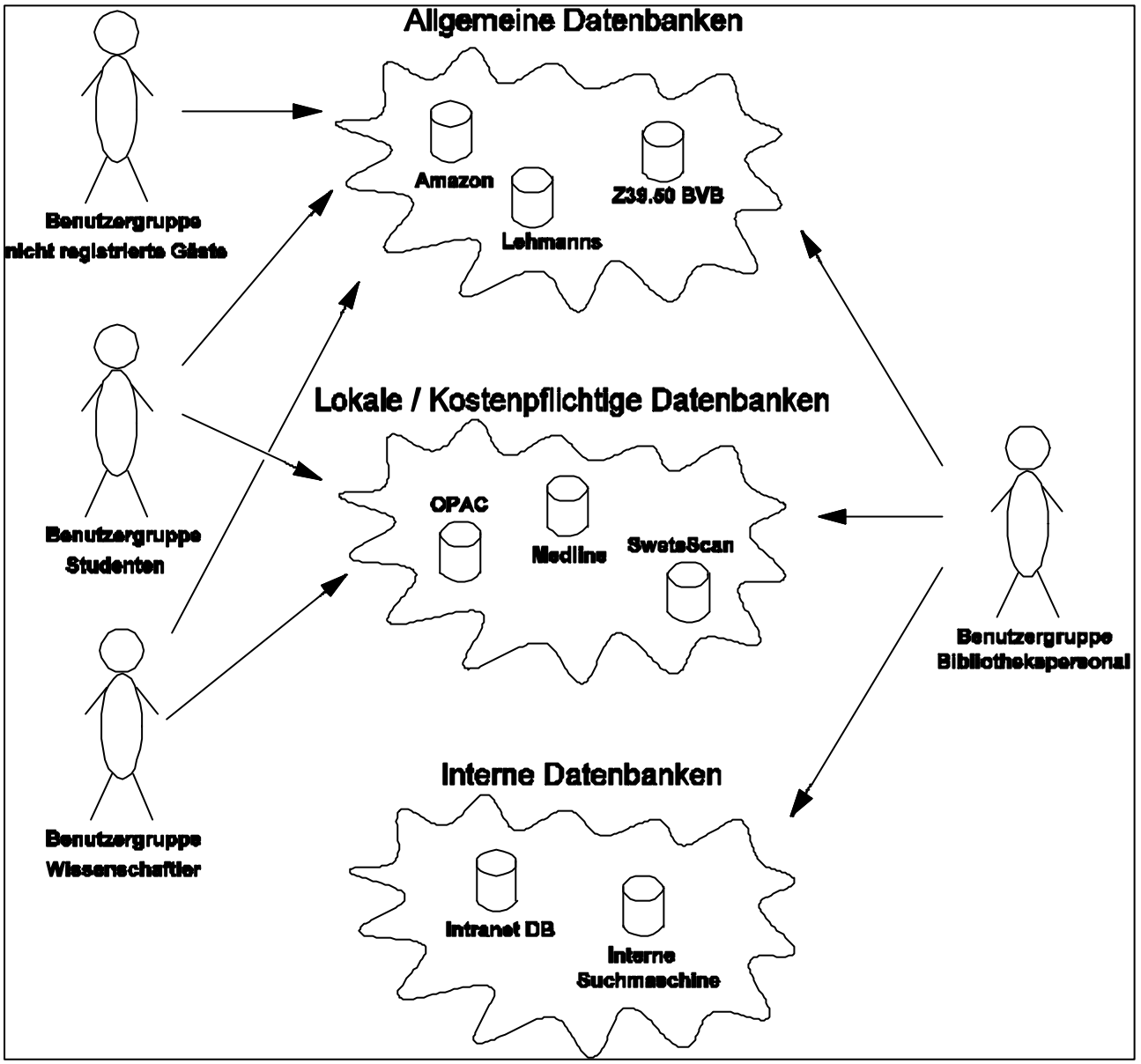
Nicht erlaubte Szenarien	IP-Definitionen	Bedeutung
Zugriff für eine Subdomain erlauben	193.31.%@deny 193.31.11.%@allow	Durch die Priorisierung von »deny« werden alle Zugriffe der 193.31 Domain zurückgewiesen. Alternative: Nur 193.31.11.%@allow verwenden.

4.4.2 Zugriffsbeschränkung durch Kollektionen

Mit Hilfe der in Elektra integrierten Benutzerverwaltung ist eine Vergabe von Zugriffsrechten auf Basis der Benutzergruppen möglich. Die Rechte werden dabei nicht direkt an die Benutzergruppen vergeben, sondern nur über eine weitere Indirektionsstufe in Form von Kollektionen. Eine Kollektion umfasst eine Zusammenstellung von Target-Datenbanken. Durch Zuweisen einer Kollektion an eine Benutzergruppe erhält diese die Berechtigung für alle darin definierten Targets.

Szenario:

Bibliothek XY	
Benutzergruppen	Gäste, Studenten, Wissenschaftler, Bibliotheksmitarbeiter
Datenbanken	OPAC, Interne Datenbanken (Intranet, interne Suchmaschine), CD-ROM Datenbanken (Medline), Suchmaschinen des Buchhandels (Amazon, Lehmanns), Verbundbibliotheken (BVB) und Datenbanken mit beschränktem Zugriff (SwetsScan)
Zugriffspolitik	Mitarbeiter der Bibliothek haben Zugang zu allen Target-Systemen Gäste dürfen nur in öffentlich zugänglichen Datenbanken recherchieren. Studenten und Wissenschaftler dürfen in allen Datenbanken recherchieren, mit Außnahme der bibliotheksinternen Datenbanken.



Lösung:

Modellierung derKollektionen:

Kollektion	Datenbanken
Allgemeine Datenbanken	Amazon, Lehmanns, BVB
Lokale / Kostenpflichtige Datenbanken	OPAC, Medline, SwetsScan
Interne Datenbanken	Intranet DB, interne Suchmaschine
Vorgehensweise: in der Target-Konfiguration der jeweiligen Datenbank wird die ID der Kollektion eingetragen	

Zuordnung der Kollektionen an Benutzergruppen:

Benutzergruppe	Kollektionen
Gäste	Allgemeine Datenbanken
Studenten	Allgemeine Datenbanken Lokale/Kostenpflichtige Datenbanken
Wissenschaftler	Allgemeine Datenbanken Lokale/Kostenpflichtige Datenbanken
Bibliothekspersonal	Allgemeine Datenbanken Lokale/Kostenpflichtige Datenbanken Interne Datenbanken
Vorgehensweise: Definieren der Kollektionen in der Benutzerverwaltung für jede Benutzergruppe	

4.4.3 Kombination von Zugriffsbeschränkungen

Befinden sich in einer Target-Definition sowohl Einträge für IP-Schemata als auch für Kollektionen, so haben die Berechtigungen durch Kollektionen höhere Priorität.

Beispiel: Benutzer X authentifiziert sich in ELEKTRA. Es wird die Benutzergruppe »Studenten« ermittelt, als IP-Adresse wird die IP 193.31.11.200 festgestellt. Für die Datenbank Medline sind Zugriffe außerhalb der Domain 194.174 verweigert. Zugriffsberechtigt sind die Benutzergruppen : Studenten, Wissenschaftler und Bibliothekspersonal. D.h. obwohl der Benutzer aus einem gesperrten IP-Bereich auf die Datenbank zugreift ist eine Recherche auf Grund seiner Gruppenzugehörigkeit erlaubt.

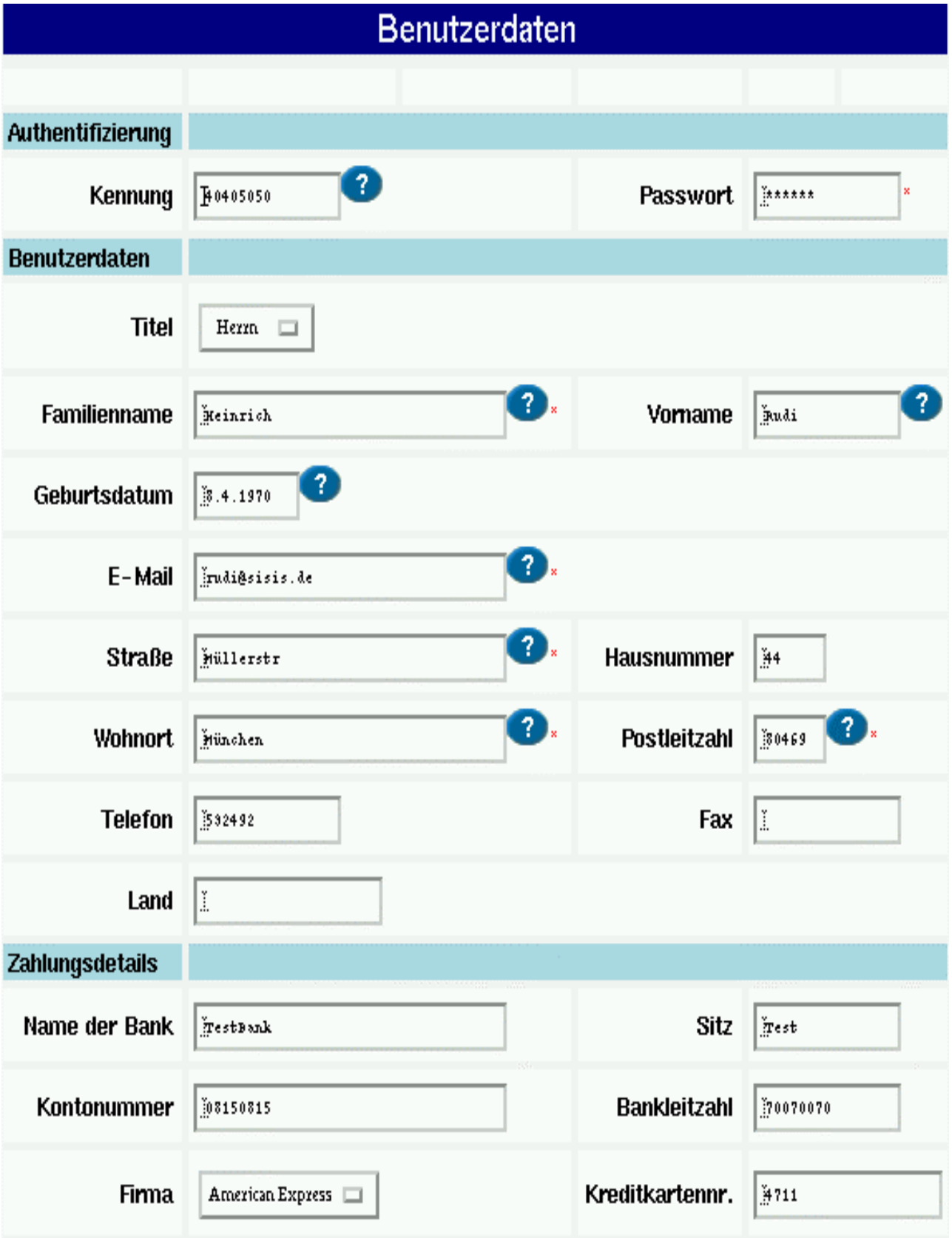
4.5 Benutzerdaten-Eingabe

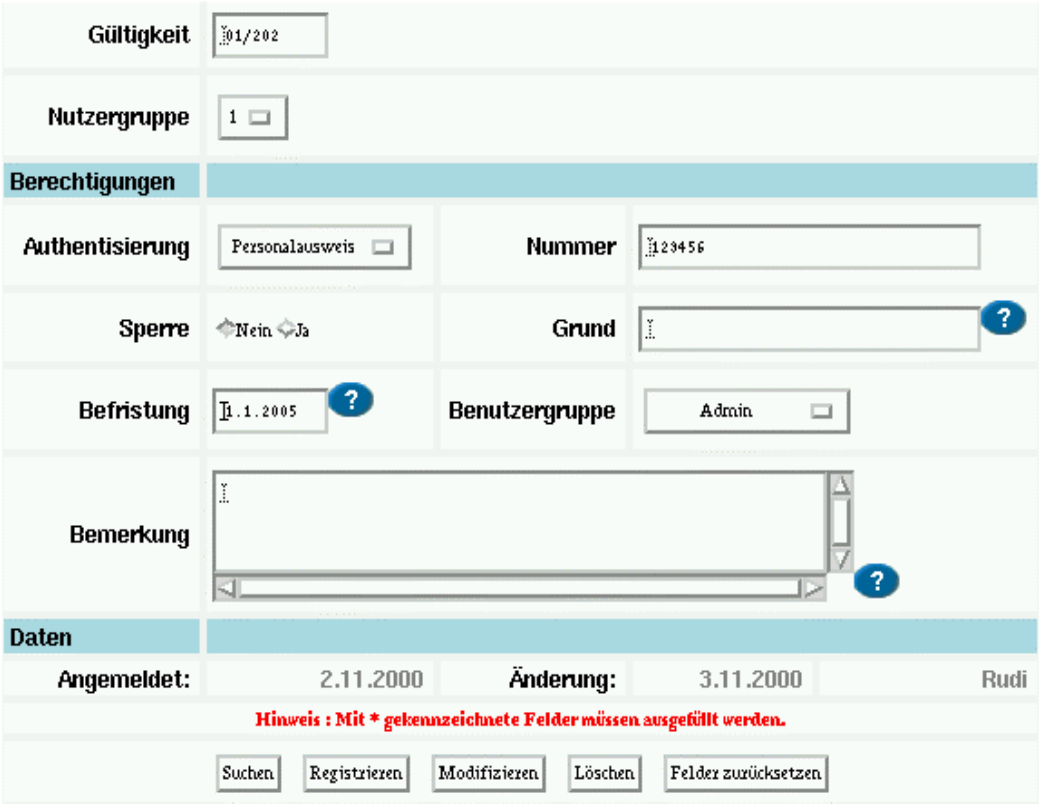
Mögliche Aktionen;

Suchen	Suche in mehreren Feldern durch Betätigen des Such-Buttons oder Suche innerhalb eines, als suchbar konfigurierten, Feldes durch klicken des Symbols hinter dem Eingabefeld. Suchterme werden automatisch rechtstrunkiert. Linkstrunkierung (mit Zeichen: %) und Wildcards sind erlaubt (Zeichen:).
Speichern	Existierende Benutzerdaten können modifiziert und neue Benutzer in ELEKTRA registriert werden
Löschen	Entfernt den aktuell angezeigten Benutzer. Die Daten werden dabei nicht aus der Datenbank entfernt, sondern nur als gelöscht markiert (Feld e_deleted in Tabelle e_usertable auf »1« gesetzt), so dass dieser Benutzer bei weiteren Suchen nicht mehr gefunden wird.
Felder Zurücksetzen	Löscht das Eingabefenster

Benutzer können in der Benutzerdatenanzeige ihre persönlichen Daten modifizieren, sofern für entsprechenden Felder in der Konfiguration das Feld aktiviert ist.
Eingabefelder, die von einer externen Benutzerverwaltung übernommen werden, können diese Änderungen beim nächsten Login des Benutzers überschreiben falls das update-Flag für das Eingabefeld gesetzt ist.

Beispiel: Eingabemaske





4.6 Benutzerdatenbank

4.6.1 Tabellen

Tabelle	Inhalt
e_usertable	Benutzerinformationen; Eingefügt durch Registrierung oder aus externem Bibliothekssystem übernommen. Wurde ein Benutzer in der Benutzerverwaltung versehentlich entfernt kann er durch Setzen des Feldes e_deleted auf »0« wieder sichtbar gemacht werden.
e_services	Lokale Kontoinformationen
e_objects	Allgemeine Ablage für Java-Klassen. Genutzt z.B. zur persistenten Aufbewahrung von Datenbankzusammenstellungen einzelner Benutzer
e_groups	ELEKTRA Benutzergruppen
e_rights	Kollektionen
e_grouprights	Zuordnung von Kollektionen zu Benutzergruppen
e_usergroup	NICHT VERWENDET IN ELEKTRA V1.0
e_userrights	NICHT VERWENDET IN ELEKTRA V1.0
e_invalidlogins	Festhalten fehlerhafter Loginversuche
e_elektratable	Hilfstabelle für SQL-Statements

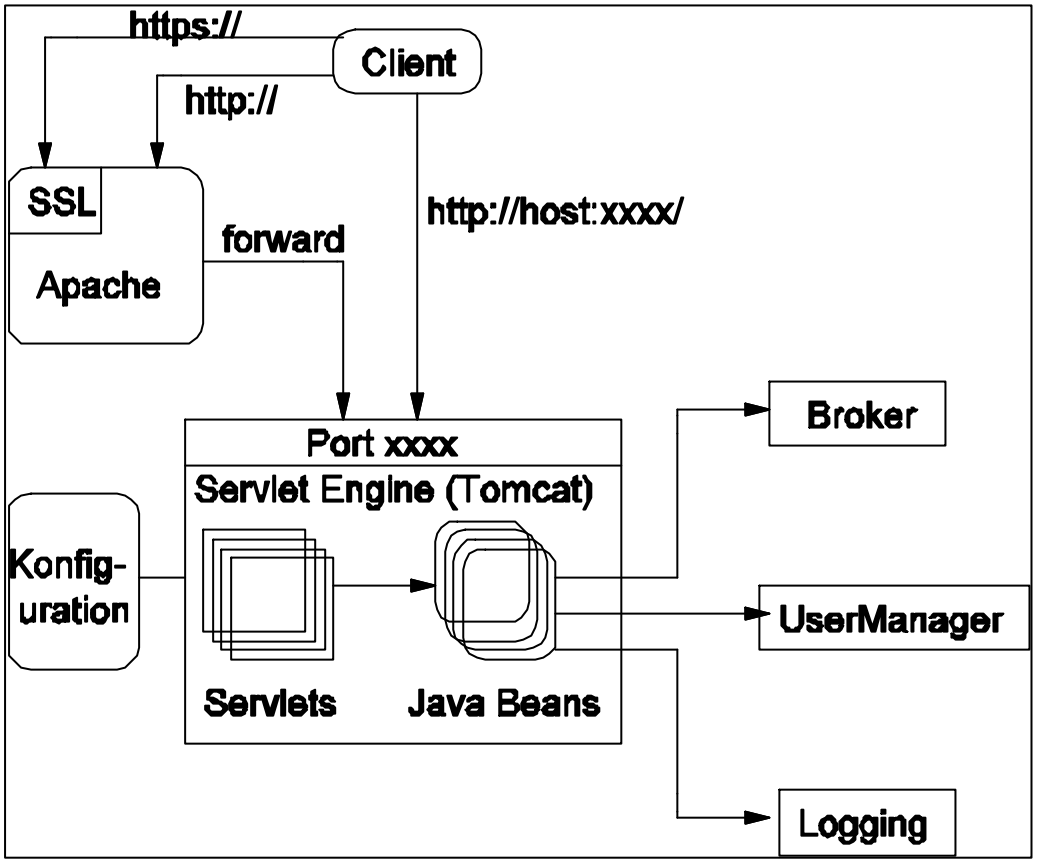
5 Servlet-/JSP-GUI

In modernen Informatiksystem erfolgt ein komplexes Zusammenspiel zwischen dem Client Rechner des Benutzers und dem Server, auf dem die Datenbanken gespeichert sind oder über den die Zugriffe auf andere Fremddatenbanken erfolgen. Insbesondere wird für den Client eine graphische Benutzerschnittstelle erwartet. In den folgenden Abschnitten wird die für Elektra II gewählte Architektur mit ihren Schnittstellen genauer beschrieben.

5.1 Features

Asynchrone Darstellung	Die asynchrone Übertragung des Elektra-Protokolls wird nicht eingeschränkt. Trifft eine Antwort von einem Target-System ein wird das Ergebnis sofort an den Benutzer weitergeleitet.
Dynamik / Performanz	Anders als bei CGI-Programmen muss für das generieren einer Webseite kein neuer Prozess gestartet werden. Beim ersten Aufruf einer JSP-Seite wird ein Servlet generiert das für alle weiteren Aufrufe wiederverwendet wird. Java Server Pages enthalten überwiegend HTML-Tags. Der Programmablauf ist weitestgehend in Java Beans ausgelagert, so dass Änderungen am Layout relativ einfach durchzuführen sind. JSP-Dateien können zur Laufzeit geändert werden ohne das Elektra zuvor reinitialisiert werden muss (Ausnahme Include-Dateien).
Verschlüsselung	Zusammen mit Apache kann die Übertragung zwischen Browser und Webserver verschlüsselt erfolgen (SSL)
Administration	Abministrationsoberfläche zur Verwaltung von Benutzern, Benutzergruppen, Kollektionen und Target-Systemen.
Benutzerdefinierte Datenbankkollektionen	Benutzer können eigene Kollektionen von Recherchedatenbanken zusammenstellen.
Statistik	Sammeln von statistischen Daten zu jedem Target-System. Anzeige aktueller Sessions.
Mehrsprachigkeit	Erweiterung um zusätzliche Sprachen durch Konfiguration von message-files

5.2 Architektur

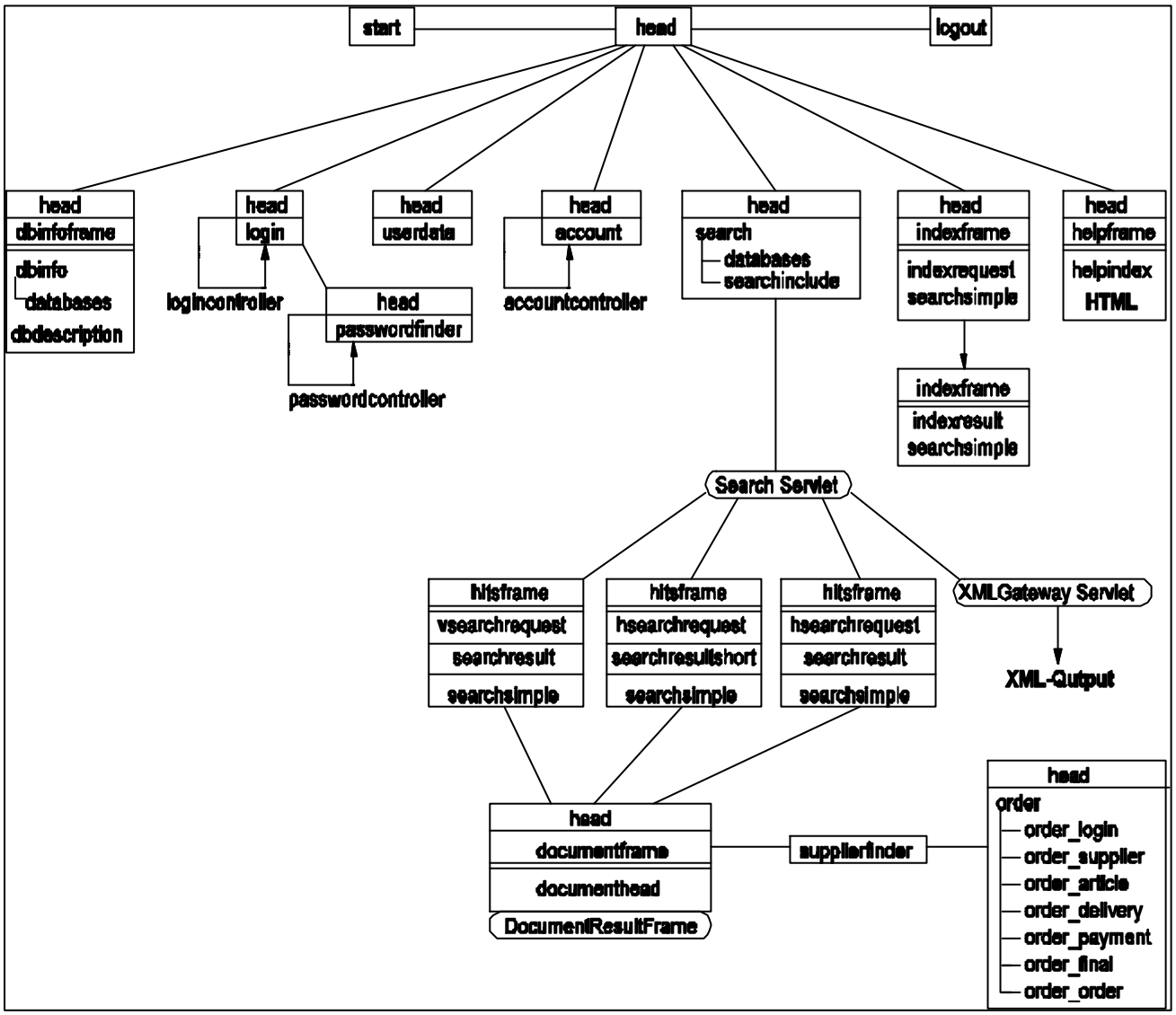


Sofern bei der Installation von Elektra, Apache und OpenSSL mitinstalliert wurden existieren drei Möglichkeiten um mittels eines Browsers auf die Benutzeroberfläche zuzugreifen:

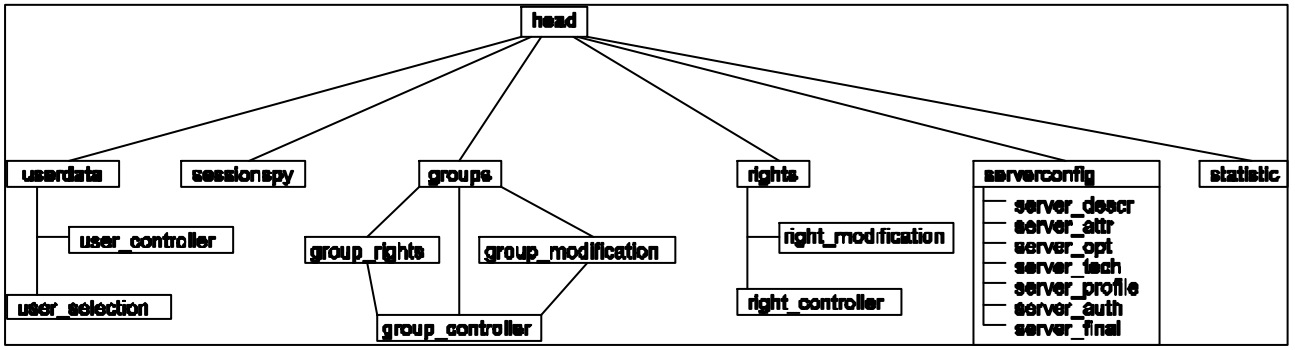
* Direkter Zugriff auf die Servlet-Engine mit »http://servlethost:servletport/jsp/start.jsp«
* Über Apache »http://servlethost/jsp/start.jsp« Etwas langsamer da der Request an Tomcat weitergeleitet werden muß.
* Mit Apache über eine verschlüsselte Verbindung (»https://servlethost/jsp/start.jsp«)

5.3 Abhängigkeiten unter den Eingabemasken

RechercheGui



AdminGui



6 Elektra Metadatengenerator und XMLGateway

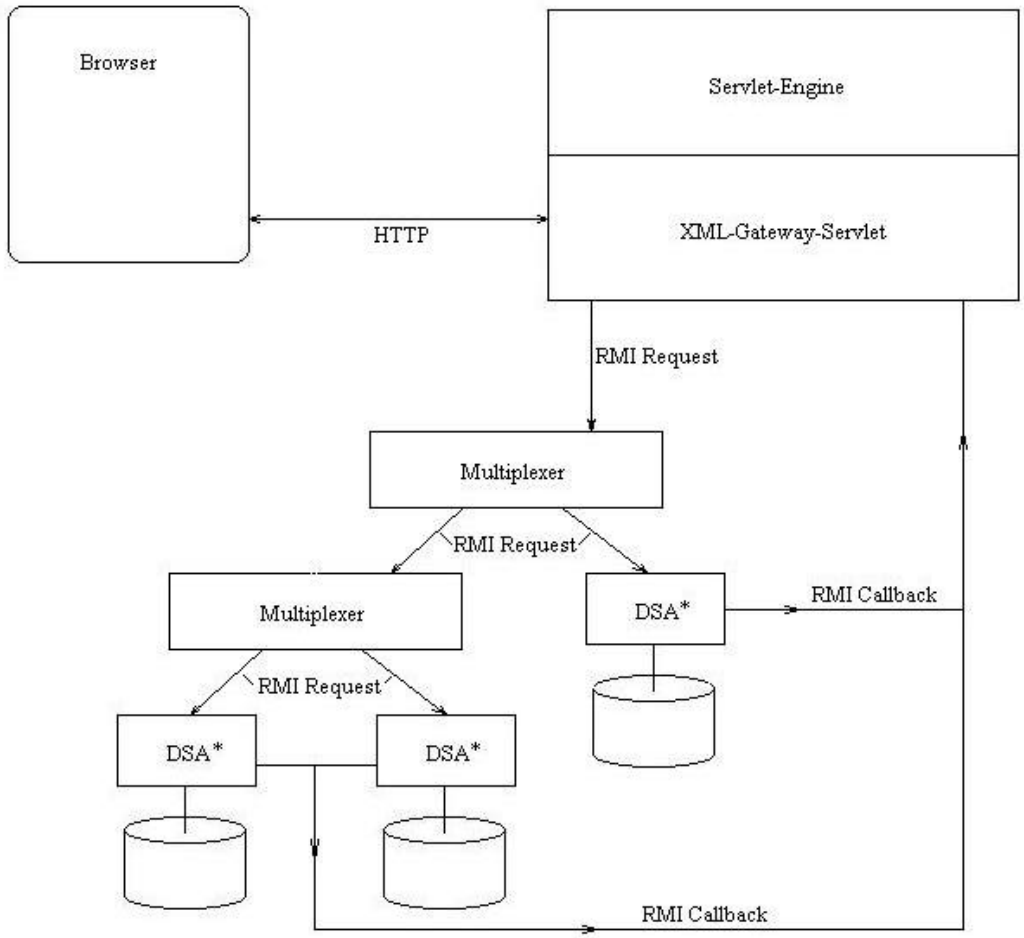
6.1 Allgemeines

Neben der proprietären ELEKTRA-internen Schnittstelle und dem standardisierten Z39.50-Target, dessen Funktionalität nur von wenigen, sehr spezialisierten Clients angesprochen werden kann, bestand Bedarf eine öffentliche, leicht zugängliche Protokollschnittstelle zu dem ELEKTRA-Broker zu schaffen. Aus Diesem Grund wurde das ELEKTRA-RMI-Protokoll auf HTTP als Transportprotokoll und XML als Darstellungsmittel umgesetzt. Anfragen werden, als URL kodiert, an ein Servlet übermittelt, von diesem auf die RMI-Requests abgebildet, und an den Broker weitergeleitet. Die Daten der synchron oder asynchron eintreffenden Ergebnisobjekte werden durch das Servlet nach XML konvertiert und mit dem Mime-Typ »text/xml« über HTTP an den Client zurückgeliefert.

6.1.1 Requests

Requests	Aufgabe
DatabaseInfoRequest	Information über vorhandene Datenbanken und deren Inhalt
SearchRequest	Anzahl der Treffer für jede Datenbank
PresentRequest	Metadaten zu den einzelnen Dokumenten in verschiedenen Formaten
CombinedSearchRequest	Zu einer Suche werden, falls vorhanden, bereits erste Datensätze mitgeliefert
ScanRequest	Liefert einen Auszug aus dem Register zu einem bestimmten Term/Suchfeld
SearchInterfaceRequest	Informationen über die Funktionalität, Suchfelder und Operatoren eines Zielsystems

6.1.2 Architektur



* DSA = Document Service Agent

6.1.3 Allgemeine Parameter

Folgende Parameter sind für alle Requests gültig:

Parameter	Beschreibung	Pflicht	Mehrfach
Request	Name des Requests (siehe 1.2)	Ja	Nein
Standalone	Falls der Wert des Feldes auf »yes« gesetzt ist, wird eine DTD mitgeliefert	Nein	Nein
Database	Pfad zu einer Datenbank. Werden keine Datenbanken spezifiziert, wird der Request an alle Datenbanken in der Multiplexer-hierarchie weitergeleitet Format: Muxer[:Muxer]*@Datenbank Beispiel: topdog:z1:v1@BVB	Nein	Ja
Timeout	Anzahl der Sekunden, die ein Request maximal benötigen darf. Wurden keine Datenbanken angegeben, wird bis zum Ablauf des Timeouts gewartet	Nein	Nein
ClientIP	IP-Adresse des Clients. Notwendig falls eine Authentifizierung beim Zielsystem erfolgt. Format: X.X.X.X	Nein	Nein
Sequence	Sequenznummer, die vom Server im entsprechenden Ergebnis zurückgeliefert wird.	Nein	Nein

6.2 Requests

6.2.1 DatabaseInfoRequest

Parameter: keine

Response:

- A Pfad zur Datenbank(serverid)
- A Name der Datenbank(dbname)
- A Url zum Original-Webinterface (dburl)
- A Beschreibung der Datenbank (dbdescription)
- A Anzahl der nachgewiesenen Titel (inventory)
- A Zuwachs an Titeln pro Jahr (increment)
- A Datenbankinhalt (Bibliographische Daten-BIB,Volltext-VT,Bilder-IMG)
- A Authentifizierungsinformationen

Beispiel:
/XMLGateway?Request=DatabaseInfoRequest&Timeout=5&Standalone=yes
(Die Datenbankbeschreibungen werden inklusive einer DTD erstellt. Es wird maximal 5 Sekunden auf das Ergebnis gewartet.)

6.2.2 SearchRequest

Parameter:

Parameter	Beschreibung	Pflicht	Mehrfach
Query	Suchanfrage in Elektra Syntax	Ja	Nein
Strict	Falls für Strict der Wert »true« gesetzt ist, wird die Anfrage zurückgewiesen, wenn sie nict semantisch korrekt auf das Zielsystem übertragen werden kann	Nein	Nein

Response:

A Anzahl der Treffer in der Datenbank(hits)
A Identifikator für nachfolgende PresentRequests
A Query in der Syntax des Zielsystems (translated_query)

Beispiel:
/XMLGateway?Request=SearchRequst&Query=ATTR-4+java+beans&Strict=true
(Suche nach Dokumenten in deren Titel die Phrase »java beans« enthalten ist. Sollte das Zielsystem z.B. keine Phrasensuche unterstützen wird keine Querytransformation vorgenommen, sondern eine Fehlermeldung zurückgegeben)

6.2.3 PresentRequest

Parameter:

Parameter	Beschreibung	Pflicht	Mehrfach
Resultid	Identifikator der Ergebnismenge und Start-wert. Format: dbpath-0:queryid:Start	Ja	Ja
Quantity	Anzahl der Records pro Datenbank	Ja	Nein
Format	Recordsyntax des Ergebnises. Erlaubt sind: DUBLINCORE,DESCRIPTION Nur für spezielle Datenbanken verfügbar: MAB, UNIMARC, USMARC, OMNIS, SOIF, HTML, OPAC, HOLDINGS	Ja	Nein

Response:

A Records in der geforderten Formatierung

Beispiel:
/XMLGateway?Request=PresentRequest&Resultid=top:s1@ZB-:5:13&Quantity=10&Format=DESCRIPTION
(holt die Records 13-22 aus dem Resultset 5 der Datenbank top:s1@ZB)

6.2.4 CombinedSearchRequest

Parameter:

Parameter	Beschreibung	Pflicht	Mehrfach
Query	Suchanfrage in Elektra Syntax	Ja	Nein
Strict	Falls für »Strict« der Wert »true« gesetzt ist, wird die	Nein	Nein

Parameter	Beschreibung	Pflicht	Mehrfach
	Anfrage zurückgewiesen, wenn sie nicht semantisch korrekt auf das Zielsystem übertragbar ist		
Hits	Anzahl der Treffer pro Datenbank	Ja	Nein
Format	Siehe PresentRequest	Ja	Nein

Response:

A Siehe SearchRequest und PresentRequest

Beispiel:

/XMLGateway?Request=CombinedSearchRequest&Query=ATTR1+Bayer&Hits=10&Format=DUBLINCORE

6.2.5 ScanRequest

Parameter:

Parameter	Beschreibung	Pflicht	Mehrfach
Term	Startterm für den Index	Ja	Nein
Attribute	Use-Attribute (numerischer Wert) des BIB-1-Attributesets (Beschreibung siehe Z39.50 Standarddoku)	Ja	Nein
Quantità	Anzahl der Einträge pro Datenbank	Ja	Nein

Response:

A Registereinträge(scanterm) und deren Häufigkeit in der Datenbank (scanocurrences)

Beispiel:

/XMLGateway?Request=ScanRequest&Term=indexing&Quantity=10&Attribute=4
(holt die ersten 10 Registereinträge ab dem Wort »indexing« (inklusive) in dem Index für Titel (Attribute 4)

6.2.6 SearchInterfaceRequest

Parameter: keine

Response:

A Verfügbare Operatoren(AND | OR | NOT | PROX)
A Verfügbare Suchfelder (BIB-1 Use-Attribute)
A Information ob ein Scan Request erlaubt ist (scan-supported).

Beispiel:

/XMLGateway?Request=SearchInterfaceRequest&Database=topdog:s1:SWETS
(Interfacefunktionalität der Swets-Datenbank)

6.3 DTD

```
<!ELEMENT elektra_protocol_responses (result*|error)>
<!ELEMENT result (header,data?)>
<!ELEMENT header (resultname,serverpath,sequence,servername,status,warnings?,errors?,data?)>
<!ELEMENT resultname (#PCDATA)>
<!ELEMENT serverpath (#PCDATA)>
<!ELEMENT sequence (#PCDATA)>
<!ELEMENT servername (#PCDATA)>
<!ELEMENT status (#PCDATA)>
<!ELEMENT errors (server_error+)>
<!ELEMENT server_error (#PCDATA)>
<!ELEMENT warnings (server_warning+)>
<!ELEMENT server_warning (#PCDATA)>
<!ELEMENT data (search_result          | present_result        | scan_result |
                sort_result            | database_info_result   |
                search_interface_result | combined_search_result)>

<!-- Search_Result ========================================================== -->

<!ELEMENT search_result (hits,queryid,translated_query)>
<!ELEMENT hits (#PCDATA)>
<!ELEMENT queryid (#PCDATA)>
<!ELEMENT translated_query (#PCDATA)>

<!-- Present_Result ========================================================= -->

<!ELEMENT present_result (resultsetid,format,start,records?)>
<!ELEMENT resultsetid (#PCDATA)>
<!ELEMENT format (#PCDATA)>
<!ELEMENT start (#PCDATA)>
<!ELEMENT records (record+)>
<!ELEMENT record (metadata)>
<!ELEMENT metadata ((dc,documentdescription) | dc | mab | marc | omnis3 | omnis4 | soif | holdings)>

<!-- DublinCore Format ====================================================== -->

<!ELEMENT dc (title*, creator*, subject*, description*, publisher*, contributor*,
              date*, type*, format*, identifier*, source*,
              language*, relation*, coverage*, rights*)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT creator (#PCDATA)>
<!ELEMENT subject (#PCDATA)>
<!ELEMENT description (#PCDATA)>
<!ELEMENT publisher (#PCDATA)>
<!ELEMENT contributor (#PCDATA)>
<!ELEMENT date (#PCDATA)>
<!ELEMENT type (#PCDATA)>
<!ELEMENT identifier (#PCDATA)>
<!ELEMENT source (#PCDATA)>
<!ELEMENT language (#PCDATA)>
<!ELEMENT relation (#PCDATA)>
<!ELEMENT coverage (#PCDATA)>
<!ELEMENT rights (#PCDATA)>

<!-- DublinCore Extentions ================================================== -->

<!ELEMENT documentdescription (formatlist?,statuslist?,bloblist?)>
<!ELEMENT formatlist (format+)>
<!ELEMENT statuslist (docstatus+)>
<!ELEMENT docstatus (#PCDATA)>
<!ELEMENT bloblist (blobdesc+)>
<!ELEMENT blobdesc (blobtype,blobcount,blobid,blobname,blobmimetype,blobsubtype,blobencoding,
                    userstring1,userstring2,userstring3,userinteger1,userinteger2,userinteger3,sourcelist?)>
<!ELEMENT blobtype (#PCDATA)>
<!ELEMENT blobcount (#PCDATA)>
<!ELEMENT blobid (#PCDATA)>
<!ELEMENT blobname (#PCDATA)>
```

```
<!ELEMENT blobmimetype (#PCDATA)>
<!ELEMENT blobsubtype (#PCDATA)>
<!ELEMENT blobencoding (#PCDATA)>
<!ELEMENT userstring1 (#PCDATA)>
<!ELEMENT userstring2 (#PCDATA)>
<!ELEMENT userstring3 (#PCDATA)>
<!ELEMENT userinteger1 (#PCDATA)>
<!ELEMENT userinteger2 (#PCDATA)>
<!ELEMENT userinteger3 (#PCDATA)>
<!ELEMENT sourcelist (blobsource+)>
<!ELEMENT blobsource (#PCDATA)>

<!-- MAB Format ============================================================ -->
<!ELEMENT mab (mabelement)>
<!ELEMENT mabelement (mabfield,mabdata)>
<!ELEMENT mabfield (#PCDATA)>
<!ELEMENT mabdata (#PCDATA)>

<!ELEMENT marc (marcelement)>
<!ELEMENT marcelement (marcfield,marcdata)>
<!ELEMENT marcfield (#PCDATA)>
<!ELEMENT marcdata (#PCDATA)>

<!-- Omnis3 Format ========================================================= -->

<!ELEMENT omnis3 (Docno, Arch_datum, ISBN, ISSN, Autor, Bandnummer, Beitragstitel,
                  Bemerkung, Heftnummer, Herausgeber, Institution, Jahr,
                  Titel, Reportnummer, Serientitel, Signatur, Standort,
                  Tagungsbezeichnung, Tagungsjahr, Tagungsort, Verlag)>
<!ELEMENT Docno (#PCDATA)>
<!ELEMENT Arch_datum (#PCDATA)>
<!ELEMENT ISBN (#PCDATA)>
<!ELEMENT ISSN (#PCDATA)>
<!ELEMENT Autor (#PCDATA)>
<!ELEMENT Bandnummer (#PCDATA)>
<!ELEMENT Beitragstitel (#PCDATA)>
<!ELEMENT Bemerkung (#PCDATA)>
<!ELEMENT Heftnummer (#PCDATA)>
<!ELEMENT Herausgeber (#PCDATA)>
<!ELEMENT Institution (#PCDATA)>
<!ELEMENT Jahr (#PCDATA)>
<!ELEMENT Titel (#PCDATA)>
<!ELEMENT Reportnummer (#PCDATA)>
<!ELEMENT Serientitel (#PCDATA)>
<!ELEMENT Signatur (#PCDATA)>
<!ELEMENT Standort (#PCDATA)>
<!ELEMENT Tagungsbezeichnung (#PCDATA)>
<!ELEMENT Tagungsjahr (#PCDATA)>
<!ELEMENT Tagungsort (#PCDATA)>
<!ELEMENT Verlag (#PCDATA)>

<!-- Omnis4 Format ========================================================= -->

<!ELEMENT omnis4 (Docno, Klasse, ZeitschriftID, ZDatenTitel, ZSerientitel, ZTitel,
                  ZTitel_Abkuerzung, ZUntertitel, ZVerlag, ZInstitution, ZHerausgeber,
                  ZStandort, ZISSN, ZExternID, ZBemerkung, HeftID, HDatenTitel, HJahr,
                  HMonat, HBandnummer, HHeftnummer, HErsteSeite, HLetzteSeite, HTagung,
                  HTagungsort, HTagungsjahr, HHerausgeber, HStandort, HSignatur, HExternID,
                  HSpeziell, HKum_Index, HRegister, HKongresse, HLieferbar, HBemerkung,
                  ArtikelID, ADatenTitel, AAutor, ATitel, AUntertitel, AErsteSeite, ALetzteSeite,
                  AExternID, ABemerkung, DokumentKlasseText, DokumentRefKlasse)>

<!ELEMENT Klasse (#PCDATA)>
<!ELEMENT ZeitschriftID (#PCDATA)>
<!ELEMENT ZDatenTitel (#PCDATA)>
<!ELEMENT ZSerientitel (#PCDATA)>
<!ELEMENT ZTitel (#PCDATA)>
<!ELEMENT ZTitel_Abkuerzung (#PCDATA)>
<!ELEMENT ZUntertitel (#PCDATA)>
<!ELEMENT ZVerlag (#PCDATA)>
```

```
<!ELEMENT ZInstitution (#PCDATA)>
<!ELEMENT ZHerausgeber (#PCDATA)>
<!ELEMENT ZStandort (#PCDATA)>
<!ELEMENT ZISSN (#PCDATA)>
<!ELEMENT ZExternID (#PCDATA)>
<!ELEMENT ZBemerkung (#PCDATA)>
<!ELEMENT HeftID (#PCDATA)>
<!ELEMENT HDatenTitel (#PCDATA)>
<!ELEMENT HJahr (#PCDATA)>
<!ELEMENT HMonat (#PCDATA)>
<!ELEMENT HBandnummer (#PCDATA)>
<!ELEMENT HHeftnummer (#PCDATA)>
<!ELEMENT HErsteSeite (#PCDATA)>
<!ELEMENT HLetzteSeite (#PCDATA)>
<!ELEMENT HTagung (#PCDATA)>
<!ELEMENT HTagungsort (#PCDATA)>
<!ELEMENT HTagungsjahr (#PCDATA)>
<!ELEMENT HHerausgeber (#PCDATA)>
<!ELEMENT HStandort (#PCDATA)>
<!ELEMENT HSignatur (#PCDATA)>
<!ELEMENT HExternID (#PCDATA)>
<!ELEMENT HSpeziell (#PCDATA)>
<!ELEMENT HKum_Index (#PCDATA)>
<!ELEMENT HRegister (#PCDATA)>
<!ELEMENT HKongresse (#PCDATA)>
<!ELEMENT HLieferbar (#PCDATA)>
<!ELEMENT HBemerkung (#PCDATA)>
<!ELEMENT ArtikelID (#PCDATA)>
<!ELEMENT ADatenTitel (#PCDATA)>
<!ELEMENT AAutor (#PCDATA)>
<!ELEMENT ATitel (#PCDATA)>
<!ELEMENT AUntertitel (#PCDATA)>
<!ELEMENT AErsteSeite (#PCDATA)>
<!ELEMENT ALetzteSeite (#PCDATA)>
<!ELEMENT AExternID (#PCDATA)>
<!ELEMENT ABemerkung (#PCDATA)>
<!ELEMENT DokumentKlasseText (#PCDATA)>
<!ELEMENT DokumentRefKlasse (#PCDATA)>

<!-- SOIF Format(Harvest) ================================================= -->

<!ELEMENT soif (soifscheme,url,soifattributelist?)>
<!ELEMENT soifattributelist (soifattribute+)>
<!ELEMENT soifattribute (soifattributename,soifattributevalue)>
<!ELEMENT soifattributename (#PCDATA)>
<!ELEMENT soifattributevalue (#PCDATA)>

<!-- Holdings Format ====================================================== -->

<!ELEMENT holdings (library*)>
<!ELEMENT library (libname,sigel,bestand)>
<!ELEMENT libname (#PCDATA)>
<!ELEMENT sigel (#PCDATA)>
<!ELEMENT bestand (#PCDATA)>

<!-- Scan_Result ========================================================== -->

<!ELEMENT scan_result (scanitems?)>
<!ELEMENT scanitems (scanitem+)>
<!ELEMENT scanitem (scanterm,scanocurrences)>
<!ELEMENT scanterm (#PCDATA)>
<!ELEMENT scanocurrences (#PCDATA)>

<!-- Sort_Result ========================================================== -->

<!ELEMENT sort_result (#PCDATA)>

<!-- Database_Info_Result ================================================= -->

<!ELEMENT database_info_result (serverid,dbname,dburl,dbdescription,dbinventory,
```

```
                                        dbincrement,dbcontent?,dbauthentification?)>
<!ELEMENT serverid (#PCDATA)>
<!ELEMENT dbname (#PCDATA)>
<!ELEMENT dburl (#PCDATA)>
<!ELEMENT dbdescription (#PCDATA)>
<!ELEMENT dbinventory (#PCDATA)>
<!ELEMENT dbincrement (#PCDATA)>
<!ELEMENT dbcontent (dbcontentitem+)>
<!ELEMENT dbcontentitem (#PCDATA)>
<!ELEMENT dbauthentification (dbauthitem+)>
<!ELEMENT dbauthitem (#PCDATA)>

<!-- Combined_Search_Result ================================================= -->

<!ELEMENT combined_search_result (resultsetid,format,start,records?,hits,translated_query)>

<!-- Search_Interface_Result ================================================ -->

<!ELEMENT search_interface_result (scan-supported,operators,attributes)>
<!ELEMENT scan-supported (#PCDATA)>
<!ELEMENT operators (operator+)>
<!ELEMENT operator (opid,opname)>
<!ELEMENT opid (#PCDATA)>
<!ELEMENT opname (#PCDATA)>
<!ELEMENT attributes (attribute+)>
<!ELEMENT attribute (#PCDATA)>

<!-- Error handling ========================================================= -->

<!ELEMENT error (errormsg,errorstack)>
<!ELEMENT errormsg (#PCDATA)>
<!ELEMENT errorstack (#PCDATA)>

<!-- END ==================================================================== -->
```

7 Archiviererserver

7.1 Überblick

Der Archiviererserver ist für die Archivierung von Dokumenten zuständig. Er bietet Schnittstellen zur Archivierung von Dokumenthierarchien für Nachweisdatenbanken sowie Schnittstellen zur Bearbeitung und Archivierung von bestellten Dokumenten. Dokumenthierarchien, wie z.B. Zeitschrift - Heft - Artikel, können beliebig für eine Nachweisdatenbank vorkonfiguriert werden. Der Archiviererserver liest beim Aufbau der Verbindung zu einer Nachweisdatenbank deren Konfigurationstabellen aus. Somit kann jeder Client jederzeit den Aufbau aller Nachweisdatenbanken erfahren. Ebenso lässt sich die graphische Benutzerschnittstelle für jede Nachweisdatenbank vorkonfigurieren, so dass sich das GUI dynamisch an die Gegebenheiten der jeweiligen Datenbank anpasst.

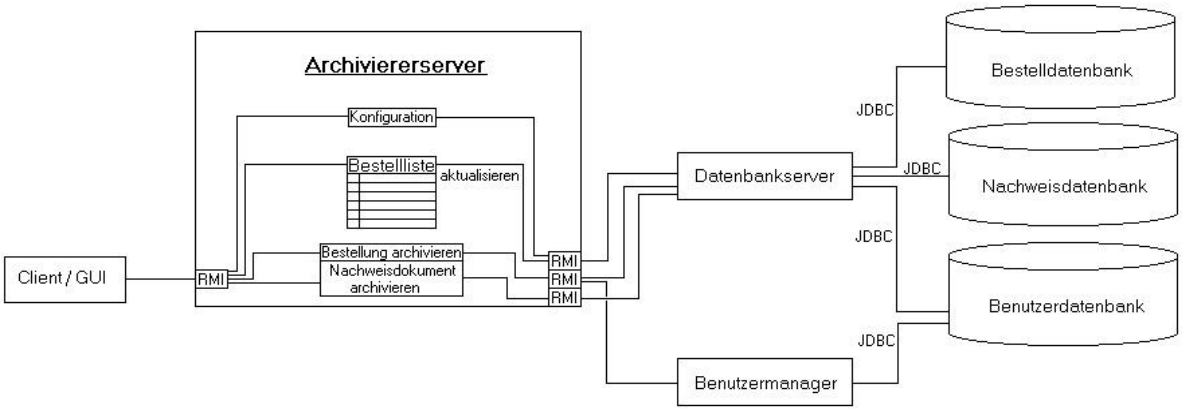


Abbildung 2: Archiviererserver

7.2 Ablauf

Nachweisarchivierung:
Beim Start des Archiviererservers baut er die Verbindung zu der ihm zugewiesenen Nachweisdatenbank auf. Dabei liest er aus deren Konfigurationstabellen folgende Information aus:

- Anzahl der Hierarchiestufen der zu archivierenden Nachweisdokumente, sowie deren Bezeichnungen
- Alle Felder der bibliographischen Daten, die zu einem Dokument gehören, z.B. Autor, Titel, ISSN usw. (Metadaten)
- Konfiguration der GUI - Schnittstelle für die Nachweisarchivierung, z.B. wo ist welches Eingabefeld positioniert und zu welchem Metadatum gehört es

Wird jetzt die GUI - Schnittstelle gestartet, so liest diese zunächst vom Archiviererserver die GUI - Konfiguration der angebundenen Nachweisdatenbank. Die graphische Schnittstelle wird dann dynamisch aufgebaut.
Der Benutzer hat jetzt die Möglichkeit, sich die bereits in der Datenbank befindlichen Dokumente anzusehen bzw. danach zu suchen. Die Anzeige eines Dokuments erfolgt über eine Baumdarstellung, wodurch die Position innerhalb der Hierarchie veranschaulicht

wird (s. Abb. 7-3). Zudem werden die bibliographischen Daten in den entsprechenden Feldern angezeigt. Hat sich der Anwender nun entschieden z.B. einen bestimmten Artikel zu archivieren, so sucht er zunächst nach der zugehörigen Zeitschrift in der Datenbank. Findet er diese, navigiert er über die Baumstruktur zum gewünschten Heft und legt dort einen neuen Artikel an, gibt die bibliographischen Daten ein, fügt evtl. vorhandene Text- und Bilddateien an und archiviert den Artikel. Ist die Zeitschrift oder das Heft noch nicht vorhanden, lassen sich diese ebenso in der Baumstruktur neu generieren.
Der Archiviererserver prüft zu archivierende Dokumente, ob diese schon in der Nachweisdatenbank vorhanden, ob die Hierarchieangaben stimmen und die Metadaten entsprechend der Konfiguration strukturiert sind. Ist alles korrekt, so wird eine neue Dokumentnummer erzeugt und das Dokument archiviert.

Bestellung archivieren:

Beim Start des Archiviererservers liest er aus der Bestelldatenbank die eingegangenen Bestellungen aus. Dieser Vorgang wird zu einem vorgegebenen Zeitpunkt wiederholt, z.B. alle 5 Minuten.
Die GUI - Schnittstelle (s. Abb. 7-5) aktualisiert ebenfalls zu vorgegebenen Zeiten oder auf Kommando die angezeigte Liste von Bestellungen. Wählt der Anwender eine Bestellung zur Bearbeitung aus, werden die für ihn relevanten Daten im GUI angezeigt, z.B. ISSN, Bandnummer., Jahrgang., evtl. Signaturen, so dass er das entsprechende Dokument finden kann. Nach dem Einscannen der Bilddateien können diese über das GUI importiert werden. Per Kommando kann der Benutzer nun die Bestellung archivieren.
Der Archiviererserver erhält über die entsprechende Schnittstelle den Auftrag, eine bearbeitete Bestellung zu archivieren. Die Informationen über die Benutzerdatenbank ist der Bestellung beigefügt, so dass keine Bindung an eine bestimmte Benutzerdatenbank vorgegeben ist. Der Server prüft jetzt zunächst, ob die Bestellung ebenfalls in der Benutzerdatenbank eingetragen ist, in die sie archiviert werden. Ist das nicht der Fall, so wird die Bestellung verworfen und aus der Bestelldatenbank gelöscht. Wenn sie dort vorhanden ist, wird der Status der Bestellung über eine Schnittstelle die Benutzermanagers auf "bearbeitet" gesetzt und anschliessend werden die Bilddateien zusammen mit den vorhanden bibliographischen Daten in die Benutzerdatenbank archiviert.

Fremddatenübernahme:

Der Archiviererserver besitzt eine Schnittstelle zur Fremddatenübernahme. So ist es z.B. möglich, Daten von Swets & Zeitlinger (Textdateien mit bibliographischen Informationen zu Zeitschriften, Heften, Artikeln - wird wöchentlich geliefert) über diese Schnittstelle einzulesen, in das gewünschte Nachweisformat umzuwandeln und anschliessend in eine Nachweisdatenbank zu übertragen. Die Bearbeitung kann auch im Batchmodus automatisiert werden. Eine entsprechende GUI - Maske ist bereits vorhanden (s. Abb. 7-4).

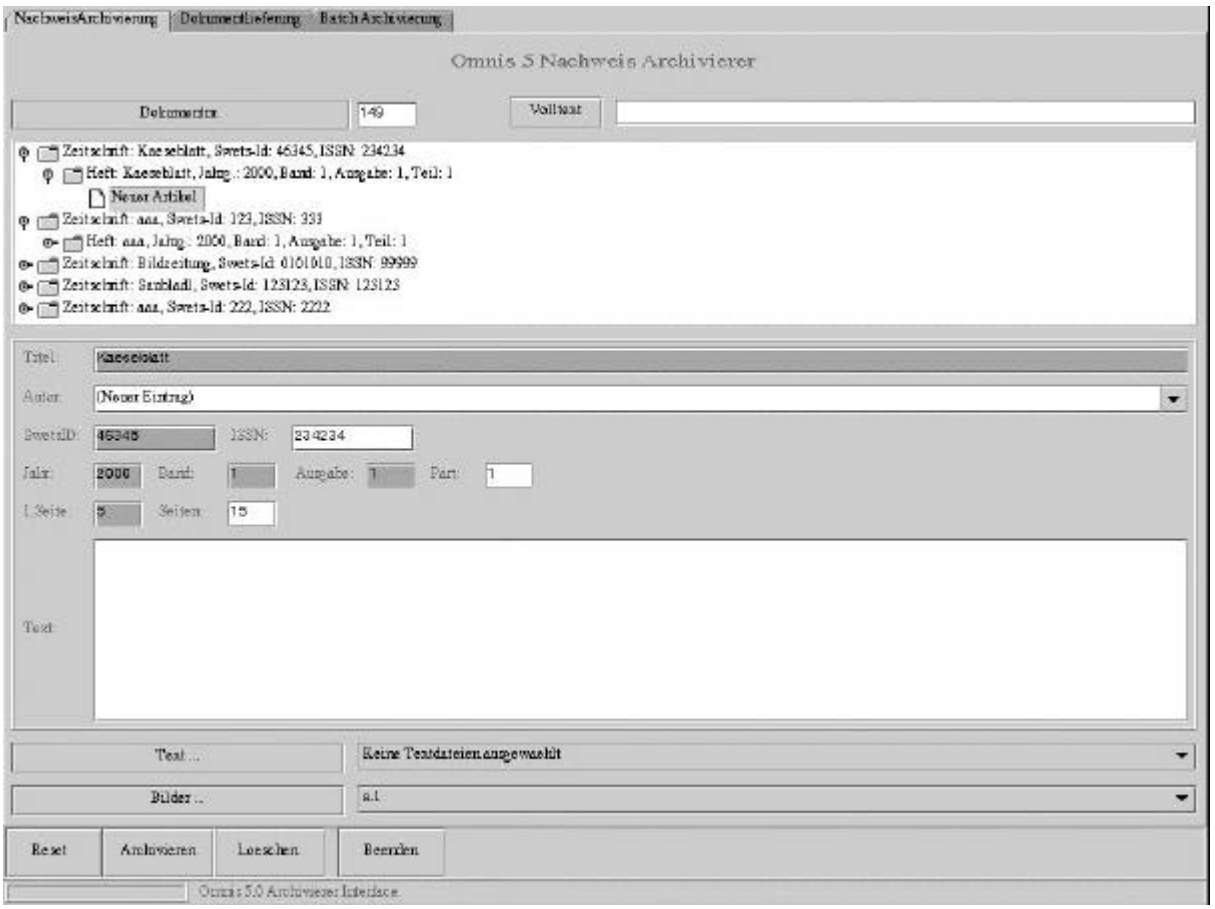


Abbildung 7-3: Nachweis-Archivierung

Abbildung 7-4: Batch-Archivierung

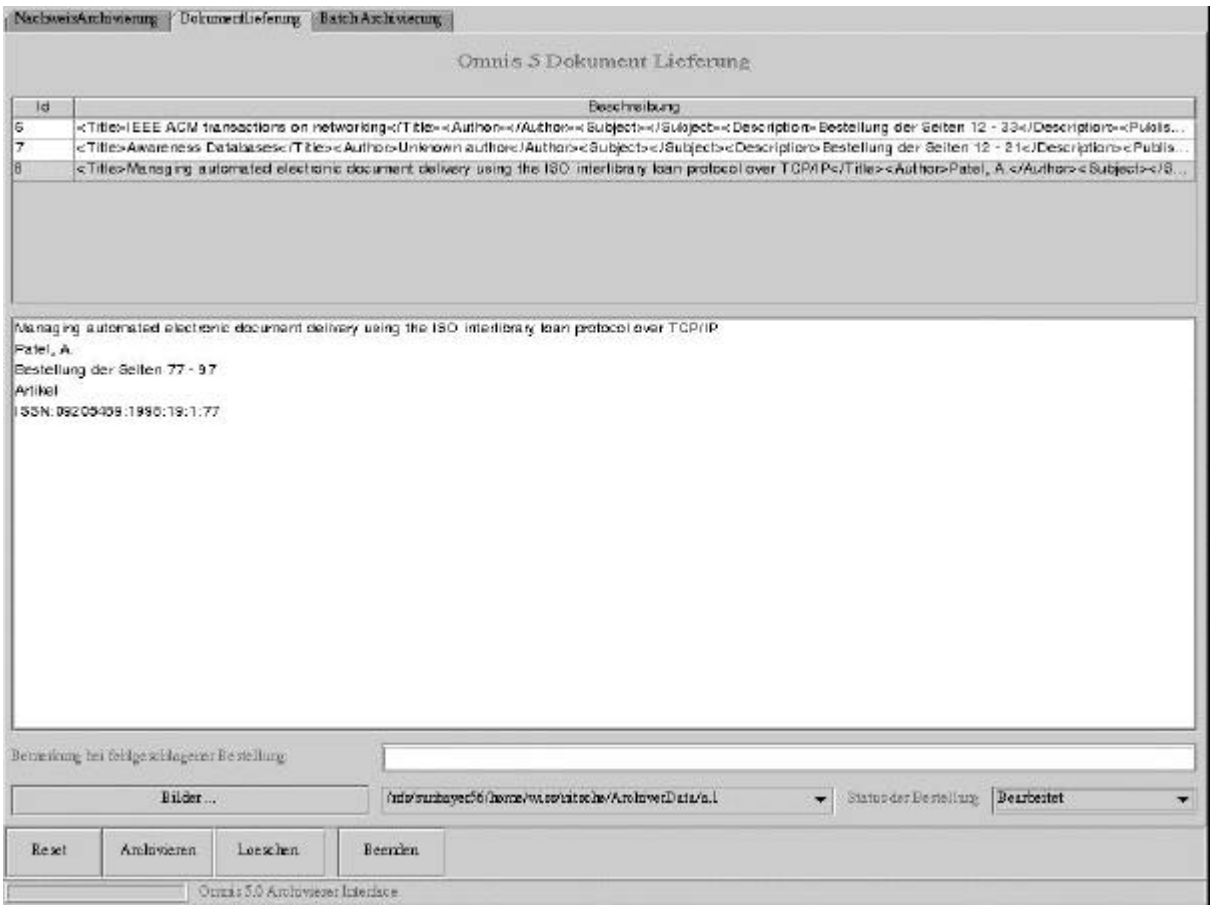


Abbildung 7-5: Bestellungen archivieren

8 Einsatz von Elektra 2

Die Preisentwicklung im Bereich elektronischer und konventioneller Zeitschriften führt dazu, dass die Universitäten Einsparungsmöglichkeiten noch mehr als bisher ausschöpfen müssen. Abonnements hochpreisiger Mehrfachexemplare in den Instituten und Lehrstühlen müssen zugunsten adäquater schneller Dokumentinformation und -lieferung aufgegeben werden. Das im Rahmen des Projektes DIBWIN entwickelte System ELEKTRA bietet als derzeit einziges marktreifes Produkt alle diese Funktionalitäten an:

- Dateninput durch Archivierung von Inhaltsverzeichnissen und Abstracts konventioneller Zeitschriften
- Texterkennung und Rechercheoberfläche
- Parallele Suche im eigenen und externen Dokumentbestand
- Integrierte Bestellung von Aufsätzen und anschließende Lieferung durch die besitzende Bibliothek
- Integrierte WWW-Oberfläche für alle Funktionalitäten

Im Rahmen der Neustrukturierung der Universität hat die Hochschulleitung der Technischen Universität München beschlossen, alle Zeitschriften der Lehrstühle und Institute zu zentralisieren, um so einen unmittelbaren elektronischen Zugriff mittels Dokumentlieferung für aller Mitglieder der Universität zu ermöglichen. Die Einsparungen durch Elimination von Dubletten sind aufgrund variierenden Bedarfs in den einzelnen Betriebseinheiten nicht präzise abzusehen, werden sich aber auf mehrere 10 TDM jährlich belaufen. Die Leitung der Universitätsbibliothek beabsichtigt, mit dem flächendeckenden Einsatz von ELEKTRA im Januar 2002 in den Teilbibliotheken der Bereiche Medizin, Maschinenbau, Chemie, Physik und des Wissenschaftszentrums in Weihenstephan sowie der Hauptbibliothek zu beginnen, aufbauend auf den langjährigen Erfahrungen in der Mathematik/Informatik Bibliothek mit dem Vorgängersystem von ELEKTRA. Im Laufe der 2. Hälfte des Jahres wird das Angebot auf die restlichen Teilbibliotheken Wirtschafts- und Sozialwissenschaften, Geografie, Architektur und Sport ausgedehnt werden. Entsprechende Gelder zur Sicherung des längerfristigen Einsatzes von ELEKTRA sind im derzeit laufenden HBFG-Antrag der Universitätsbibliothek beantragt. Ein Workshop mit der Firma SISIS zum Ausbau von ELEKTRA zu einem SUBITO-Portal wird im Oktober 2001 stattfinden. Durch diese geplanten Entwicklungen wird ELEKTRA mehr denn je in bundesweite Standards des Bibliotheks- und Informationswesens eingebunden sein.

Elektra 2 wird im Moment eingesetzt bei folgenden Institutionen:

- Bibliothek der TU - Cottbus
- Bayerische Staatsbibliothek (BSB) in München
- Universitätsbibliothek - Köln (ab Oktober 2001)

Testinstallationen von Elektra 2 sind hier im Einsatz:

- Konrad-Zuse-Zentrum (ZIB)
- Universitätsbibliothek Bonn (ab Oktober 2001)

(Stand: September 2001)

9 Publikationen

1. R. Bayer:
   XML Databases: Modeling and Multidimensional Indexing.
   DEXA Conference 2001, Invited Talk, Technische Univ. München, Sept. 2001

2. R. Bayer, R. Heinrich, D. Nitsche:
   The ELEKTRA Digital Library Meta-System.
   Workshop "Von Suchmaschinen zu Virtuellen Bibliotheken",
   ZIB/KOBV Zuse Zentrum Berlin, 26.-27. Juni 2000

3. R. Bayer: Integration of Digital Services for Libraries.
   Eingeladener Hauptvortrag, International Conference on Digital Libraries: Advanced methods and technologies, digital collections. 18.-22. Oktober, 1999, Sankt Petersburg, Russland , Oktober 1999

4. H. Haddouti, Wolfgang Wohner, Rudolf Bayer.
   Towards a Scalable System Architecture in Digital Libraries. DEXA 1999: 852-861, Florence, Italy, August 30 - September 3, 1999.

5. W. Kowarschick, P. Vogel, R. Bayer: Elektra:
   An Electronic Article Delivery Service: In: Proc. Of the 8th International Workshop on Database and Export System Applications, (DEXA 97, Toulouse, France) IEEE Sept. 1997, S. 272 – 277.

6. R. Bayer (ed):
   Neue Informations und Kommunikationstechnologien für wiss. Bibliotheken. Bayer. Kultusministerium ISBN 3-598-11350-1 Bericht IKB Kommission, Saur Verlag, 1997.

7. R. Bayer, M. Dörr, H. Haddouti and S. Wiesener:
   The German National Bibliography 1601 – 1700: Digital Images in a Cooperative Cataloging Project. In: Proceedings of the IEEE Forum on Research and Technology, Advances in Digital Libraries, IEEE ADL'97, May 7-9, 1997, Washington D.C.

10 Ausblick

Mit Ende der Projektlaufzeit zum 31.Oktober 2000 konnte mit der Firma Sisis ein Partner gefunden werden, um Wartung und Weiterentwicklung der in Elektra II realisierten Komponenten zu gewährleisten. Sisis beteiligt sich dabei sowohl an der fachlichen Konzeption, wie auch an deren technischen Umsetzung in einem kommerziellen System.

Als mittelfristige Ziele der Kooperation wurden folgende Punkte definiert:

- Vermarktung von Elektra als eigenständiges Informationsportal
- Integration des Brokers in die Sisis Produktpalette
- Einbinden des Sisis-Fernleihemoduls
- Erweiterung des Aufsatzlieferdienstes der TU-München um eine Subito.3-Inputschnittstelle