Otto-von-Guericke-Universität Magdeburg

Fakultät für Informatik
Institut für Technische und Betriebliche Informationssysteme

Bachelorarbeit

Entwicklung eines Webtools zur Bearbeitung des Datenmodells der Stoffdatenbank GSBL

Verfasser:

Philipp Müller

21. Februar 2016

Betreuer:

Prof. Dr. rer. nat. habil. Gunter Saake,
Dr.-Ing. Eike Schallehn,
M.Sc. Stefan Barthel

Universität Magdeburg
Fakultät für Informatik
Postfach 4120, D–39016 Magdeburg
Germany

Philipp, Müller:
Entwicklung eines Webtools zur Bearbeitung des Datenmodells der Stoffdatenbank GSBL
Bachelorarbeit, Otto-von-Guericke-Universität Magdeburg, 2016.

Danksagung

Ich bedanke mich in erster Linie bei meinem Betreuer Herrn **Stefan Barthel** vom Umweltbundesamt, welcher mir die Erstellung dieser Arbeit ermöglicht hat und mir stets mit seinen Ideen und Wissen zur Seite stand.

Weiterhin bedanke ich mich bei **Dr.-Ing. Eike Schallehn** für seine Hilfe zu allen organisatorischen Anliegen und seinen Ratschlägen für den letzten Feinschliff zu dieser Arbeit.

Abschließend möchte ich mich noch bei **Prof. Dr. Gunter Saake** bedanken, für seine Bereitschaft die Aufgabe des Gutachters dieser Arbeit zu übernehmen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Abkürzungen

AJAX	Asynchronous JavaScript and XML
ANSI	American National Standards Institute
ARPA	Advanced Research Project Agency
BMUB	Bundesumweltministerium
CERN	European Organization for Nuclear Research
CGI	Common Gateway Interface
CSS	Cascading Style Sheets
CSV	Comma-separated values
DB	Datenbank
DBMS	Datenbankmanagementsystem
DBS	Datenbanksystem
DCL	Data Control Language
DDL	Data Definition Language
DML	Data Manipulation Language
DOM	Document Object Model
DQL	Data Query Language
GSBL	Gemeinsamer zentraler Stoffdatenpool des Bundes und der Länder
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
ID	Identifikationsnummer

IP	Internet Protocol
IPv4	Internet Protocol Version 4
IPv6	Internet Protocol Version 6
NAP	Network Access Point
PHP	PHP Hypertext Preprocessor
POP	Point of Presence
SPARC	Standards Planning and Requirements Committee
SQL	Structured Query Language
TCP	Transmission Control Protocol
UBA	Umweltbundesamt
URL	Uniform Resource Locator
WWW	World Wide Web
XML	Extensible Markup Language

1. Einleitung

Datenbanken werden heutzutage in jedem Lebensbereich in dem große Mengen an Daten anfallen als ein System zur elektronischen Datenverwaltung genutzt und mit ihrer Nutzung wird sichergestellt, dass diese großen Datenmengen effizient, strukturiert, widerspruchsfrei, sicher und dauerhaft gespeichert sind. Mit der Nutzung einer Datenbank wird sich zudem eine schnelle und fehlerfreie Anforderung der Daten erhofft. Die Art und Weise wie die Daten in einer Datenbank gespeichert und bearbeitet werden können, wird durch ein Datenmodell festgelegt. Das Datenbankmodell bildet somit die theoretische Grundlage für eine Datenbank und legt deren Infrastruktur und somit deren grundlegende Effektivität und Effizienz fest. Eine gute Datenmodellierung ist Pflicht, denn selbst mit einer guten Datenbanksoftware lässt sich eine schlechte Datenmodellierung nur schwer ausgleichen. Das Hauptziel der Datenmodellierung ist deshalb die eindeutige Definition und Spezifikation der zu verwaltenden Daten in Objekte und Attribute sowie eine optimierte Bestimmung der Zusammenhänge zwischen den einzelnen Objekten. Das Ergebnis dieser Datenmodellierung sind Datenmodelle, welche oft mehrere Stufen durchlaufen bis hin zu einer einsatzfähigen Datenbank. Die Datenmodellierung ist jedoch selbst bei Inbetriebnahme der Datenbank ein stetiger iterativer Prozess, denn es kommt im Laufe der Zeit meist zu Veränderungen und Erweiterungen der Datenbank, die eine Neuoptimierung der Datenbank unumgänglich macht.

Eine große Datenbank mit einem sehr umfangreichen Datenmodell ist der GSBL (Gemeinsamer zentraler Stoffdatenpool des Bundes und der Länder), welcher von der deutschen Verwaltung vorrangig als Chemikalien-Informationsquelle genutzt wird. Die bisherige Arbeit der chemischen Fachleute des GSBL-Teams ist jedoch sehr fehleranfällig bei der Erstellung, beziehungsweise der Änderung des Datenmodells des GSBLs. Um die Kosten zur Behebung dieser Fehler zu verringern sollen zukünftig diese Fehler auf ein Minimum reduziert werden. Die Verantwortlichen des GSBLs wünschen sich deshalb eine Möglichkeit zur Fehlerreduzierung mit Hilfe einer maßgeschneiderten Anwendungssoftware zur Bearbeitung des Datenmodells. Ein Nutzer dieser Anwendungssoftware soll zukünftig alle möglichen Änderungen am Datenmodell darüber vornehmen können und er soll darauf aufmerksam gemacht werden, ob die von ihm gewünschten Änderungen fehlerfrei sind. Um zusätzlich noch die Möglichkeiten der heute gut vernetzten Welt ausnutzen zu können, soll diese Anwendung über das Internet nutzbar sein, weshalb diese Anwendung auch als „Webtool“ bezeichnet werden kann.

Ziel dieser Arbeit ist es, eine erste prototypische Umsetzung dieses Webtools zur Bearbeitung des Datenmodells des GSBLs zu entwickeln. Im Zuge dessen werden Datenbankbegrifflichkeiten sowie alle Technologien, welche bei der Entwicklung des Webtools zum Einsatz kamen und wie die Kommunikation zwischen Client und Server über das Internet abläuft, in den Grundlagen erläutert. Anschließend erfolgt

eine Vorstellung des GSBLs und dessen Datenmodell sowie eine Formulierung der Anforderungen an das Webtool. Im Zuge der Implementierung wird die Entwicklung des Webtools erläutert und anhand von anschaulichen Programmcodeauszügen wird die Umsetzung, getrennt nach Client und Server, gezeigt. Mit Kapitel 5 erfolgt eine abschließende Gegenüberstellung, der in Kapitel 3 gestellten Anforderungen und der aus Kapitel 4 gezeigten Implementierung, einhergehend mit einem Vergleich inwiefern die geforderten Anforderungen umgesetzt wurden.

2. Grundlagen

Im folgenden Abschnitt werden die Grundlagen zu dieser Arbeit erklärt. Zunächst wird auf das Thema Datenbanken eingegangen und anschließend werden die Grundbegriffe des Internets und die Programmiersprachen, welche für das Erstellen des Webtools verwendet wurden, näher erklärt.

2.1. Datenbankgrundlagen

Alle notwendigen Grundlagen, zum Verständnis der Arbeit, werden in diesem Kapitel behandelt. Für eine detaillierte und ausführliche Betrachtung sei jedoch auf das Buch „Datenbanken Konzepte und Sprachen“ [SSH10] verwiesen.

2.1.1. Grundlegende Datenbankterminologie

Im folgenden Kapitel gilt es die wichtigsten Begrifflichkeiten in Bezug auf Datenbanken zu vermitteln.

2.1.1.1. Datenbank

Bei einer Datenbank handelt es sich um eine organisierte Sammlung von Daten. Zur Verwaltung einer Datenbank wird ein Datenbankmanagementsystem (DBMS) benutzt, welches den darauf operierenden Anwendungssystemen und Benutzern verborgen bleibt. Zum Speichern der Daten werden diese auf nichtflüchtigen physischen Speichermedien gesichert[SSH10].

2.1.1.2. Datenbankmanagementsystem

Ein Datenbankmanagementsystem ist eine Software zur Verwaltung von einer oder mehrerer Datenbanken. Mit dieser Software werden alle Anfragen, wie Änderungs- oder Einfügeoperationen der DB bearbeitet und durchgeführt. Zur Übermittlung der Befehle wird eine Datenbanksprache, wie zum Beispiel SQL benutzt. Weitere Aufgaben des DBMS ist die Verwaltung der Speicherhierarchien und des Zwischenspeichers, auch Puffern genannt, welches ein wichtiges Maß über Funktionalität und Geschwindigkeit des Gesamtsystems darstellt. Weiterhin wird das Datenbankmodell durch das DBMS definiert, wodurch alle Daten einheitlich beschrieben werden[SSH10].

2.1.1.3. Datenbanksystem

Das Datenbanksystem ist die Kombination der Datenbank, welche als Datenbasis fungiert und des DBMS, welches das Verwaltungsprogramm darstellt. Es ist ein System zur

Beschreibung, Speicherung und Wiedergewinnung umfangreicher Datenmengen, welches von mehreren Anwendungsprogrammen oder Anwendern benutzt wird[SSH10], wie es in Abbildung 2.1 anschaulich dargestellt wird.

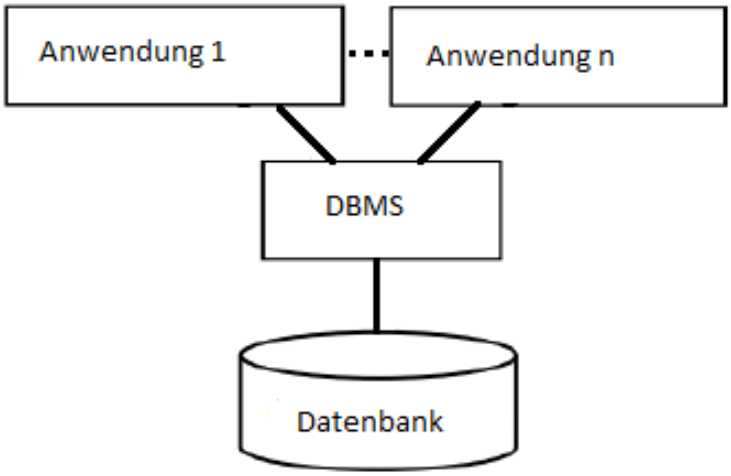


Abbildung 2.1.: Aufbau des Datenbanksystems (angelehnt an [SSH10])

2.1.1.4. Relationale Datenbankmanagementsysteme

Das am weitesten verbreitete Modell ist das von Edgar Frank Codd, aus dem Jahr 1970, vorgestellte Relationenmodell, welches sich in der Praxis etabliert hat und wird von den erfolgreichsten DBMS zu Grunde gelegt. Es werden dabei gleichartige Objekte durch n-Attribute beschrieben, wobei jedes Attribut durch einen bestimmten Ausprägungstyp, wie ganzzahlige Zahlen, Fließkommazahlen, Zeichen oder Wahrheitswerte charakterisiert wird. Die Relation ist die sequentielle Auflistung dieser Objekte, welche durch ihre Eigenschaften, auch Attribute genannt, präsentiert werden. Jedes einzelne Objekt in dieser Relation wird auch als Tupel bezeichnet[Cod70].

Wie in der Abbildung 2.2 zu erkennen ist, kann eine Relation anschaulich als Tabelle verstanden werden, wobei Attribute als Spaltenüberschriften, die Menge aller Attributnamen als Relationenschema, eine Zeile als Tupel und der Verbund aller Tupel als Relation verstanden werden kann.

Des Weiteren werden Attribute oder auch Attributkombinationen, welche ein Tupel eindeutig identifizieren als Schlüssel bezeichnet. Als Schlüsselattribut eignet sich hierfür die Vergabe einer Identifikationsnummer, kurz ID genannt, da jedem Tupel somit eine unterschiedliche Identifikationsnummer zugewiesen werden kann.

2.1.2. Architektur und Eigenschaften eines Datenbanksystems

In diesem Abschnitt wird auf die grundlegenden Architekturen und den Eigenschaften eines Datenbanksystems eingegangen.

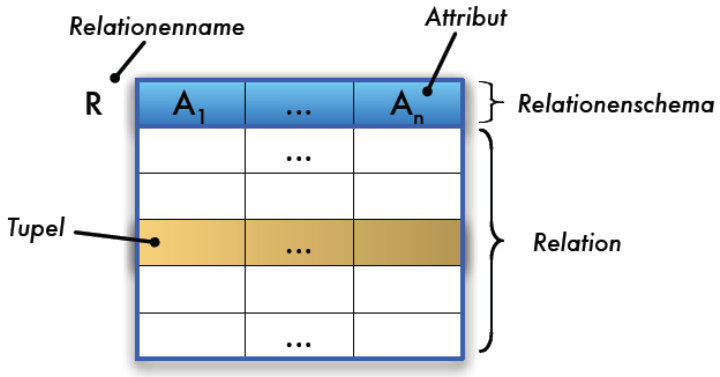


Abbildung 2.2.: Aufbau einer Relation (vgl. [SSH10])

2.1.2.1. Schemaarchitektur

Die Schemaarchitektur begründet sich auf der 1975 vom Standards Planning and Requirements Committee (SPARC) des American National Standards Institute (ANSI) entwickelten Drei-Ebenen-Schema-Architektur. Diese beschreibt die Trennung verschiedener Beschreibungsebenen für Datenbanksysteme. Eine tiefer gehende Betrachtung zeigt [oDBMS75], da hier die Schemaarchitektur zum ersten Mal definiert wurde. Bei den drei Schemata handelt es sich um folgende:

- Das externe Schema beschreibt das Ergebnis der Sichtdefinition auf den Gesamtdatenbestand und verweist zumeist auf das lokale konzeptuelle Schema. Dies geschieht beispielsweise durch eine View, die ihre Ergebnisse aus einer oder mehreren unterschiedlichen Tabellen bezieht.
- Das konzeptuelle Schema ist das Ergebnis der Datendefinition. Es liefert eine anwendungsunabhängige Gesamtsicht auf den Datenbestand. Die konzeptuelle Gesamtsicht erfolgt in relationaler Darstellung.
- Das interne Schema legt die Dateiorganisation und Zugriffspfade auf physische Daten fest, welche durch das konzeptuelle Schema in abstrakterer Weise genutzt werden.

Die Zusammenhänge zwischen den drei Schemata seien in Abbildung 2.3 veranschaulicht dargestellt. Das externe und das konzeptuelle Schema sind unabhängig von der jeweils darunter liegenden Schicht und können somit angepasst werden, ohne dass an weiteren Schichten Änderungen vorgenommen werden müssen. Bezüglich der konzeptuellen Ebene wird von einer Implementierungsunabhängigkeit oder physischen Datenunabhängigkeit gesprochen, bei der Änderungen der Dateiorganisationen und Zugriffspfade keinen Einfluss auf das konzeptuelle Schema haben. Bei der externen Ebene wird von einer Anwendungsunabhängigkeit oder logischen Datenunabhängigkeit gesprochen, bei der Änderungen am konzeptuellen und gewissen externen Schemata keine Auswirkungen auf andere externe Schemata und Anwendungsprogramme haben. Bei der Weitergabe von Anfragen und Änderungstransaktionen zwischen den Ebenen, sind diese in einer formalen Beschreibungssprache mit festgelegter Semantik verfasst. Die Anfragebearbeitung zeigt auf, welchen Weg die Anfragen und Änderungsoperationen zu bewältigen haben. Hierfür muss eine extern formulierte Operation in eine interne Operation, welche sich auf die Datenstruktur bezieht, umgewandelt werden. Des Weiteren beschreibt die Datendarstel-

lung welchen Weg die Ergebnisse bei der Überführung von der internen Datenstruktur in die externe Darstellung gehen sollen.

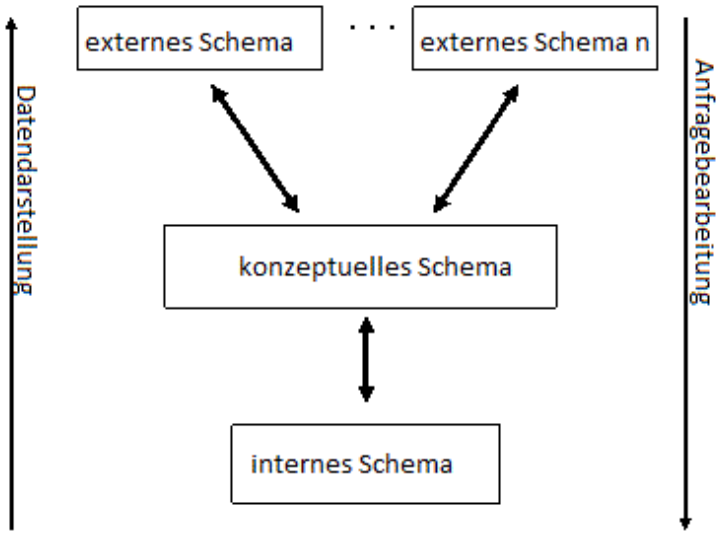


Abbildung 2.3.: Drei-Ebenen-Schema-Architektur (angelehnt an [SSH10])

2.1.2.2. Anwendungsarchitektur

Die Anwendungsarchitektur beschreibt, was bei der Nutzung einer Applikation sowie was für konkrete Aufgaben anfallen und welche Komponenten und Schnittstellen genutzt werden. Die Architektur von Datenbankanwendungen beruht typischerweise auf der Basis des Client-Server-Modells. Das Client-Server-Modell beschreibt dabei den Zustand, dass ein Dienstnehmer, auch Client genannt, die Dienste eines Dienstanbieters, auch Server genannt, nutzt. Zusätzlich ist der Client auch in der Lage selbst eine Server-Rolle bezüglich anderer Dienste oder zur Weiterverteilung selbst abonnierter Dienste übernehmen zu können. Aufgrund der Zugehörigkeit des Client-Server-Modells zur Netzwerktheorie erfolgt in Abschnitt 2.2.1.2 eine ausführliche Beschreibung dieses Modells. Das DBMS kann somit als Server verstanden werden, der Dienste wie Datenbankabfragen, -änderungen und -löschungen von anfragenden Clients, wie Benutzern an Computern, bearbeitet. Jedoch können von Implementierung zu Implementierung die eigentlichen Funktionalitäten variabel verteilt werden. Die Funktionalität einer Anwendung wird in drei Funktionsgruppen aufgeteilt, die stufenlos mehr client- oder serverseitig integriert werden können:

- Die Benutzerschnittstelle welche die Präsentation und Benutzerinteraktion weitergibt.
- Die Anwendungslogik oder auch „Business“-Logik genannt.
- Die Datenmanagementfunktionalität, einschließlich der Anfragebearbeitung und Transaktionskontrolle, welche Aufgaben wie Speichern, Ändern und ähnliches übernimmt.

Neben dieser Zwei-Schichten-Architektur von Client und Server existiert auch noch eine Drei-Schichten-Architektur, bei der die Anwendungslogik in einer eigenen Schicht realisiert wird (vgl. Abbildung 2.4). Diese zusätzliche Instanz, die zwischen Client und Server

existiert, wird Applikationsserver genannt[SSH10].

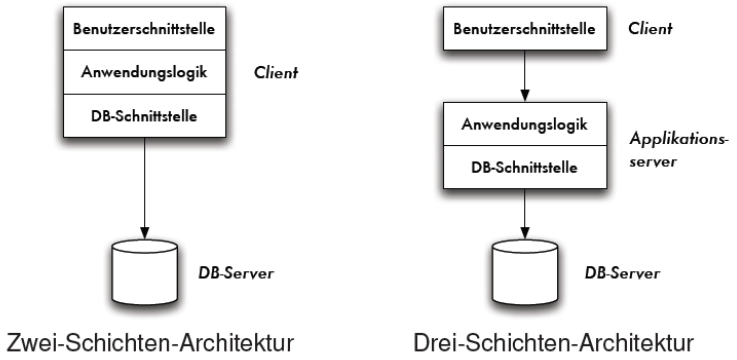


Abbildung 2.4.: Anwendungsarchitekturen (vgl. [SSH10])

2.1.2.3. ANSI-SPARC-Architektur

Die ANSI-SPARC-Architektur, welches auch unter Drei-Ebenen-Schema-Architektur bekannt ist, unterteilt das DBS in Komponenten, Bausteine, Elemente oder auch Werkzeuge und zeigt Schnittstellen sowie den Kommunikationsverlauf zwischen diesen auf. Zur klaren Abgrenzung werden Objekte der feineren Granularität als Elemente und Objekte mit gröberer Granularität als Komponenten bezeichnet. Es wird dabei in nachfolgende Komponenten unterschieden wie es auch in [oDBMS75] gezeigt wird:

- Die Definitionskomponente dient zur Datenbestimmung auf der konzeptuellen Ebene, zur Datendefinition der Dateiorganisation auf der internen Ebene und zur Sichtdefinition auf der externen Ebene.
- Mit der Programmierkomponente erfolgt die DB-Programmierung mit den bereitgestellten Datenbankoperationen, Einbettungen und Masken.
- Die Benutzerkomponente, die auch als Übergangskomponente bezeichnet wird, beinhaltet Anwendungsprogramme, die auf die Datenbank zugreifen sowie interaktive Anfrage- und Änderungswerkzeuge.
- Die Transformationskomponente wird für die Datentransformation von der internen zur externen Darstellung und umgekehrt genutzt. Dafür werden Werkzeuge zur Optimierung, Auswertung und für den Plattenzugriff zur Verfügung gestellt.
- Das Data Dictionary (Datenwörterbuch) ist eine zentrale Komponente des Systems, da es sämtliche Daten der Definitionskomponente aufnimmt und alle anderen Komponenten mit diesen Informationen versorgt.

Die genauen Beziehungen zwischen den einzelnen Elementen seien in Abbildung 2.5 veranschaulicht dargestellt.

2.1.2.4. Fünf-Schichten-Architektur

Die Transformationsschritte sind in der ANSI-SPARC-Architektur noch etwas ungenau und schemenhaft beschrieben sind, weshalb diese zur Fünf-Schichten-Architektur weiterentwickelt wurde. Die Fünf-Schichten-Architektur stellt eine detaillierte Beschreibung

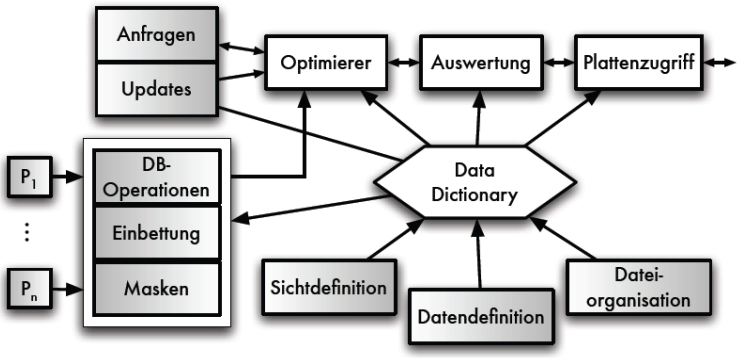


Abbildung 2.5.: ANSI-SPARC-Architektur (vgl. [SSH10])

der Transformationskomponente dar und konkretisiert die an der Transformation einer Anfrage oder Änderung teilnehmenden Elemente eines DBMS. Die Transformierung verläuft vom Datenbankmodell, also dem abstrakten konzeptuellen Schema, bis hin zum Speicherzugriff, also der physischen, internen Ebene. Die Fünf-Schichten-Architektur ist jedoch kein normierter Standard, sondern lediglich ein Industriestandard, welcher eine vieler möglicher Sichtweisen auf das System widerspiegelt[SSH10].

In Abbildung 2.6 werden die Schnittstellen und die einzelnen Elemente sowie deren Aufgabe gezeigt. Das Betriebssystem und die Geräteschnittstelle zum externen Speicher sind jedoch kein Teil eines DBS und werden deshalb nicht weiter betrachtet. Folgend noch eine kurze Begriffserklärung zu den einzelnen Elementen und Schnittstellen wie sie auch in [SSH10] gezeigt wird:

- Die Mengenorientierte Schnittstelle stellt die Datenabfrage- und Datenmanipulationssprache auf ganzen oder Teilen von Tabellen und Sichten zur Verfügung.
- Das Datensystem übersetzt und optimiert die Datenbankanfragen für die Satzorientierte Schnittstelle, unter Ausnutzung der Zugriffspfade und nimmt eine Auswertung der Relationenalgebra-Operatoren vor, da diese die Antwortzeiten auf eine Anfrage entscheidend beeinflussen.
- Die Satzorientierte Schnittstelle bewerkstelligt, unter Nutzung der Zugriffspfaden und den typisierten Datensätzen, den Zugriff auf die innere Darstellung der Relation.
- Das Zugriffssystem bildet die konzeptuellen Darstellungen auf die interne Darstellung ab.
- Die Interne Satzschnittstelle verwaltet die einzelnen Tupel. Außerdem sind die Speicherstrukturen der Zugriffspfade in ihr implementiert.
- Das Speichersystem setzt die Operationen der Internen Satzschnittstelle auf die virtuelle Adressen des Adressraums um. Die Anwendungsobjekte sind in ihrer internen Speicherdarstellung abstrakt als interne Datensätze sichtbar.
- Die Systempufferschnittstelle manipuliert den virtuellen Adressraum, welcher durch Seiten und Seitenadressen realisiert wird.
- Die Pufferverwaltung bildet die internen Seiten auf Blöcke der Dateischnittstelle des Betriebssystems ab und speichert die Daten im Primärspeicher zwischen.

- Die Dateischnittstelle operiert auf Blöcken des externen Speichers und gibt diese Aufgabe an das Betriebssystems weiter.

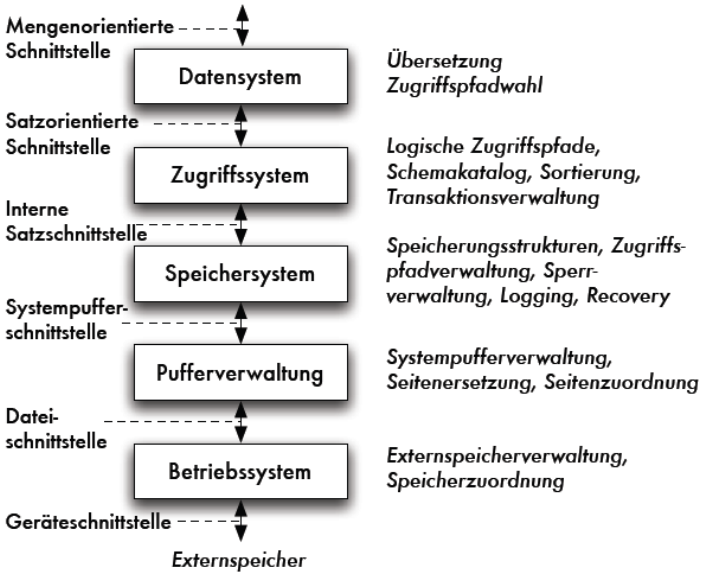


Abbildung 2.6.: Fünf-Schichten-Architektur (vgl. [SSH10])

2.1.2.5. Datenbanktransaktionen

Eine Transaktion ist eine Folge von Operationen, auch Aktionen genannt, welche die Datenbank von einem konsistenten Zustand in einen konsistenten, eventuell veränderten Zustand überführt. Da auf einer Datenbank mehrere Benutzer, Applikationen oder andere Instanzen potentiell zur gleichen Zeit auf dieselben Daten zugreifen können, müssen die Abläufe der einzelnen Transaktionen geregelt werden, damit anschließend die Datenbank in keinem fehlerhaften Zustand zurückgelassen wird. Die Überführung von einem konsistenten Zustand in einen anderen konsistenten Zustand, bei der Ausführung einer Transaktion muss dem ACID-Prinzip genügen[HR83].

„ACID“ ist ein Akronym für die vier folgenden englisch übersetzten Eigenschaften und setzt sich aus deren Anfangsbuchstaben zusammen wie es auch in [HR83] detailliert gezeigt wird:

- Atomicity (Atomarität) - Transaktionen dürfen nur als atomare Einheit ausgeführt werden, was soviel heißt, dass sie nur vollkommen oder überhaupt nicht ausgeführt werden dürfen. Somit entstehen keine Zwischenzustände, welche eventuell inkonsistent sein könnten.
- Consistency (Konsistenz) - Zur Integritätserhaltung darf eine Transaktion das System aus einem konsistenten Zustand nur in einen eventuell veränderten konsistenten Zustand überführen. Die Datenbank muss sowohl vor dem Beginn, als auch nach der Beendigung einer Transaktion jeweils in einem konsistenten Zustand sein.
- Isolation - Ein Nutzer, der mit einer Datenbank arbeitet, sollte stets den Eindruck haben, dass er mit dieser Datenbank alleine und unabhängig arbeitet.

- Durability (Dauerhaftigkeit) - Das Ergebnis einer erfolgreich ausgeführten Transaktion muss dauerhaft in der Datenbank gespeichert sein.

Beim gemeinsamen Zugriff auf Datenobjekte durch Transaktionen werden Sperren, auch Locks genannt, verwendet, welche eine Exklusivität gegenüber anderen zugreifenden Instanzen sicherstellen und es nicht zu Konflikten kommt. Bei der Ausführung einer Transaktion wird eine Sperre auf ein Objekt gesetzt, womit eine andere Transaktion erst bei der Aufhebung der Sperre auf das Objekt zugreifen kann. Es wird dabei im Groben zwischen Lese- und Schreibsperren unterschieden.

Ein weiterer Aspekt, welcher durch Sperren betrachtet werden muss, sind Deadlocks. Dies sind Verklemmungen, welche entstehen, wenn zwei oder mehrere unterschiedliche Objekte durch unterschiedliche Transaktionen gesperrt werden und diese auf die Endsperrung der jeweils anderen Transaktion warten, um auf das gesperrte Objekt zugreifen zu können. Dieser Umstand muss vom System erkannt und aufgelöst werden, da es sonst zu Fehlern in der Datenbank kommt.

2.1.3. Structured Query Language

In diesem Abschnitt werden nur die wichtigsten SQL-Befehle, welche zum Verständnis der Arbeit benötigt werden, kurz erklärt. Als weiterführende Literatur sei deshalb auf [Kri08] und [SSH10] verwiesen. „Structured Query Language, kurz SQL genannt, ist eine Datenbanksprache für relationale DBMS und beinhaltet die vier verschiedene Typen von Datenbanksprachen, welche zur Verwaltung einer Datenbank genutzt werden. Dazu zählt die „Data Control Language“ (DCL), „Data Definition Language“ (DDL), „Data Manipulation Language“ (DML) und „Data Query Language“ (DQL). Durch diese vier Typen ist es möglich, Daten zu verändern, zu definieren und abzufragen sowie Berechtigungen zu vergeben. Darüber hinaus bietet SQL die Anwendung von relationaler Algebra und deren Operationen sowie zusätzliche Funktionen, wie „MIN“, „MAX“, „SUM“, „COUNT“ und mehr zur Aggregation, als auch einfache arithmetische Operationen wie Addition, Subtraktion, Multiplikation und Division.

2.1.3.1. SQL-Anfrageoperationen

Eine typischer SQL-Befehl zur Abfrage von Daten einer Datenbank sieht wie folgt aus:

```
select Attributliste
from Tabelle
where Bedingung A = B
group by Attributliste
having Aggregatfunktionen
order by Attribut
```

Das Schlüsselwort „SELECT“ gibt die Auswahl von Spalten durch Angabe einer Attributliste an, auch Projektion genannt an. Für den Fall, dass alle Attribute einer Tabelle ausgewählt werden sollen, so bietet sich die Verwendung von „*“ an. Mit dem Schlüsselwort „DISTINCT“ ist es außerdem möglich Tupel-Duplikate auszublenden.

Nach dem Schlüsselwort „FROM“ steht die Tabelle, auf welcher die Abfrage erfolgen soll. Dies kann auch auf dem Verbund mehrerer Tabellen geschehen.

Mit dem Schlüsselwort „WHERE“ wird auf die Auswahl von Zeilen einer Tabelle anhand einer Bedingung, auch Selektion genannt, festgelegt. Mehrere Bedingungen können mit „AND“ Verbunden werden oder mit „OR“ können verschieden alternative Bedingungen zur Selektion angegeben werden.

Die GROUP-BY-HAVING-Klausel ist ähnlich wie die WHERE-Klausel, welche die Ausgabe begrenzt. Die GROUP-BY-HAVING-Klausel schränkt jedoch auf Basis von Aggregatfunktionen ein, was mit der where-Klausel nicht möglich ist. Nach „GROUP BY“ steht die Attributliste und es werden alle Zeilen, welche die gleichen Werte für diese Attributliste enthalten, in jeweils einer Gruppe zusammengefasst. Mit „HAVING“ erfolgt dann die Beschränkung auf Basis der angebenden Aggregatfunktionen.

Abschließend kann das Ergebnis der Abfrage noch mit „ORDER BY“ nach einem oder mehreren Attributen sortiert werden.

2.1.3.2. SQL-Änderungsoperationen

Um eine bereits bestehende Datenbanktabelle bearbeiten zu können, sind Änderungsoperationen nötig. SQL bietet hierfür die Operationen „INSERT“, „UPDATE“ und „DELETE“ an.

Mit „UPDATE“ wird ein bereits vorhandener Datensatz aktualisiert, die Syntax sieht wie folgt aus:

```
update Basisrelation
set Attribut1 = Wert1
...
AttributN = WertN
[ where Bedingung ]
```

Nach „UPDATE“ steht der Name der Basisrelation, in der sich der Datensatz befindet. Nach „SET“ bekommen die ausgewählten Attribute den neuen Wert zugewiesen und mit dem optionalen „WHERE“ werden die Tupel ausgewählt auf welche die Bedingungen zutreffen.

Mit „INSERT“ wird ein neuer Datensatz hinzugefügt, die Syntax sieht wie folgt aus:

```
insert
into Basisrelation
[ (Attribut1, ..., AttributN) ]
values (Wert1, ..., WertN)
```

Nach „INSERT INTO“ steht der Name der Basisrelation, in welche der neue Datensatz eingefügt werden soll. Die Angabe der Attributliste ist optional und mit „VALUES“

werden die neuen Werte zugewiesen.

Mit „DELETE“ lassen sich ein oder mehrere bereits bestehende Datensätze löschen, die Syntax sieht wie folgt aus:

```
delete
from basisrelation
[ where bedingung ]
```

Nach „DELETE FROM“ steht der Name der Basisrelation, in welcher sich die zu löschenden Datensätze befinden und nach dem optionalen „WHERE“ stehen die Bedingungen für die zu löschenden Datensätze.

2.2. Internet und Webanwendungen

Im nachfolgenden Abschnitt werden die Grundbegriffe und die Grundzüge zur Funktionsweise des Internets näher erläutert. Da es jedoch nicht erforderlich ist und den Rahmen dieser Arbeit sprengen würde, wird die Funktionsweise des Internets nur soweit erklärt, wie es für das Verständnis der Arbeit notwendig ist. Für tiefgründigeres Wissen über die Funktionsweise von Netzwerken sei auf das Buch „Computernetzwerke“ von Andrew Tanenbaum [Tan12] verwiesen.

2.2.1. Das Internet

Das Internet oder auch World Wide Web (WWW) genannt, ist ein weltweites Netzwerk von Computern und Rechnernetzwerken, welches die Übermittlung von Daten ermöglicht. Es ist eine Weiterentwicklung des ARPANET, einem Projekt der Advanced Research Project Agency (ARPA) des US-Verteidigungsministeriums zur Vernetzung von Universitäten und Forschungseinrichtungen.

Die Grundlagen des Internets wurden 1989 vom Informatiker Tim Berners-Lee am Genfer Institut für Teilchen-Physik entwickelt. Dieser entwickelte einen neuen Netzdienst, der auf dem Netzwerk des CERN (European Organization for Nuclear Research) basiert. Dieser Netzdienst erleichterte den Austausch von wissenschaftlichen Dokumentationen über Ländergrenzen hinaus. In der Folgezeit entwickelte er außerdem die Seitenbeschreibungssprache „Hypertext Markup Language“ (HTML), das Hypertext Transfer Protocol (HTTP), die URL, den ersten Browser „WorldWideWeb“ und den ersten Webserver „CERN httpd“. Erst im Jahr 1990 wurde das Internet jedoch über die Universitäten hinaus für die Öffentlichkeit zugänglich gemacht, nachdem die US-amerikanische National Science Foundation beschloss, das Internet für kommerzielle Zwecke nutzbar zu machen[Sal95].

2.2.1.1. Netzwerkinfrastruktur

Ein Computernetzwerk besteht aus mindestens zwei miteinander verbundenen Rechnern und die Nutzer des Netzwerkes können miteinander kommunizieren und gemeinsame

Ressourcen nutzen. Für die Kommunikation spielt es keine Rolle, ob die Rechner die gleiche Hardware oder das gleiche Betriebssystem nutzen.

In einem Netzwerk agieren die Rechner als Client, Server oder gar beides. Jedem Rechner ist innerhalb des Netzwerkes eine eindeutige Adresse, über die er sich identifizieren lässt, zugeordnet. Um das Netzwert nutzen zu können sind diese mit Netzwerkkarten ausgestattet und über einen Switch untereinander verbunden.

Das Internet gilt als ein weltweites Netz von autonomen Computernetzwerken. Zugang zum Internet erhält ein Nutzer, indem er sich über eine Telefonleitung bei einem Internetprovider oder Online-Dienst einwählt. Der Einwahlknoten eines Providers wird mit Point of Presence (POP) bezeichnet, der wiederum über einen Network Access Point (NAP) ein Backbone-Netz anbindet, das die schnelle Verbindung der Netze garantiert[Pom12].

2.2.1.2. Client-Server-Modell

Das Client-Server-Modell ist das Standardkonzept für die Verteilung von Aufgaben innerhalb eines Netzwerks. Zur Begriffsklärung ist zu erwähnen, dass Client und Server sowohl für Software als auch für Hardware Verwendung finden. Ein Web-Server ist eigentlich ein Programm, das auf die Kontaktaufnahme eines Clients wartet, um eine bestimmte Dienstleistung für ihn zu erfüllen. Die Bezeichnung Server ist aber auch für den Rechner zutreffend, auf dem diese Software läuft. Mit dem Begriff Client verhält es sich ähnlich. Damit wird sowohl der mit dem Server verbundene Rechner bezeichnet als auch die Software, welche die Anfrage an den Server verschickt[Pom12].

Die Aufgaben werden vom Server auf verschiedene Rechner verteilt. Bei den Aufgaben, welche auch als Dienst bezeichnet werden und vom Server angeboten werden, handelt es sich beispielsweise um den Versand und Empfang von E-Mails, dem Zugriff auf Webseiten oder gar spezifische Anforderungen spezieller Software.

Der Client ist in der Lage, diesen Dienst aktiv zu nutzen indem er diese vom Server anfordert. Die Kommunikation zwischen Client und Server ist abhängig vom Dienst, das heißt, der Dienst bestimmt, welche Daten zwischen beiden ausgetauscht werden. Der Server ist dazu jederzeit in Bereitschaft und wartet passiv auf die Anforderungen eines Clients um diese bearbeiten zu können. Die Regeln der Kommunikation für einen Dienst, wie Format, Aufruf des Servers, Bedeutung der zwischen Server und Client ausgetauschten Daten, werden durch ein für den jeweiligen Dienst spezifisches Protokoll festgelegt[Pom12].

Eine Software, welche für ihre Aufgaben und Funktionen vom Client-Server-Modell Gebrauch macht, wird als Client-Server-System bezeichnet. Dieses System besteht mindestens aus zwei Komponenten, der Server- und der Client-Komponente, die in der Regel auf verschiedenen Rechnern ablaufen.

2.2.1.3. Datenübertragung im Internet

Im Internet werden Dokumente in der Auszeichnungssprache HTML von Servern bereitgestellt. Ein Web-Browser, welcher die Rolle des Clients einnimmt, kann diese Dokumente vom Server anfordern. Der Server übermittelt die angeforderten Daten, welche anschließend vom Browser der HTML-Spezifikation entsprechend auf dem Client-Rechner dargestellt werden. Bei Webbrowsern erfolgt das Anfordern von Daten meist über die URL, welche sich aus dem Protokoll, der Serveradresse und der Pfad- und Dateiangabe zusammensetzt.

Die Kommunikation zwischen Server und Client wird über das Hypertext Transfer Protocol (HTTP) festgelegt. Bei diesem Protokoll wird zunächst eine TCP/IP-Verbindung zwischen Server und Client aufgebaut. Bei der Übertragung werden die Daten, welche zwischen Client und Server übertragen werden, in Pakete aufgeteilt. Das Transmission Control Protocol (TCP) ist für die Aufteilung der Daten beim Sender und für die Wiederzusammensetzung beim Empfänger verantwortlich und das Internet Protocol (IP) regelt die Zustellung der einzelnen Datenpakete[Tan12].

Nachdem die Verbindung steht, übermittelt der Client seine Anfrage an den Server über den „GET“-Befehl. Dieser verarbeitet die Anfrage vom Client und sendet ihm daraufhin die Antwort über die TCP/IP-Verbindung. Abschließend wird dann die TCP/IP-Verbindung nach kompletter Übertragung der Daten vom Server getrennt[Tan12].

Um eine Verbindung zwischen Client und Server jedoch zu ermöglichen, muss jedem Internetteilnehmer eine eindeutige Adresse zugewiesen werden. Dazu dient die auf dem IP-Protokoll basierende IP-Adresse. Die IP-Adresse teilt sich in vier 8-Bit Felder auf, welche durch einen Punkt voneinander getrennt und als Dezimalwert angegeben werden. Dieses Nummerierungsschema wird auch als IPv4-Adresse bezeichnet, dabei stehen die ersten drei Felder für die Netzwerkkennung und das letzte Feld für die Rechnernummer. Da jedoch mit zunehmender Teilnehmerzahl am Internet die verfügbaren Adressen knapp werden, wurden die IPv6-Adresse eingeführt, welche den möglichen Adressraum von $4{,}3*10^9$ auf $7{,}9*10^28$ vergrößert[Tan12].

2.2.2. Webanwendungen

Eine Webanwendung, auch Webapplikation oder kurz Web-App genannt, liegt auf einem Webserver vor auf dem über das Internet oder ein Intranet zugegriffen werden kann. Über einem Webbrowser lässt sich die Webanwendung anzeigen und bedienen. Sie ermöglicht eine dynamische Interaktion zwischen Nutzer und Server, welche je nach Anwendung vollständig auf dem Client ausgeführt wird oder teils auf dem Client und teils auf dem Server.

2.2.2.1. Clientseitige Dynamik

Mit Hilfe von JavaScript und Document Object Model (DOM) können Webseiten clientseitig ohne Neuladen der Webseite verändert werden. Der Zugriff auf alle Elemente eines Dokuments wird über die Implementierung des DOM realisiert. Skriptsprachen

wie JavaScript werten Benutzereingaben in HTML-Formularfeldern aus und können darauf basierend eine Modifikation des Dokuments vornehmen[Pom12].

Eine weitere Möglichkeit für clientseitige Dynamik lässt sich durch Java-Applets verwirklichen. Java-Applets sind kleine browserunabhängige Java-Programme mit Sicherheitseinschränkungen und benutzen die Implementierung der Programmiersprache. Zur Ausführung muss der Browser jedoch über eine virtuelle Java-Maschine verfügen, wodurch die Java-Applets in einem vom Browser reservierten Bereich ausgeführt werden[Pom12].

2.2.2.2. Serverseitige Dynamik

Serverseitige dynamische Webseiten werden erst mit der Anfrage des Clients an dem Server erzeugt und dann als HTML-Datei dem Webbrowser übermittelt. Hierbei ist zu unterscheiden zwischen Skriptsprachen, die als Servererweiterung anzusehen sind und Programmen auf dem Server, die über das Common Gateway Interface (CGI) mit dem Server kommunizieren.

Das CGI-Programm erhält zusammen mit der URL und weiteren Parametern, welche zum Beispiel aus HTML-Formularen stammen, alle erforderlichen Daten, um eine dynamisch generierte HTML-Seite bereitzustellen. Es schreibt außerdem die Kommunikationsregeln zwischen Client und Server fest. Mit einem CGI-Programm besteht zudem die Möglichkeit, auf die Dienste eines Datenbankservers zurückzugreifen.

Eine Alternative zu CGI-Programmen stellen Skriptsprachen wie PHP dar. Hierbei übermittelt der Browser die Anfrage nach einem PHP-Script oder einer HTML-Datei zusammen mit den benötigten Parametern. Der Web-Server lädt das Skript, beziehungsweise die HTML-Datei mit eingebettetem PHP und stellt die übermittelten Parameter als Umgebungsvariablen dem Skript zur Verfügung. Über das Skript wird die HTML-Ausgabe generiert und an den Client zurückgesandt[Pom12].

2.3. Grundlagen der Webprogrammierung

In diesem Grundlagenkapitel werden alle Technologien, welche für die Erstellung des Webtools verwendet wurden, näher erläutert.

2.3.1. HTML

HTML ist eine textbasierte Auszeichnungssprache, welche die Struktur einer Webseite zum Anzeigen in einem Webbrowser enthält. Sie bildet zusammen mit der deklarativen Stilsprache Cascading Style Sheets (CSS) die fundamentalen Technologien zur Entwicklung und Gestaltung von Webseiten. CSS ist für das Erscheinungsbild oder auch Layout genannt, einer Webseite verantwortlich. In diesem Kapitel wird jedoch nur HTML behandelt, denn es war für die Erstellung des Webtools nicht notwendig ein benutzerdefiniertes Layout zu erstellen. Das Layout wird später an das des GSBLs angepasst. Die aktuelle

Version von HTML, welche auch für die Erstellung des Webtools genutzt wurde, ist die HTML5-Version.

2.3.1.1. Dokumentenstruktur und Metatags

Ein HTML-Dokument gliedert sich in die vier folgenden Bereiche, Dokumententyp-Deklaration, HTML-Root-Tag, den Head-Abschnitt und den Body-Abschnitt.

Am Anfang erfolgt zunächst die Dokumententyp-Deklaration. Hier wird angegeben nach welchem Standard die Webseite erstellt wurde und der Root-Tag umschließt das gesamte Dokument. Im Head-Abschnitt werden Informationen wie der Seitentitel, Stilvorlagen, Skripte, Referenzen auf externe Dateien und weitere Metainformationen angegeben. Diese Informationen werden jedoch nicht vom Browser dargestellt. Der Körper oder auch Body-Abschnitt genannt, eines HTML-Dokuments enthält alle Elemente, welche für die Gestaltung des Inhalts, wie Text, Hyperlinks, Bilder, Tabellen, Listen und mehr zuständig sind. Diese werden dann vom Browser entsprechend der CSS-Stilvorgaben dargestellt.

Ein HTML-Element besteht aus Auszeichnungen, den sogenannten Tags, den Attributen und dem Inhalt. Alle HTML-Auszeichnungen befinden sich dabei zwischen den spitzen Klammern „<“ und „>“. Jeder Tag hat dabei einen Start-Tag und einen Ende-Tag, wobei der Ende-Tag dem Anfangstag entspricht, jedoch mit einem vorgestellten „/“ nach „<“. Auch zulässig ist dabei die Schreibweise, dass Start- und Ende-Tag in einem geschrieben wird, indem „/“ vor „>“ steht. Ein gutes Beispiel hierfür ist der Zeilenumbruch „
“. Bei HTML wird zwar nicht zwischen Groß- und Kleinschreibung unterschieden, jedoch wird immer die Kleinschreibung bei der Erstellung der Tags genutzt. Ein Tag kann Attribute enthalten, dabei wird den Attributen über den Zuweisungsoperator „=“ ein Wert zugewiesen. Das Verwenden von mehreren Attributen innerhalb eines Tags ist zulässig, diese müssen aber durch ein Leerzeichen getrennt werden. Die Attributwerte befinden sich immer in doppelten Hochkommata, diese Schreibweise ist seit HTML5 jedoch nicht mehr zwingend vorgeschrieben. Die Angabe der Landessprache ist ein Attribut im HTML-Tag und gehört zu den Universalattributen, ebenso wie eine Identität „id“, die als eindeutiger Bezeichner für ein Element benutzt wird. Alle HTML-Auszeichnungen müssen sich zwischen den HTML-Tags befinden[Pom12].

```
<html lang="de">
...
</html>
```

Innerhalb der Head-Tags ist ein Title-Tag vorgeschrieben, dessen Inhalt im Titelbalken des Anwendungsfensters vom Browser erscheint.

```
<head>
<title>Text der Titelzeile</title>
</head>
```

Die Meta-Tags liefern die Meta-Informationen über Inhalt und Autor der Webseite und bieten Hinweise für Suchmaschinen an.

```
<meta charset="utf-8">
<meta name="description" content="Struktur einer Webseite" />
<meta name="keywords" content="HTML, Metatags" />
<meta name="author" content="gp" />
```

Im Body-Abschnitt befinden sich die Anweisungen für den Browser zum Aufbau der eigentlichen Webseite.

```
<body>
...
<body>
```

Innerhalb des Body-Tags können die Elemente gruppiert werden. Bisher bestand lediglich die Möglichkeit mit den Tags Div-Tags „ “ und „ “ sowie den Span-Tags „“ und „“ die sogenannten Container einzurichten. Mit dem Class-Attribut und dem ID-Attribut können diese Container eindeutig referenziert werden. Seit HTML5 wurden weitere Elemente zur Strukturierung eingeführt, die ein semantisches Markup unterstützen und somit auch Class-Attribute verzichtbar machen. Hierzu gehören die logischen Bereiche einer Seite wie Kopfbereich, Navigation, Fußsbereich und Abschnitte für Artikel[Pom12].

Ein Kommentar in HTML lässt sich nur im Quellcode selbst einsehen und kann sich über mehrere Zeilen erstrecken. Dies sei am folgenden Beispiel gezeigt.

```
<!--
Das ist ein Kommentar
-->
```

Im Gesamten betrachtet kann ein einfaches HTML-Grundgerüst somit folgendermaßen aussehen.

```
<!DOCTYPE html>
<!-- Dokument htmlcss/index0.html -->
<html lang="de">
<head>
<meta charset="utf-8">
<meta name="description" content="Struktur einer Webseite" />
<meta name="keywords" content="HTML, Metatags" />

<meta name="author" content="gp" />
<title>HTML Living Standard</title>
</head>
<body>
<header> Der Kopfbereich einer Seite </header>
<section> Der Inhaltsbereich </section>
<footer> Die Fusszeile </footer>
</body>
</html>
```

2.3.1.2. Textstrukturierung

Text ist in erster Linie gegliedert durch Überschriften und Abschnitte. Innerhalb des Textes gibt es Hervorhebungen, Streichungen und anderweitig zu gestaltende Bereiche wie Listen, Adressen, Programmquelltext oder Einrückungen[Pom12].

Überschriften werden als Headings mit den Tags „<h1>“ und „</h1>“ bezeichnet. Die 1 steht für die Wertigkeit der Überschrift. Insgesamt gibt es sechs Wertigkeiten, wofür die Zahlen von 1 bis 6 genutzt werden. Der eigentliche Text wird in Abschnitten, welche auch als auch Paragraphen bezeichnet werden, unterteilt.

```
<p>
Erster Abschnitt
</p>
<p>
Zweiter Abschnitt
</p>
```

Eine grafische Unterteilung kann mit horizontalen Linien gemacht werden. Der Tag wird mit „<hr/>“ ausgezeichnet. Ein Zeilenumbruch wird mit dem Tag „
“ erzwungen. Zu Beachten ist, dass dies ein harter Zeilenumbruch ist, darüber hinaus werden durch den Browser auch automatisch Zeilenumbrüche realisiert, je nach Bildschirmauflösung und Fenstergrößse[Pom12].

Weitere Textstrukturierungsmöglichkeiten, welche zu nennen sind, sind Streichungen mit „<del>“, der Einrückung „<blockquote>“ und einer Vorformatierung „<pre>“. Im vorformatierten Bereich wechselt der Zeichensatz und die Leerzeichen werden nicht mehr entfernt[Pom12].

HTML stellt für Aufzählungslisten mit „<ul>“, nummerierte Listen „<ol>“ und Definitionslisten „<dl>“ die benötigten Tags bereit. Bei Aufzählungslisten und nummerierten Listen werden die Listenelemente in den Tag „<li>“ eingebunden. Definitionslisten unterteilen sich in den zu definierenden Ausdruck mit „<dt>“ und der Definition „<dd>“ selbst.

2.3.1.3. Tabellen

Tabellen sind ein Beschreibungselement innerhalb von HTML, mit denen Daten in Zellen positioniert werden. Vor der Einführung von CSS wurden Tabellen als universelles Gestaltungsraster benutzt. Heute finden Tabellen nur in der ursprünglichen Bedeutung Anwendung.

Eine Tabelle wird durch den Tag „<table>“ eingeleitet. Bestandteile der Tabelle sind dabei der Tabellenkopf „<thead>“ mit den Spaltenüberschriften „<th>“, die Tabellenzeilen „<tr>“ und Tabellenzellen „<td>“ sowie eine Zusammenfassung im Fußbsereich „<tfoot>“. Die Kombination der Tags „<thead>“ und „<table>“ „<tfoot>“ wird im Zusammenhang mit dem Tag „<tbody>“ benutzt. Die Tabellen beginnen immer links oben und werden dann zeilenweise mit den Definitionen von

Tabellenzellen aufgefüllt.

Es wird dabei zwischen zwei Tabellenarten unterschieden, den regelmäßigen und den unregelmäßigen Tabellen. In unregelmäßigen Tabellen können sich Datenzellen über mehrere Spalten, angesprochen durch das Attribut „colspan“ oder über mehrere Zeilen, angesprochen durch das Attribut „rowspan“, ausdehnen. Die Anzahl der belegten Spalten beziehungsweise Zeilen wird hierbei durch die Attributswerte definiert[Pom12].

Wichtige Angaben zur Gestaltung einer Tabelle sind der Abstand zwischen den Zellen, welche durch das Attribut „cellspacing“ angegeben werden sowie die Polsterung, das ist der Abstand der Zelleninhalte zum Rand, welche durch das Attribut „cellpadding“ angegeben wird. Jedoch sollten mit Verwendung von HTML5 diese Attribute nicht mehr verwendet werden, denn in HTML5 wird nur noch das Attribut „border“ für die Gestaltung der Zellenränder unterstützt. Alle weiteren Attribute werden in der CSS Stilvorlage definiert[Pom12].

2.3.1.4. Formulare

Interaktive Benutzereingaben auf Webseiten werden über Formulare vorgenommen. Neben den Bedienelementen, wie Checkboxen, Radiobuttons, Eingabefeldern und mehr stellen die Formulare auch die Kommunikationsschnittstelle zwischen Anwender und Web-Server her.

Formulardaten können sowohl clientseitig mit JavaScript als auch serverseitig mittels dort laufender CGI-Programme ausgewertet werden. Serverseitig werden die übermittelten Daten ausgewertet oder als Umgebungsvariablen den Skripten beziehungsweise Programmen zur weiteren Verarbeitung verfügbar gemacht. Ein Formular wird mit dem Tag „<form>“ eingeleitet. Die im Tag definierten Attribute legen die nach der Eingabebestätigung auszuführende Aktion fest. Innerhalb der Formulare gibt es die Tags „<input>“ mit Type-Attribut für unterschiedliche Datentypen, „<textarea>“ und „<select>“. Die Positionierung der Formularelemente lässt sich dabei gut mit Tabellen vornehmen[Pom12].

Zur Versendung der Formulardaten kann im „<form>“-Tag das Attribut „action“ genutzt werden. Der Wert von action kann zum Beispiel eine URL sein, die eine Webseite mit Anweisungen zur Auswertung der übermittelten Daten angibt. Die Art der Datenübermittlung wird im Attribut method durch post oder get festgelegt. Die Get-Methode hängt die Parameter sichtbar an die gesendete URL an und begrenzt die Datenmenge. Vorzuziehen ist die Post-Methode, die Daten unabhängig von der URL übermittelt und größere Informationsinhalte überträgt[Pom12].

Um die Formularelemente eindeutig identifizieren zu können, werden zwei weitere Parameter von „<form>“ genutzt, nämlich die Werte für „name“ und „id“, deren Werte frei vom Programmierer zu vergeben sind.

2.3.1.5. Hyperlinks

Hyperlinks oder kurz Links sind Verweise in einem HTML-Dokument, mit denen bei Aktivierung durch Mausklick eine andere Position im Dokument, eine neue Datei oder ein neuer Web-Server aufgerufen wird.

Sensitive Bereiche für Hyperlinks können Texte, Grafiken, Bereiche in Grafiken oder Buttons sein. Diese genannten Bereiche werden auch als Schaltflächen bezeichnet. Ein Hyperlink ist mit dem Anchor-Tag „<a>“ definiert und benötigt zur Auszeichnung als sensitives Element das wichtigste Attribut „href“, welches für Hypertextreferenz steht. Die Elemente innerhalb des Anchor-Tags stellen dabei die Schaltfläche dar, wofür standardmäßig ein Text verwendet wird, der als Link ausgezeichnet ist und in blauer und unterstrichener Farbe dargestellt wird.

Dem Attribut „href“ wird die aufzurufende Referenz zugewiesen. Die Anzeige kann im eigenen Fenster oder in einem neuen Fenster erfolgen. Wertzuweisungen an das Attribut „target“ sind „_ blank“ für das Öffnen in einem neuen Fenster beziehungsweise „_ self“ für das Öffnen im selben Fenster[Pom12].

Die Referenz auf eine neue Datei kann dabei absolut durch Angabe der vollständigen Pfadangabe, als auch relativ zur aktuellen Seite erfolgen. Eine neue Datei kann zum Beispiel folgendermaßen durch die relative Pfadangabe referenziert werden.

```
<a href=" ../pfad/neuedatei.html#kap1" target ="_blank">zur neuen Seite</a>
```

Die beiden Punkte im Link bedeuten, dass in das Elternverzeichnis der aktuellen Seite gegangen wird, um von dort den Dateipfad „pfad/neuedatei.html# kap1“ entlangzugehen, zum Laden der neuen Datei[Pom12].

2.3.2. JavaScript

JavaScript stellt eine objektbasierte Skriptsprache dar und ergänzt die Funktionalität von Web-Browsern, da diese den Inhalt einer Webseite nur statisch abbilden können. Durch JavaScript lassen sich jedoch Elemente durch Benutzereingriffe dynamisch verändern, ohne eine Seite neu hoch zu laden.

Plattformunabhängigkeit besteht dadurch, dass JavaScript vom Web-Browser interpretiert wird, da die Interpretationen von JavaScript jedoch von von Browser zu Browser abweicht, erfordert dies meist den Test der Skripte mit mehreren Browsern oder die aufwendige Programmierung von Browserweichen oder Ausweichlösungen. Mit den modernen Browsern und der Annäherung an Standards wie dem ECMA-262 wird dieses Problem zukünftig an Bedeutung verlieren[Pom12].

2.3.2.1. Syntax und Einbindung in HTML

Die Syntax von JavaScript ist in vielen Fällen gleich mit der Syntax der objektorientierten Programmiersprache Java, jedoch sind beide Sprachen nicht in Verbindung zu

setzen. JavaScript unterscheidet zwischen Großs- und Kleinschreibung. Namen können aus Ziffern, Buchstaben und Unterstrich bestehen, weitere Sonderzeichen sind jedoch nicht zulässig. Des Weiteren darf das erste Zeichen eines Namens Ziffer sein. Leerraum im Quelltext wird weitgehend ignoriert[Pom12].

Mit dem Schlüsselwort „VAR“ werden die Variablen deklariert, der Datentyp wird dabei implizit über den Wert zugewiesen. Die Zuweisung eines Startwertes ist bei der Deklaration nicht notwendig, sondern kann auch später erfolgen.

Funktion werden mit dem Schlüsselwort „FUNCTION“ deklariert, in runden stehen die übergebenen Eingabevariablen, während in den geschweiften Klammern der Anweisungsblock eingeschlossen wird. Innerhalb der Funktionen können dann alle gängigen Programmierkonstrukte wie bedingte Anweisungen und Schleifen, wie sie auch in Java vorkommen, verwendet werden.

Zur Einbindung von JavaScript in HTML-Programmcode bestehen mehrere Möglichkeiten, wie das Einbinden innerhalb eines HTML-Tags als unmittelbare Anweisung, mittels „<script>“ Auszeichnung als Block innerhalb einer HTML-Seite, im Kopf einer HTML-Seite oder als externe Datei referenziert[Pom12].

Die Programmanweisungen von JavaScript werden mit dem HTML-Tag „<script type = “text/javascript“>“ und dem Type-Attribut von der eigentlichen HTML-Auszeichnung getrennt. JavaScript-Anweisungen innerhalb eines Script-Blocks werden dann ausgeführt, wenn der Browser beim Laden diese Anweisungen erreicht. Mit dem Type-Attribut wird die Sprachversion spezifiziert. Die Anweisungen des Blocks sollten als Kommentar gekennzeichnet sein, damit der Browser gegebenenfalls diese Anweisungen überlesen kann.

Ist der JavaScript-Code hingegen innerhalb einer Funktion definiert, dann muss diese explizit aufgerufen werden. Der Aufruf erfolgt in der Regel ereignisgesteuert. Ereignisse können beispielsweise das Laden einer Seite „ONLOAD“ oder ein Mausereignis „ONMOUSEOVER“, „ONCLICK“ sein. Die Anweisungen der Funktion werden üblicherweise im Kopf der HTML-Seite eingetragen, sofern hier nicht Elemente referenziert werden, die noch nicht zur Verfügung stehen[Pom12].

Die Auslagerung von JavaScript in externe Dateien ist der Einbettung nach Möglichkeit vorzuziehen. Der Programmcode kann somit auch mehrfach verwendet werden und es kann eine umfangreiche Skriptbibliothek unter der Erweiterungsbezeichnung „.js“ erstellt werden. Mit dem Attribut „src“ im Script-Tag wird die Datei referenziert, die dann aber selbst nicht mehr den Script-Tag enthalten darf.

2.3.2.2. Elemente in JavaScript

JavaScript ist objektbasiert und dient zur Konstruktion benutzerdefinierter Objekte. der Nutzung von vordefinierten Objekten und dem Zugriff auf das DOM bedient man sich der Elemente der Programmiersprache, welche im folgenden erläutert werden.

Ein Programm ist eine Folge von Befehlen, deren Abarbeitung durch Kontrollstrukturen steuerbar ist. Jede Anweisung muss mit einem Semikolon abgeschlossen werden und es wird unterschieden in einfache Anweisungen wie

```
Variable = 4;
```

und Verbundanweisungen oder auch Blöcke genannt, die der Gruppierung dienen. Etwa ein Konstrukt wie

```
if (Bedingungen) then {
// Anweisungsblock 1
} else {
// Anweisungsblock 2
}
```

In Ausdrücken werden Operanden mit Operatoren verknüpft. Als Beispiel für einen arithmetischen Ausdruck kann die Summe einer Variablen mit einer Konstanten angeführt werden.

```
Summe = Wert + 5;
```

Die Anweisung ist nicht als mathematische Gleichung zu verstehen. „Summe“ ist ein Bezeichner für eine Variable, das Gleichheitszeichen ist der Zuweisungsoperator. Auf der rechten Seite der Anweisung befindet sich ein Ausdruck. Auf die Variable mit dem Bezeichner „Wert“ soll die Konstante 5 addiert werden. Das erledigt der Additionsoperator. Nach Auswertung des Ausdrucks erfolgt die Zuweisung an die Variable „Summe“. Der Bezeichner ist ein vom Programmierer frei zu vergebender symbolischer Name für die Variable, unter dem der Speicherplatz innerhalb des Programms aufgerufen wird[Pom12].

JavaScript gilt als typenlose Sprache und es werden Variablen vom Typ „Zahl“, „Zeichenketten“ und „boolesche Werte“ erkannt. Darüber hinaus existieren noch die Typen „null“, „undefined“ und „object“. Der Datentyp einer Variablen kann sich innerhalb des Programmablaufs auch verändern. Mit dem Schlüsselwort „VAR“ wird die Deklaration von Variablen eingeleitet[Pom12].

Die Verknüpfung der Variablen, auch Operanden genannt, erfolgt durch Operatoren. Diese sind zu unterscheiden in arithmetische Operatoren, Zuweisungsoperatoren, Vergleichsoperatoren, logische Operatoren und String-Operatoren. Die Reihenfolge der Auswertung entspricht den mathematischen Regeln und kann durch runde Klammerpaare beeinflusst werden. In der Tabelle 2.1 wird eine anschauliche Übersicht zu allen gängigen Operatoren gegeben.

Ein weiteres Element von JavaScript sind Arrays. Bei Arrays handelt es sich um Datenfelder beliebiger Datentypen, mit denen mehrere Elemente unter einem Namen verwaltet werden. Die Adressierung eines bestimmten Wertes erfolgt bei indizierten Arrays über den Index oder bei assoziativen Arrays über den nicht-numerischen Schlüssel. Arrays werden über die literale Notation oder über den Array-Konstruktor erzeugt. Zur Deklaration eines Arrays besteht ebenfalls die Möglichkeit den Konstruktor des Objekts

Arithmetische Operatoren	
+	Addition
-	Subtraktion
*	Muliplikation
/	Division
%	Modulo, Rest einer Ganzzahl
++	Inkrement, erhöht Ganzzahl
–	Dekrement, verringert Ganzzahl
-	Vorzeichenwechsel
Zuweisungsoperatoren	
=	einfacher Zuweisungsoperator
+=	als alternative Schreibweise von a=a+b und a+=b
-=	als alternative Schreibweise von a=a-b und a-=b
*=	als alternative Schreibweise von a=a*b und a*=b
/=	als alternative Schreibweise von a=a/b und a/=b
%=	als alternative Schreibweise von a=a%b und a%=b
Vergleichsoperatoren	
==	Gleichheit zweier Ausdrücke
!=	ungleich
>	größer
>=	größer gleich
<	kleiner
<=	kleiner gleich
Logische Operatoren	
&&	und
!	nicht
——	oder
Stringoperatoren	
+	Verknüpfungsoperator
parseInt()	Wandlung in Ganzzahl
parseFloat()	Wandlung Gleitkommazahl
Objekteoperatoren	
.	Punktoperator, Trennt Objekte, Methoden, Eigenschaft

Tabelle 2.1.: Operatoren in JavaScript

mit dem Schlüsselwort „NEW“ aufzurufen und mit der Eigenschaft „LENGTH“ wird die Anzahl der Elemente angegeben.

Die Programmanweisungen werden sequentiell nacheinander in der Reihenfolge ihres Auftretens abgearbeitet. Abweichungen hiervon werden durch Kontrollstrukturen vorgenommen. Diese Kontrollstrukturen sind an Bedingungen geknüpft und stellen Schleifen und Verzweigungen dar. Verzweigungen werden mit den Anweisungen „IF“, „IF ... ELSE“ oder „SWITCH“ ausgeführt. Mit dem Schlüsselwort „IF“ wird eine Bedingung ausgewertet, welche einen boolschen Wert zurückgibt. Liefert diese Bedingung den Wert „TRUE“, erfolgt die Abarbeitung der nachfolgenden Programmanweisung beziehungsweise des in „“ geschweiften Klammern folgenden Anweisungsblocks, während bei „FALSE“ die Ausführung unterbleibt[Pom12].

Mit der Verzweigung „IF ... ELSE“ besteht die Möglichkeit, eine Alternative abzubilden. Entweder wird der auf „IF“ folgende Anweisungsblock ausgeführt oder die dem „ELSE“ folgenden Anweisungen werden abgearbeitet. If-Else-Konstrukte können darüber hinaus auch geschachtelt werden, wobei der „ELSE“ Zweig selbst wieder ein „IF“ nach sich ziehen kann. Bei Mehrfachverzweigungen lässt sich außerdem die Fallunterscheidung mit „SWITCH“ ausführen. Mit „SWITCH“ wird der Wert einer Variablen abgefragt und mit „CASE“ in die Verzweigungen geleitet. Die Switch-Anweisung arbeitet immer nur einen Zweig der Anweisungen ab. Nimmt eine Variable keinen der abgefragten Werte an, dann kann die Ausführung mit „default“ fortgesetzt werden und mit „break“ wird eine Verzweigung beendet[Pom12].

Anweisungsblöcke, die wiederholt ausgeführt werden, sind in Schleifen eingebettet. Dabei wird in Zählschleifen und in Wiederholungen unterschieden. Zählschleifen werden genau n-mal wiederholt und die Laufbedingungen während der Abarbeitung bleiben unverändert. Die Anzahl der Durchläufe hängt bei Wiederholungen von den Laufbedingungen ab, welche sich bei jedem Durchlauf verändern.

Eine Zählschleife wird mit dem Schlüsselwort „FOR“ eingeleitet. In runden Klammern folgen drei Ausdrücke getrennt durch Semikolon. Der erste Ausdruck ist zur Initialisierung der Schleifenvariablen, der zweite Ausdruck ist für die Formulierung der Laufbedingung und der dritte Ausdruck zur Änderung der Schleifenvariablen. Der zu wiederholende Schleifenkörper wird dann in geschweifte Klammern eingeschlossen.

Eine andere Form von Wiederholungen wird durch die While-Schleife repräsentiert. Auf das Schlüsselwort „WHILE“ folgt die Formulierung einer Bedingung in runden Klammern. Liefert die Auswertung des Ausdrucks den Wert „true“, dann wird der Schleifenkörper so lange abgearbeitet, bis die Änderung der Bedingung die Beendigung der Schleife zur Folge hat. Als Alternative lässt sich auch die Do-While-Anweisung nutzen , bei der die Bedingung erst am Ende der Schleife abgefragt wird, wodurch die Schleife mindestens einmal durchgelaufen wird und erst nach dem Durchlauf entschieden wird, ob ein nächster Durchlauf erfolgt.

Ein weiteres Element in JavaScript sind Funktionen, welche in sich abgeschlossene Programme darstellen und Teilaufgaben lösen können. Durch den Einsatz von

Funktionen wird der Programmcode übersichtlicher und redundanzfrei, das heißt wiederkehrende Aufgaben sind nur an einer Stelle im Quelltext vorhanden und können von einer anderen beliebigen Stelle im Programmcode aufgerufen werden[Pom12].

Eine Funktion wird eigenständig mit lokal gültigen Parametern definiert und deren Ausführung erfolgt durch Funktionsaufruf, der Aufruf einer Funktion ist in der Regel an ein Ereignis gebunden. Funktionen können danach aber auch von Funktionen aus aufgerufen werden. Üblicherweise werden die Funktionen im Kopfbereich einer HTML-Datei eingebettet oder sind in einer externen Datei gespeichert.

Das Schlüsselwort für eine Funktion ist „FUNCTION“, hiernach folgt der Name der Funktion mit einer optionalen Parameterliste in runden Klammern und anschließend die Anweisungen der Funktion in geschweiften Klammern. Funktionen tauschen Daten mit dem aufrufenden Programm über die Parameterliste oder einen Rückgabewert aus. Innerhalb eines Funktionsblocks sind mit „VAR“ deklarierte Variable nur lokal in diesem Block gültig. Außerhalb von Funktionen deklarierte Variable gelten als global und sind dann auch innerhalb von Funktionen sichtbar.

Der Aufruf einer Funktion erfolgt durch den Funktionsnamen mit den Argumenten in der Reihenfolge der Parameterliste und die Rückkehr aus einer Funktion erfolgt nach Abarbeitung aller Programmzeilen oder bei Erreichen der Anweisung „RETURN“. Es geht dann mit der auf den Funktionsaufruf folgenden Anweisung im rufenden Programm weiter. Hinter das Schlüsselwort „return“ kann ein Rückgabewert angegeben werden, der im rufenden dann zur weiteren Bearbeitung dem Programm zur Verfügung steht.

2.3.2.3. Objekte in JavaScript

Allgemein sind Objekte oder auch Klassen genannt, zunächst einmal abstrakte Baupläne, nach denen ein konkretes Objekt konstruiert wird. Durch Aufruf des Konstruktors wird ein Objekt instanziiert. Neben den Eigenschaften, auch Attribute genannt, eines Objektes ist auch das Verhalten in Methoden, die auf Objekte angewandt werden, definiert. In der Notation werden die hierarchisch gegliederten Objektebenen durch den Punktoperator getrennt. Der wiederum trennt das Objekt von der Eigenschaft beziehungsweise Methode. Auf einen Methodennamen folgt immer ein Paar runder Klammern und innerhalb der Klammern befinden sich die Methodenparameter[Pom12].

JavaScript selbst erkennt dabei Objekte unterschiedlicher Typen. Dazu zählen integrierte Objekte wie „MATH“, „DATE“ oder „ARRAY“. Weiterhin unterliegen Objekte in JavaScript einer hierarchischen Struktur und es kann über die verfügbaren Methoden auf die Eigenschaften dieser Objekte zugegriffen werden. Dabei ist immer streng der Pfad zu den Objekten, beginnend bei der obersten Ebene, zu beachten. Die oberste Ebene der Objekthierarchie wird durch das Browserfenster selbst repräsentiert und wird mit „WINDOW“ bezeichnet. Unterhalb dieser Ebene befinden sich die Objekte „DOCUMENT“, „LOCATION“, „NAVIGATOR“, „FRAMES“ und „HISTORY“. Des Weiteren ist es möglich, Objekte selbst zu definieren, welche in der Applikation deklariert und instanziiert werden[Pom12].

2.3.2.4. Asynchronous JavaScript and XML

AJAX ist ein Zusammenwirken von HTML, JavaScript, CSS und XML und bietet die Möglichkeit, über ein im Hintergrund laufendes Kommunikationsobjekt, Teile einer Webseite asynchron nachzuladen, ohne dabei die Seite komplett neu laden zu müssen.

Zur Kommunikation zwischen Client und Server mittels AJAX werden die Methoden „GET“ und „POST“ über das Objekt „XMLHttpRequest“ genutzt. Die „GET-Methode“ fordert Daten von einer spezifischen Ressource an. Die „POST-Methode“ kann auch genutzt werden, um Daten anzufordern, zusätzlich bietet sie aber noch die Möglickeit an, Daten zur Verarbeitung an den Server mitzuschicken. Der Server schickt nach der Verarbeitung der Anfrage die Daten mittels HTTP zurück an den Client, dabei werden die Daten in einem Format gesendet, welches einfach von JavaScript zur weiteren Verarbeitung analysiert werden kann[Bri09].

2.3.3. PHP

PHP wurde 1995 von Rasmus Lerdorf als Skriptsprache zur Programmierung von Webanwendungen und zur Erstellung dynamischer Webseiten vorgestellt und zusammen mit Andi Gutmans und Zeev Suraski weiterentwickelt. Die Abkürzung PHP bedeutete ursprünglich „Personal Home Page Tools“, wurde später aber durch „PHP Hypertext „PHP Hypertext Preprocessor“ ersetzt [Sch14].

PHP ist von Perl und C abgeleitet, weshalb auch die Kontrollstrukturen „IF“, „SWITCH“, „FOR“, „WHILE“ und „DO“ identisch mit den entsprechenden Befehlen in der Sprache C sind. Auch die in C zur Verfügung stehenden Operatoren sind nahezu identisch.

Wichtige Unterscheide, die zu nennen sind, ist die Unterscheidung zwischen Groß- und Kleinschreibung. Des Weiteren erfolgt keine Deklaration von Variablen, außer Feldern und Objekten, deshalb ist der Datentyp einer Variable abhängig vom Kontext und nimmt automatisch den am besten geeignetsten Datentyp an. Folglich kann eine Variable dadurch auch während der Laufzeit seinen Datentyp ändern! Zur Sicherheit gibt es den zusätzlichen Operator „===“, welcher bei einem Vergleich von zwei Variablen „var1===var2“ nur dann den Wert „true“ zurückliefert, wenn beide Variablen vom gleichen Datentyp sind und die Inhalte ebenfalls gleich sind. Analoges gilt für den Operator „!==“, bei dem geprüft wird, ob zwei Variablen neben unterschiedlichen Inhalt auch einen unterschiedlichen Datentyp besitzen[Sch14].

Weiterhin ist zu beachten, dass Variablen immer mit einem Dollarzeichen „$“ beginnen, dadurch werden auch innerhalb einer konstanten Zeichenkette die Variablen automatisch erkannt und substituiert, also durch den Inhalt der Variablen ersetzt. Des Weiteren sind Zeichenketten in PHP sehr mächtig und als Konkatenierungsoperator wird der Punkt „.“ verwendet.

PHP kann vollständig in HTML eingebettet werden, weshalb auch innerhalb einer HTML-Datei der PHP-Code jederzeit eingefügt werden kann. Im Kopf besteht dabei

die Möglichkeit, die verwendeten Funktionen zu deklarieren und im Rumpf kann an jeder gewünschten Stelle der Code geschrieben werden. Dies geschieht ,indem der PHP-Code mit dem öffnenden Tag „<?php“ begonnen wird und diesem mit dem schließenden Tag „?>“ abschließt. Wenn in einer HTML-Datei jedoch auch PHP-Code verwendet wird, so muss die Dateiendung auf „.php“ geändert werden, damit der Webserver den PHP-Code auch wirklich ausführen kann.

3. Anforderungen und Konzept des Webtools

Im folgenden Kapitel wird der GSBL und dessen Datenmodell näher erklärt, um zu verstehen auf welchen Datenbestand das Webtool arbeiten soll. Anschließend erfolgt eine Auflistung aller vom Auftraggeber, dem deutschen Umweltbundesamt (UBA), gestellten Anforderungen und dem daraus resultierendem Konzept für das Webtool, um eine ordnungsgemäße Evaluierung zu ermöglichen.

3.1. GSBL

Der gemeinsame zentrale Stoffdatenpool des Bundes und der Länder, kurz GSBL, ist die mittlerweile größte und bedeutendste Chemikalien-Informationsquelle der deutschen Verwaltung und wird vom Bundesumweltministerium (BMUB) gemeinsam mit den Umweltministerien von 14 Bundesländern, nur Bayern und Brandenburg fehlen, betrieben. Das UBA betreut den GSBL als Koordinierungsstelle und betreibt allen verwaltungstechnischen und administrativen Aufwand. Die Betreibung des GSBL beruht auf der Grundlage einer Verwaltungsvereinbarung aus dem Jahr 1994 unter dem Lenkungsausschuss, dessen Vorsitz das BMUB, Referat Z II 3 UI inne hat[BMU].

Der GSBL umfasst aktuelle und verlässliche Informationen über umweltrelevante Eigenschaften von chemischen Stoffen und Zubereitungen, welche von großer Bedeutung für alle Bereiche des Umweltschutzes und zur Gefahrenabwehr sind. Dazu zählen Informationen von ungefähr 63.000 Reinstoffen, 31.000 Zubereitungen und Gemischen sowie 223.000 stoffbezogene und rechtliche Regelungen. Dienststellen des Bundes, der Länder und der Gemeinden nutzen den GSBL zur Abschätzung der möglichen Umweltgefahren eines Stoffes oder zur Risikobewertung im Bereich des Gesundheits- und Verbraucherschutz. So werden zum Beispiel Einsatzkräfte von Feuerwehr und Polizei dabei unterstützt, die Gefährdung von Mensch und Umwelt bei Verkehrs- und Arbeitsplatzunfällen oder auch bei Bränden mit chemischen Substanzen besser beurteilen zu können[BMU].

Ein weiterer Schwerpunkt des GSBL Informationsangebots sind die rechtlichen Regelungen für chemische Stoffe in den unterschiedlichsten Rechtsbereichen: vom Chemikalien-, Abfall-, Wasserrecht über das Transport- und das Lebensmittelrecht bis hin zum Arbeits- und Gesundheitsschutz sowie dem Boden- und Immissionsschutz. Darüber hinaus hat die Öffentlichkeit die Möglichkeit, auf einen Teil des Datenbestands, über den sogenannten „GSBLpublic“ zuzugreifen, welcher im Internet über http://www.gsbl.de/ zu erreichen ist[GSB].

In Abbildung 3.1 sei das Ergebnis der Suche nach „Natriumchlorid“ im „GSBLpublic“ gezeigt. Wie in der Abbildung zu sehen ist, werden neben den Namen auch noch Angaben zum „Registriername“, der „Struktur“, der „Stoffart“ und mehr des gesuchten Stoffes angezeigt.

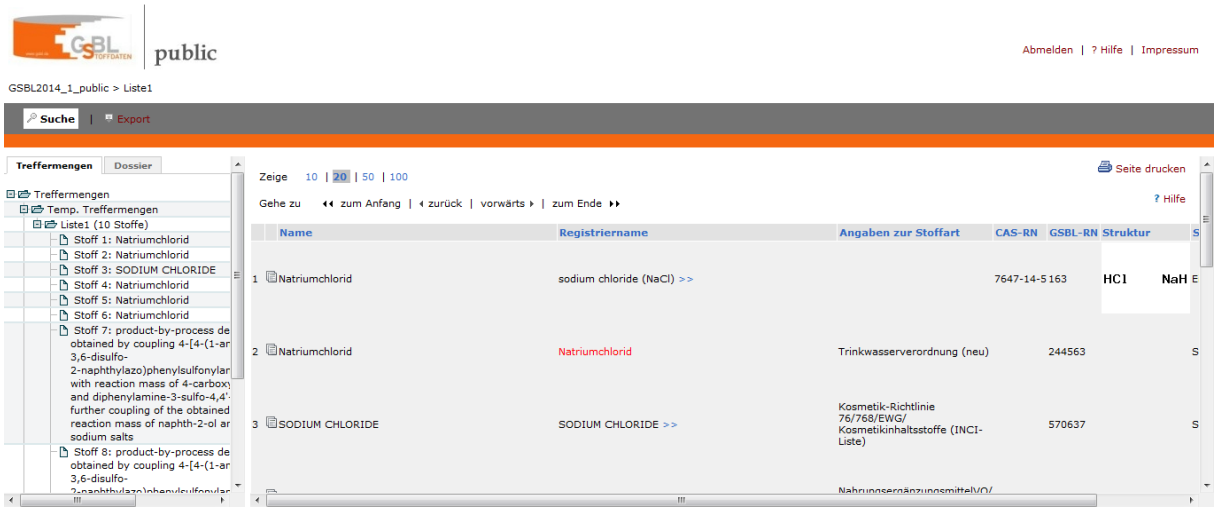


Abbildung 3.1.: Suche im GSBLpublic nach Natriumchlorid(Aufruf am 06.12.2015)

Wie in Abbildung 3.2 dargestellt, erhält der Nutzer beim Auswählen eines der Ergebnisse der Suchanfrage eine detailliertere Auflistung aller dazugehörigen Begriffe des gesuchten Stoffes, wie „IDENTMERKMALE“ oder „RECHTSEIGENSCHAFTEN“.

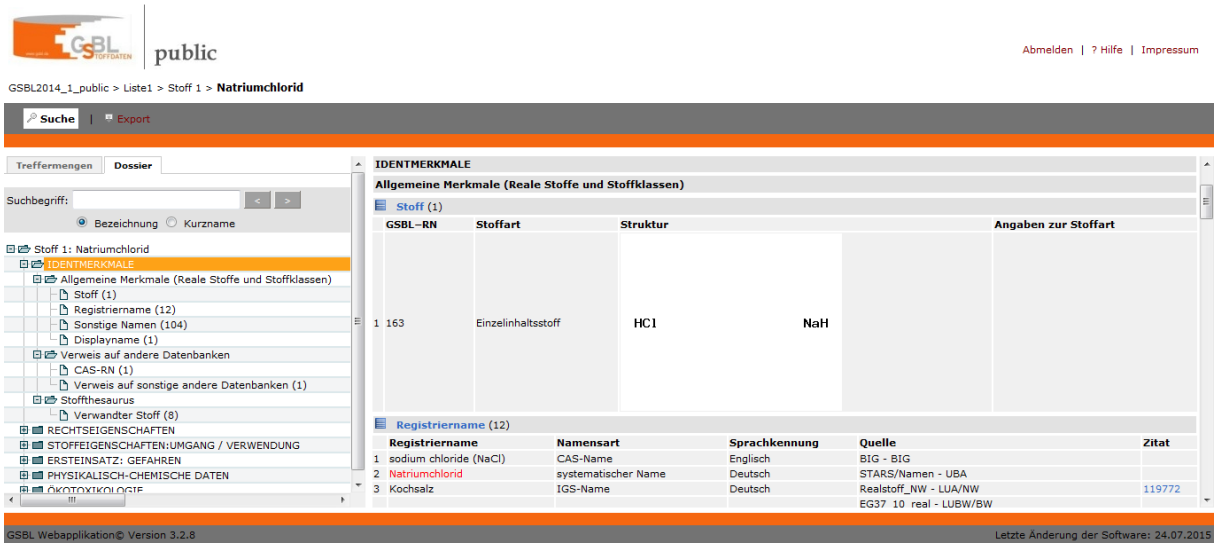


Abbildung 3.2.: Begriffe zu Natriumchlorid im GSBLpublic(Aufruf am 06.12.2015)

Der Begriff „IDENTMERKMALE“ ist in diesem Beispiel ein Oberbegriff für weitere Begriffe, wie „Allgemeine Merkmale (Reale Stoffe und Stoffklassen)“ oder „Stoffthesaurus“. Die Navigation über diese Baumstruktur endet bei den Merkmalen,

wie „Stoff“, „Registriername“ oder „Sonstige Namen“, welche alle Merkmale des Begriffes „Allgemeine Merkmale (Reale Stoffe und Stoffklassen)“ sind. Bei der Auswahl eines dieser Merkmale werden auf der rechten Seite der Webseite des „GSBLpublic“ alle zu diesem Merkmal zugehörigen Felder mit ihren Werten angezeigt. In Abbildung 3.3 sei dies für die Auswahl des Merkmals „Registriername“ gezeigt. Die Felder sind in diesem Beispiel die Spaltenüberschriften, wie „Namensart“, „Sprachkennung“ oder „Quelle“ der Tabelle auf der rechten Seiten.

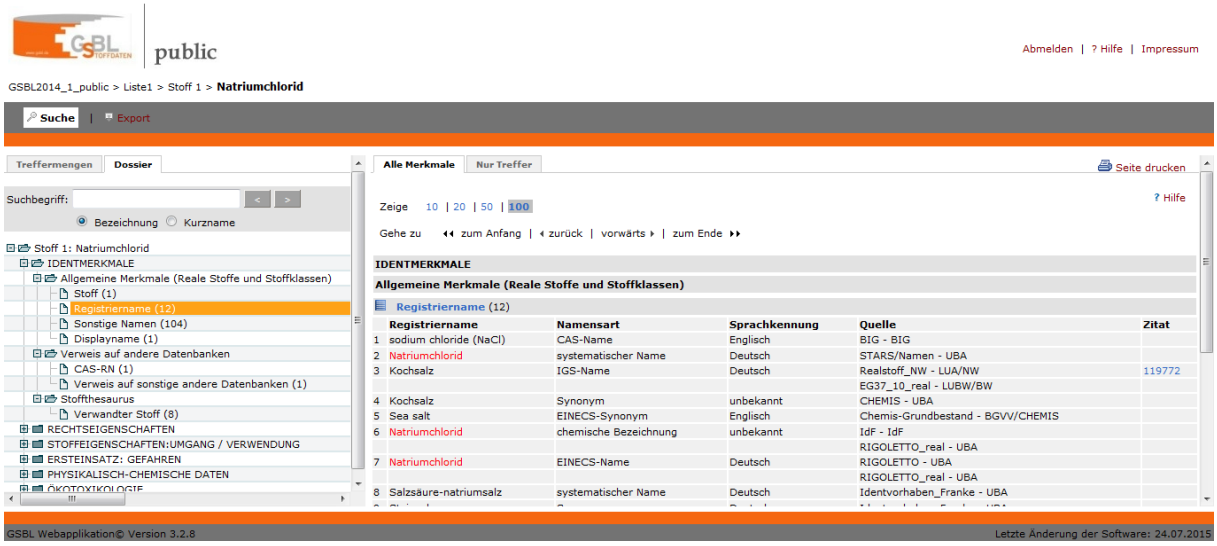


Abbildung 3.3.: Merkmal „Registriername“ im GSBLpublic(Aufruf am 06.12.2015)

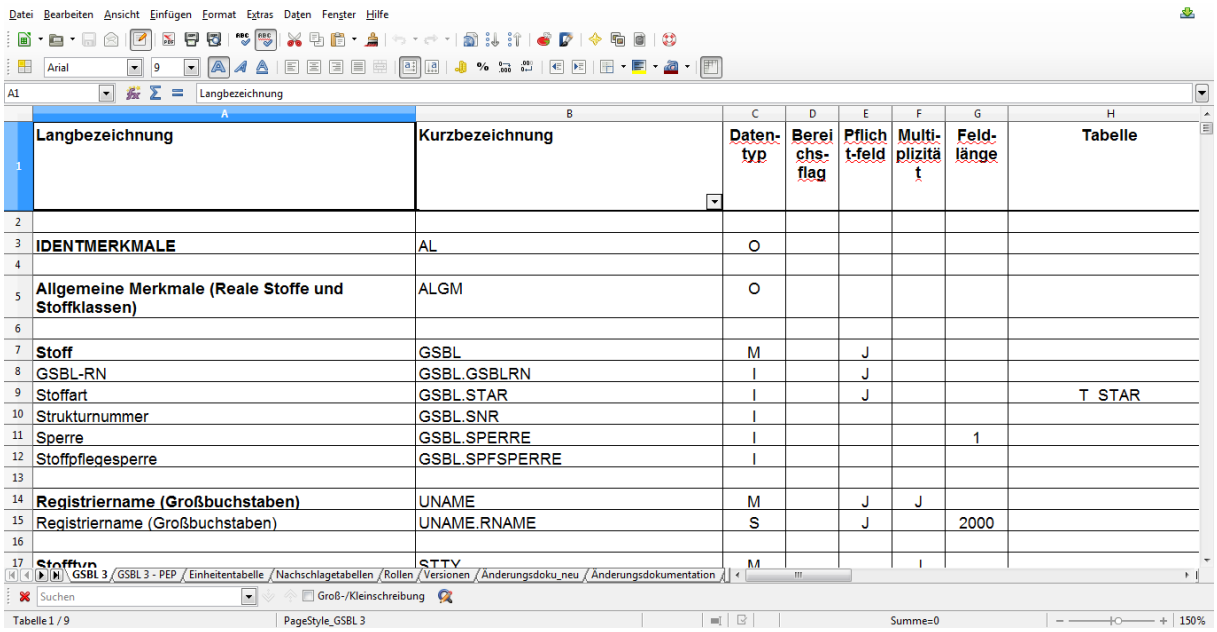
Die Oberbegriffe, Merkmale und Felder im GSBL stellen einen Teil der Begriffshierarchie des Datenmodells dar, somit erfolgt deren ausführliche Beschreibung in Abschnitt 3.2.

3.2. Aufbau des Datenmodells

Zum weiteren Verständnis wird in diesem Abschnitt der Aufbau des Datenmodells dargelegt.

Das Datenmodell des GSBL liegt aktuell, wie in Abbildung 3.4 zu erkennen, nur als Excel-Tabelle vor, in der alle Begriffe mit deren Eigenschaften aufgeführt werden. Jedem Begriff ist eine einmalig vorkommende Lang- und Kurzbezeichnung zugeordnet, welche den Begriff eindeutig identifiziert. Des Weiteren gibt es insgesamt 19 verschiedene Eigenschaften, deren Befüllung von der Art des Begriffes abhängig ist.

Die Begriffe unterteilen sich in drei Arten von Begriffen, dazu zählen die bereits erwähnten Oberbegriffe, Merkmale und Felder. Die Oberbegriffe dienen als Sammelbegriffe für andere Oberbegriffe oder für Merkmale. Im Datenmodell selbst erfolgt keine weitere Differenzierung der Oberbegriffe. Es ist also nicht sofort erkennbar, auf welcher Stufe der Begriffshierarchie der Oberbegriff steht. Deshalb wurden für den weiteren



	Langbezeichnung	Kurzbezeichnung	Datentyp	Bereichsflag	Pflicht-feld	Multiplizität	Feldlänge	Tabelle
2								
3	**IDENTMERKMALE**	AL	O					
4								
5	**Allgemeine Merkmale (Reale Stoffe und Stoffklassen)**	ALGM	O					
6								
7	**Stoff**	GSBL	M		J			
8	GSBL-RN	GSBL.GSBLRN	I		J			
9	Stoffart	GSBL.STAR	I		J			T_STAR
10	Strukturnummer	GSBL.SNR	I					
11	Sperre	GSBL.SPERRE	I				1	
12	Stoffpflegesperre	GSBL.SPFSPERRE	I					
13								
14	**Registriername (Großbuchstaben)**	UNAME	M		J	J		
15	Registriername (Großbuchstaben)	UNAME.RNAME	S		J		2000	
16								
17	**Stofftyp**	STTY	M			I		

Abbildung 3.4.: Auszug der Excel-Datei mit dem Datenmodell

Verlauf der Arbeit die Oberbegriffe in drei weitere Arten unterteilt. Dazu gehören Oberbegriffe der ersten Kategorie(H1), der zweiten Kategorie(H2) und der dritten Kategorie(H3). Im weiteren Verlauf der Arbeit wird die Zugehörigkeit der Oberbegriffe zur Kategorie mit den Bezeichnungen H1, H2 oder H3 abgekürzt. Jedem H1-Oberbegriff können mehrere H2-Oberbegriffe zugeordnet werden und jedem H2-Oberbegriff können mehrere H3-Oberbegriffe zugeordnet werden. Es muss jedoch jeder H2-Oberbegriff genau einem H1-Oberbegriff zugeordnet sein, gleiches gilt für H3-Obergriffe, welche genau einem H2-Oberbegriff zugeordnet sind.

Die zweite Art von Begriffen sind die Merkmale. Sie repräsentieren die Tabellen des GSBL auf Datenbankebene. Jedes Merkmal muss genau einem Oberbegriff zugeordnet sein und einem Oberbegriff können mehrere Merkmale zugeordnet sein. Einem Oberbegriff dürfen jedoch nur Merkmale zugeordnet sein, wenn dieser keinen weiteren H2- oder H3-Oberbegriffe unter sich hat.

Die dritte Art von Begriffen sind die Felder, diese repräsentieren auf Datenbankebene die Spalten einer Tabelle und sind genau einem Merkmal zugeordnet und ein Merkmal muss mindestens ein Feld haben, da eine Tabelle ohne Spalten nicht existieren kann.

In Abbildung 3.5 sei die Begriffshierarchie verdeutlicht dargestellt. Der „XOR“-Knoten, welcher ein exklusives Oder darstellt, dient als Veranschaulichung, dass beispielsweise ein H1-Oberbegriff entweder mindestens einen weiteren H2-Oberbegriff unter sich hat oder mindestens ein Merkmal unter sich hat.

Die bisherige Betreuung des GSBL-Datenmodell erfolgt durch die Fachleute für Chemie aus dem Fachbereich IV (Chemikaliensicherheit) des UBAs. Wie bereits

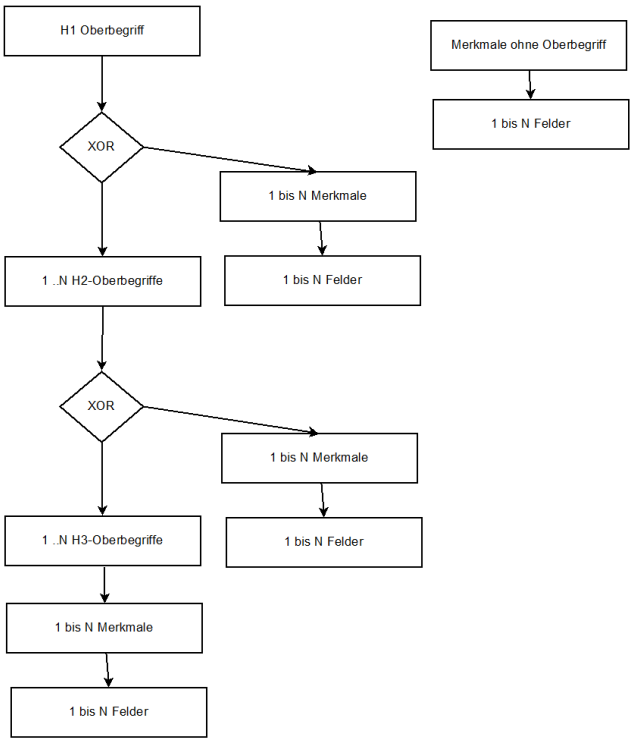


Abbildung 3.5.: Begriffshierarchie

erwähnt, liegt das Datenmodell aktuell nur als Excel-Tabelle vor, welche durch das GSBL-Team im UBA manuell erstellt wurde und gepflegt wird. Diese Art der Pflege ist sehr fehleranfällig, denn es kann leicht vorkommen, dass Pflichtfelder falsch oder gar nicht befüllt sind, Merkmale oder Begriffe falsch zugeordnet sind beziehungsweise sich schlecht zuordnen lassen oder das zu einem H1-Oberbegriff weder H2-Oberbegriffe noch Merkmale zugeordnet sind. Um diese Fehler in Zukunft minimieren zu können, hat sich der Fachbereich IV des UBAs dazu entschlossen die Excel-Tabelle abzuschaffen und das Datenmodell künftig als Relationale Datenbanktabelle zu führen. Um die Pflege der Datenbanktabelle für das GSBL-Team zu erleichtern, sollen Änderungen an der Datenbanktabelle durch eine über das Internet ausführbare Eingabemaske, also einem Webtool, erfolgen. Welche Anforderungen das GSBL-Team an diese Eingabemaske hat, sei daher im nächsten Kapitel ausführlich beschrieben.

3.3. Anforderungen und Konzept

Nachdem im letzten Abschnitt der Aufbau des Datenmodells erklärt wurde, werden die Anforderungen des Auftraggebers an das Webtool zur Bearbeitung des Datenmodells näher erläutert und welchen Pflichten an dessen Funktionen gestellt sind.

Mit Hilfe des Webtools soll es möglich sein, das Datenmodell der GSBL-Datenbank über einen Browser zu bearbeiten, auch wenn keine Fachkenntnisse zu Datenbanken sowie deren Aufbau und Funktionsweisen beim Anwender vorhanden sind. Der Anwender

muss lediglich über das Fachwissen der chemischen Stoffe und deren Beziehungen zueinander verfügen, beziehungsweise sich mit der GSBL-Datenbank auskennen, um mit dem Webtool ordnungsgemäß arbeiten zu können. Deshalb soll die Oberfläche des Webtools einfach gestaltet werden, sodass ein intuitives Arbeiten ermöglicht wird.

Der Auftraggeber wünscht sich, dass das Datenmodell in eine SQL-Datenbanktabelle überführt wird um den Pflegeaufwand des Datenmodells zu verringern. Das Webtool soll als Eingabemaske für Änderungen am Datenmodell dienen, deshalb soll das Webtool die Daten aus der SQL-Datenbanktabelle laden und nach Beendigung der Arbeit wieder erfolgreich dorthin zurück speichern. Über das Webtool soll es möglich sein, theoretisch alle Begriffe sowie deren Eigenschaften bearbeiten zu können. Dazu gehört das Löschen von Begriffen, das Hinzufügen von neuen Begriffen sowie das Verschieben und Bearbeiten von bereits vorhandenen Begriffen.

Um diese Aktionen jedoch korrekt durchführen zu können, ist eine geeignete Navigation im Webtool gesucht. Die Begriffe des GSBL sind hierarchisch miteinander verknüpft, deshalb wünscht sich der Auftraggeber eine ähnliche Navigation, wie sie auch schon im „GSBLpublic“ existiert (siehe Navigationsleiste auf der linken Seite von Abbildung 3.2). In Worten beschrieben, soll über eine klappbare Baumstruktur die Möglichkeit bestehen, Begriffe und deren Eigenschaften auszuwählen. Es soll also möglich sein, einen Oberbegriff auszuwählen, sodass darauf folgend dessen zugeordnete Unterbegriffe zu sehen sind und diese dann wiederum auswählbar zu machen. Des Weiteren ist ein dynamisches Laden während der Bearbeitungen gesucht, sodass der Browser beim Auswählen und Bearbeiten von Begriffen nicht jedes Mal die Seite neu lädt.

Beim Bearbeiten von bereits vorhandenen Begriffen und deren Eigenschaften ist darauf zu achten, dass nur Änderungen vorgenommen werden können, welche auch mit der Definition des Datenschemas konform gehen. Es darf also nicht möglich sein, die Werte von Eigenschaften so zu verändern, dass ein nicht definierter Wert nach Beendigung der Bearbeitung für diese bestimmte Eigenschaft zurückbleibt, womit die Integrität des Datenmodells gefährdet ist.

Der Ablauf des Hinzufügens von neuen Begriffen in das Datenmodell soll so von statten gehen, dass der Anwender zunächst über die Navigation den Oberbegriff auswählt, unter dem der neue Begriff stehen soll und anschließend über die Maske die Werte zu dessen Eigenschaften einträgt. Auch hier ist wieder darauf zu achten, dass nur Werte eingetragen werden, welche auch von der Definition her zulässig sind, um die Integrität nicht zu verletzen.

Das Löschen von Begriffen soll ebenfalls möglich sein. Wenn dieser Schritt vom Anwender getätigt wird, sollen auch alle an dem Begriff dranhängenden Unterbegriffe sowie deren Merkmale und Feldern mit gelöscht werden um weiterhin die Integrität des Datenmodells wahren zu können.

Als Letztes soll es noch möglich sein, bereits vorhandene Begriffe zu verschieben, sodass diese dann als Unterbegriff eines anderen bereits vorhandenen Begriffs aufgeführt werden. Der Auftraggeber verlangt, dass bei dieser Aktion auch alle Unterbegriffe des

zu verschiebenden Begriffs ebenfalls korrekt mit verschoben werden. Die Auswahl der Begriffe soll wieder über die navigierbare Baumstruktur erfolgen.

Der Auftraggeber verlangt zudem, dass nach Abschluss einer Datenbanktransaktion noch zusätzlich in eine Log-Datei geschrieben werden soll, welche Einträge in der SQL-Tabelle geändert wurden und wie der Datenstand dieser Einträge vor der Änderung aussah. Dies ist wichtig damit das GSBL-Team nachvollziehen kann, welche Änderungen gemacht wurden und um diese bei Notwendigkeit wieder rückgängig machen zu können.

Es ist noch zu erwähnen, dass es dem Auftraggeber reicht, wenn nur ein Anwender gleichzeitig mit der ersten prototypischen Umsetzung des Webtools arbeiten kann. Somit ist der Zugriff von mehreren Benutzern und auch die Pflicht parallele Transaktionen zu koordinieren für die Erstellung des Webtools nicht weiter zu beachten. Dies wäre jedoch ein wichtiges Thema für zukünftige Arbeiten und zur Weiterentwicklung des Webtools unablässig.

4. Implementierung

In diesem Abschnitt werden die verschiedenen Möglichkeiten zur Umsetzung des Webtools diskutiert und welche Möglichkeiten genutzt und wie diese umgesetzt wurden.

4.1. Auswahl der Technologien

Zunächst gilt es darzulegen, welche Technologien zur Umsetzung des Webtools genutzt werden sollen.

Bei der Auswahl des Datenbanksystems wurde darauf geachtet, dass dieses frei für die wissenschaftliche Verwendung zur Verfügung steht. Darum wurde sich für MySQL entschieden, denn bei MySQL handelt es sich um ein Open-Source-Software zur Verwaltung von SQL-Datenbanken. MySQL ist zudem auf allen gängigen Betriebssystem wie Unix, Mac OS X, Linux und Windows lauffähig. Darüber hinaus ist es über PHP möglich direkt MySQL-Befehle vom Webtoolserver an die Datenbank zu schicken.[MyS].

Laut Anforderung des Auftraggebers soll das Datenmodell zukünftig als SQL-Tabelle geführt werden und nicht mehr als aufwändig zu pflegende Excel-Tabelle. Dafür muss die bereits bestehende Excel-Tabelle in eine SQL-Tabelle überführt werden. Das direkte Importieren des Inhalts einer Excel-Tabelle in MySQL ist jedoch nicht möglich, weshalb dieses erst in ein anderes Datenformat überführt werden, welches auch von MySQL unterstützt wird. Dafür bieten sich zwei Datenformate an. Das erste Dateiformat ist CSV, welches für „Comma-separated values“ steht. Dieses Datenformat beschreibt den Aufbau einer Textdatei zur Speicherung oder zum Austausch einfach strukturierter Daten, wodurch Tabellen oder eine Liste unterschiedlich langer Listen abgebildet werden. Als zweites könnte „Extensible Markup Language“, abgekürzt XML, genutzt werden. XML ist eine Auszeichnungssprache zur Darstellung hierarchisch strukturierter Daten in Form von Textdateien und wird für den plattform- und implementationsunabhängigen Austausch von Daten zwischen Computersystemen eingesetzt.

Die Umwandlung in eine CSV-Datei ist einfacher zu handhaben, somit ist diese der Umwandlung in eine XML-Datei vorzuziehen. Zur Feststellung der Zugehörigkeit von Oberbegriffen und Merkmalen zu anderen Oberbegriffen sowie die Zuordnung von Feldern zu Merkmalen wurden fünf weitere Spalten hinzugefügt. Die Spalten „H1“, „H2“ und „H3“ beinhalten die Kurzbezeichnungen der Oberbegriffe, unter welche der Begriff geführt wird. Des Weiteren wird für Merkmale die Spalte „Merkmal_Clear“ mit der Bezeichnung des Merkmals, um welches es sich handelt, befüllt. Bei Feldern steht in der Spalte „Merkmal_Clear“ die Bezeichnung des Merkmales, unter welchem das Feld geführt wird und die Spalte „Feld_Clear“ beinhaltet die Bezeichnung des Feldes. Die Kurzbezeichnung eines Merkmales setzt sich dabei aus der Kurzbezeichnung des

zugeordneten Oberbegriffes und dem Inhalt der Spalte „Merkmal_Clear“ zusammen. Gleiches gilt analog zu Felder mit der Spalte „Feld_Clear“. Die vorhandene CSV-Datei lässt sich in eine SQL-Tabelle importieren. Da die Spaltennamen jedoch nicht in die SQL-Tabelle übernommen werden können, ist diese als einzige Anpassung noch mittels des SQL-Befehl „ALTER TABLE CHANGE“ vorzunehmen.

Auf die vollständige SQL-Tabelle lässt sich serverseitig mit den in PHP eingebundenen SQL-Befehlen arbeiten. Clientseitig wird HTML als Grundkonstrukt für das Webtool genutzt, denn HTML wird von jedem Webbrowser unterstützt. Mit HTML lässt sich nur eine statische Seite aufbauen, zur Interaktion von Anwender und Webtool wird deshalb JavaScript und AJAX zur dynamischen Gestaltung verwendet.

4.2. Clientseitige Implementierung des Webtools

Im folgenden Abschnitt wird die clientseitige Implementierung des Webtools gezeigt. Dazu gehört der Aufbau der Webmaske, über welche der Nutzer das Webtool bedient und die Funktionsweise der Bedienungsoptionen.

4.2.1. Aufbau der Webmaske

Bevor mit der Implementierung der Funktionen begonnen werden kann, gilt es zu klären, wie über dem Webtool navigiert werden soll, damit eine intuitive und einfache Bedienung ermöglicht wird.

Die Navigation über eine klappbare Baumstruktur lässt sich über zwei Methoden realisieren. Die erste Mögluchkeit wäre eine Auswahl und Navigation wie sie auch von der Ordner- und Datenstruktur des Explorers bekannt ist. Der Nutzer würde so über diese Struktur bis zum gesuchten Begriff navigieren und diesen dann zur Bearbeitung auswählen. Vorteilhaft ist bei dieser Variante, dass ein computererfahrender Nutzer mit dieser Art der Navigation im Normalfall vertraut ist und somit auch keine Probleme bei der Navigation hätte. Nachteilig dürfte sein, dass die Begriffsübersicht unübersichtlich wird wenn zu viele Oberbegriffe auf einmal aufgeklappt werden. Außerdem ist ein dynamisches Nachladen der Begriffe somit obsolet, denn die gesamte Baumstruktur muss schon beim Laden der Webseite vollständig als HTML-Knoten Struktur vorhanden sein. Eine zweite Methode wäre daher, die Baumstruktur über verkettete Auswahllisten, wie sie auch in vielen Onlineformularen zu finden sind, zu realisieren. Bei dieser Möglichkeit werden Formularlisten zur Auswahl der Begriffe verwendet. Der Benutzer wählt über die erste Liste den gesuchten H1-Oberbegriff, danach wird eine zweite Liste mit den zugehörigen H2-Oberbegriffen dynamisch befüllt und so weiter. Auch hier ist der Vorteil, dass die Navigation intuitiv und einfach erfolgt. Ein kleiner Nachteil wäre, dass die Liste mit den dazugehörigen Unterbegriffen erst dynamisch vom Server angefordert werden muss, weshalb der Nutzer nicht sofort beim Auswählen eines Begriffes darauf zugreifen kann. Ein dynamisches Laden der Begriffe ist jedoch von der Anforderung her erwünscht und somit ist die zweite Navigationsmethode der Ersten vorzuziehen.

Nach der Navigation zum gewünschten Begriff gilt es als nächstes die zum Begriff

gehörigen Eigenschaften anzuzeigen. Dies erfolgt über Formularfelder. Formularfelder haben den Vorteil, dass sie ebenfalls dynamisch befüllt werden können und bei der Bearbeitung der Eigenschafen der Inhalt vom Nutzer geändert werden kann. Um sicherzustellen, dass nur bestimmte Eingaben bei der Bearbeitung der Eigenschaften möglich sind, werden für einige Eigenschaften Auswahllisten mit vorher definierten Auswahlmöglichkeiten genutzt.

Der HTML-Code für die Navigation sieht somit wie folgt aus:

```
<div id="wrapper" style="clear:left;">
<div id="auswahlliste">
<form id="select_form">
H1 w&auml;hlen:<br />
<select id="H1">
<option value="0">Option w&auml;hlen</option>
</select>
<br /><br />
H2 w&auml;hlen:<br />
<select id="H2">
<option value="0">Option w&auml;hlen</option>
</select>
<br /><br />
H3 w&auml;hlen:<br />
<select id="H3">
<option value="0">Option w&auml;hlen</option>
</select>
<br /><br />
<br /><br />
Merkmal w&auml;hlen:<br />
<select id="ME">
<option value="0">Option w&auml;hlen</option>
</select>
<br /><br />
Feld w&auml;hlen:<br />
<select id="FE">
<option value="0">Option w&auml;hlen</option>
</select>
<br /><br />
</form>
</div>
```

Zur besseren Übersicht wurde die Navigationsauswahl der Begriffe und Eigenschaftsübersicht des Begriffes voneinander getrennt und nebeneinander gesetzt. Der Aufbau der Eigenschaftsübersicht zu den Begriffen sei im folgenden Programmcodeauszug gezeigt.

```
<div id="eingabefelder">
<form>
<table border="0" cellpadding="0" cellspacing="4">
```

```
<tr>
<td align="right">LANGBEZ:</td>
<td><input id="LANGBEZ" name="LANGBEZ" type="text"
 size="60" maxlength="60" ></td>
</tr>
<tr>
<td align="right">KURZBEZ:</td>
<td><input id="KURZBEZ" name="KURZBEZ" type="text"
 size="60" maxlength="60" ></td>
</tr>
<tr>
<td align="right">MERKMAL_CLEAR:</td>
<td><input id="MERKMAL_CLEAR" name="MERKMAL_CLEAR" type="text"
 size="60" maxlength="60" ></td>
</tr>
<tr>
<td align="right">FELD_CLEAR:</td>
<td><input id="FELD_CLEAR" name="FELD_CLEAR" type="text"
 size="60" maxlength="60" ></td>
</tr>
<tr>
<td align="right">Datentyp:</td>
<td>
<select id="DATENTYP" name="DATENTYP">
<option value=""></option>
</select>
</td>
</tr>
...
</form>
</div>
</div>
```

Um Änderungen durchführen zu können, wurden zusätzlich noch einige Buttons, welche bestimmte Aktionen auslösen, zur Maske hinzugefügt. Dazu gehört die Freigabe zur Bearbeitung der Eigenschaften, das Zurücksetzen der Inhalte von bearbeiteten Eigenschaften, das Speichern der Änderungen und das Löschen eines ausgewählten Begriffes. Die Optionen zum Verschieben und Hinzufügen von Einträgen wurden auf zwei andere Seiten ausgelagert. Es hat sich während der Programmierung gezeigt, dass es umfangreicher ist alle Funktionen auf eine Seite unterzubringen, womit die Maske des Webtools wie in Abbildung 4.1 aussieht.

Die Seite zum Hinzufügen neuer Einträge hat den gleichen Aufbau wie die eigentliche Hauptseite, wie in Abbildung 4.2 zu sehen ist. Unterschiedlich sind nur die Auswahlmöglichkeiten bei den verketteten Auswahllisten. Neu hinzu kommt die Auswahlmöglichkeit „Neuen Eintrag als ... hinzufügen“.

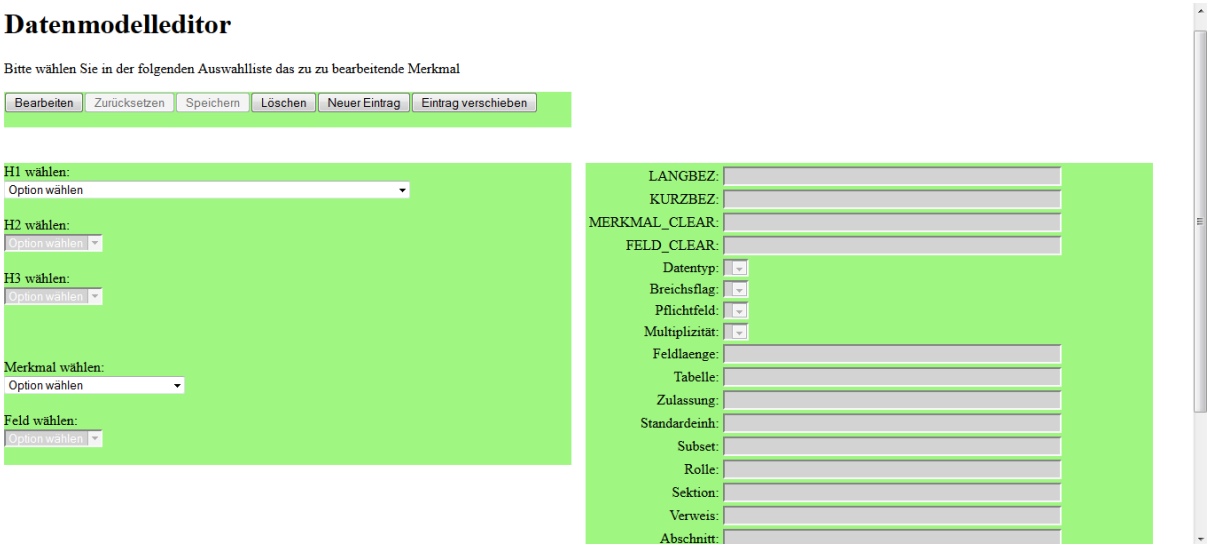


Abbildung 4.1.: Aufbau der Webtoolmaske

Bei der Seite für das Verschieben von Einträgen wurde die Übersicht zu den Eigenschaften durch eine weitere Navigationsliste zur Auswahl eines Begriffes ersetzt, wie in Abbildung 4.3 zu sehen ist. Während auf der linken Seite der Begriff ausgewählt wird, den der Nutzer verschieben möchte, so wird auf der rechten Seite der Begriff ausgewählt worunter der zu verschiebende Begriff künftig geführt werden soll. Um schon clientseitig gewährleisten zu können, dass die Verschiebung eines Begriffes die Integrität nicht verletzt, muss der Nutzer erst den zu verschiebenden Begriff auswählen. Erst danach hat er die Möglichkeit zu sehen, worunter er den ausgewählten Begriff verschieben kann, da die Auswahlmöglichkeiten darauf aufbauend beschränkt werden.

4.2.2. Programmierung der Funktionen

Nachdem im vorherigen Abschnitt nur der Aufbau der Webmaske dargestellt wurde, wird im folgenden Abschnitt gezeigt, wie die Funktionen zur Interaktion mit der Datenbank programmiert wurden sind. Zu erwähnen ist, dass jegliche Änderung an der Datenbank ein Neuladen des Webtools auslöst, um die Integrität der angezeigten Daten über das Webtool zu erhalten, womit das Webtool nicht zu hundertprozentig dynamisch ist.

4.2.2.1. Funktionsweise der Hauptseite

Um Überhaupt die Funktionen des Webtools ermöglichen zu können muss Javascript geladen werden, gleichzeitig werden einige Variablen, welche das Arbeiten mit dem Client ermöglichen definiert, wie im folgenden Codeauszug dargestellt ist.

```
<script type="text/javascript" src="jquery-1.11.2.min.js"></script>
<script type="text/javascript">

/*
* lastSelectedEntry gibt an welcher Begriff zuletzt selektiert wurde
* wenn lastSelectedEntry = 0 ist
```

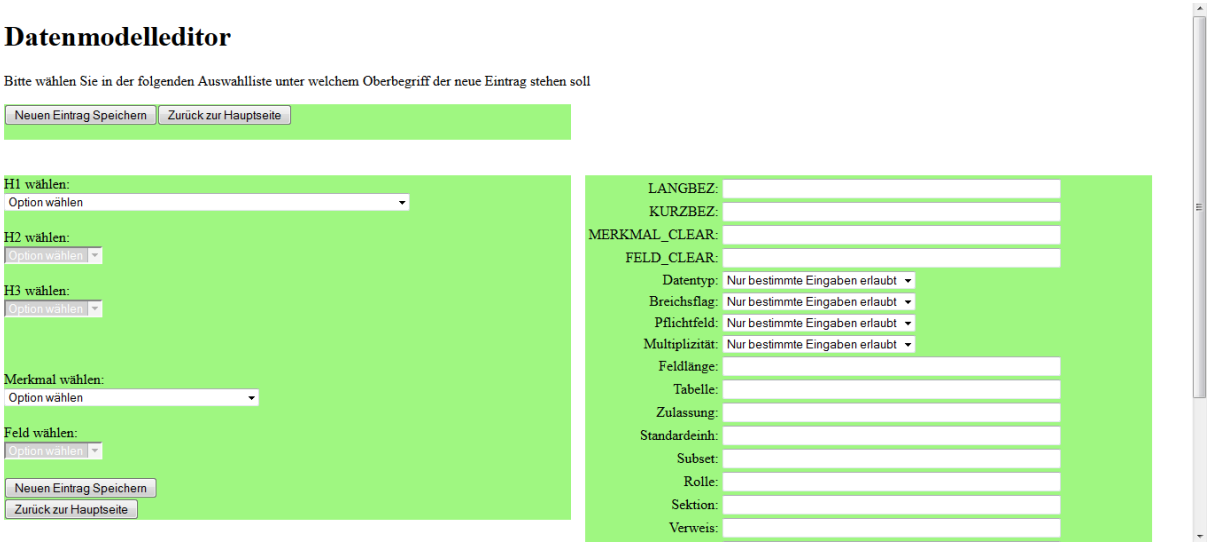


Abbildung 4.2.: Seite für das Hinzufügen eines neuen Eintrages

```
* dann ist kein Begriff zuletzt selektiert
*/
var lastSelectedEntry = 0;

/*
* lastSelectedO gibt an welcher Oberbegriff zuletzt aktuell war,
* bevor ein Merkmal selektiert wurde
* wenn lastSelectedO = 0
* dann ist kein Oberbegriff selektiert
*/
var lastSelectedO = 0;

/*
* selectedEntryTyp gibt an von welchem Typ
* der ausgew"ahlte Eintrag ist
* "O" = Oberbegriff
* "M" = Merkmal
* "F" = Feld
* ""  = kein Eintrag wurde zuletzt ausgew"ahlt
*/
var lastSelectedEntryTyp = "";

/*
* Zwischenvariablen f"ur die Auswahllisten der Attribute
*/

var lastSelectedDATENTYP = "";
var lastSelectedBREICHSFLAG = "";
var lastSelectedPFLICHTFELD = "";
```

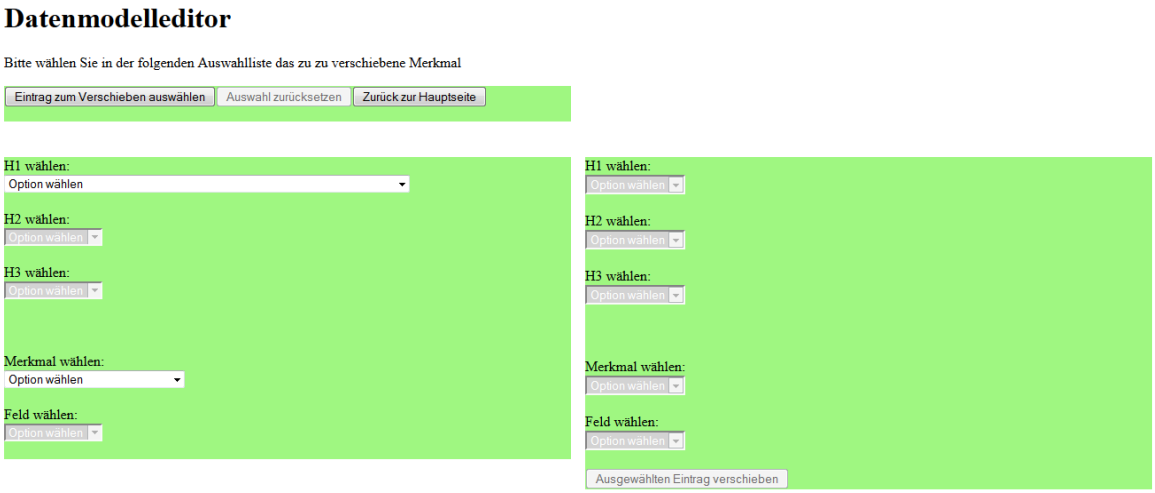


Abbildung 4.3.: Seite für das Verschieben eines Eintrages

```
var lastSelectedMULTIPLI = "";
```

Anschließend erfolgt die Initialisierung der Webmaske. Hierbei wird bestimmt, welche Auswahllisten und Felder anfangs als Auswahl zur Verfügung stehen. Dafür ist auch das Laden der H1-Oberbegriffe sowie der Merkmale ohne Oberbegriffe in den entsprechenden Auswahllisten notwendig. Dafür müssen diese vom Server angefordert werden. AJAX bietet für die Kommunikation mit dem Server die Methoden „GET“ und „POST“ an. Die „GET“-Methode ist schneller und einfacher und kann in den meisten Fällen genutzt werden. Die „POST“-Methode hat jedoch einige Vorteile. Dazu gehört, dass die Menge an Daten zum Versenden nicht begrenzt ist und dass sie sicherer und robuster als die „GET“-Methode gilt, wenn es um das Senden von Nutzereingaben geht, falls diese Sonderzeichen enthalten. Speziell, wenn es um das Anfordern und Ändern von Datenbankinhalten geht, bei denen fast immer die Eingaben vom Benutzer berücksichtigt werden, ist die „POST“-Methode immer vorzuziehen. Aus diesem Grund wird durchgehend die Methode „$.post“ zur Kommunikation mit dem Server verwendet.

```
$("select#H1").html("<option>Begriffe werden geladen</option>");
$.post("select_H1.php", {}, function(data){
$("select#H1").removeAttr("disabled");
$("select#H1").html(data);
});
$("select#ME").html("<option>Begriffe werden geladen</option>");
$.post("select_ME.php", {}, function(data){
$("select#ME").removeAttr("disabled");
$("select#ME").html(data);
});
$("select#H2").attr("disabled","disabled");
$("select#H3").attr("disabled","disabled");
$("select#ME").attr("disabled","disabled");
$("select#FE").attr("disabled","disabled");
```

```
$("select#H1move").attr("disabled","disabled");
$("select#H2move").attr("disabled","disabled");
$("select#H3move").attr("disabled","disabled");
$("select#MEmove").attr("disabled","disabled");
$("select#FEmove").attr("disabled","disabled");

empty_Attribute();
disable_Attribute();
document.getElementById("deletebutton").disabled = false;
document.getElementById("backbutton").disabled = true;
document.getElementById("savebutton").disabled = true;
```

Im zu sehenden Codeauszug wird nach „$.post“ beispielsweise „select_H1.php“ aufgerufen. Über diese PHP-Seite erfolgt das Anfordern der Daten von der SQL-Datenbank. Diese Anforderung wird vom Server bearbeitet, weshalb eine Erklärung zur Funktionsweise im entsprechenden Abschnitt erfolgt.

Um zu erkennen, wann immer der Benutzer eine Auswahl in der Auswahlliste tätigt, ist eine Funktion notwendig. AJAX bietet hierfür die Metohde „change“ an. Mit dieser Methode kann eine Funktion gestartet werden, welche den Inhalt der Webmaske dynamisch verändert. Wenn der Nutzer also beispielsweise einen Begriff aus der Auswahlliste der H1-Oberbegriffe wählt, so müssen alle dazugehörigen Merkmale oder H2-Oberbegriffe in den jeweiligen Auswahllisten nachgeladen werden, wie es im folgenden gekürzten Codeauszug zu sehen.

```
$("select#H1").change(function(){
$("select#H3").empty();
...
document.getElementById("savebutton").disabled = true;

/**
* @var string id
* Ist der dazugeh"orige Valuewert(KURZBEZ) zum Labelwert(LANGBEZ)
*/

var id = $("select#H1 option:selected").attr('value');
if(id==0){
$("select#H2").empty();
$("select#H2").html('<option value="0">Option w&auml;hlen</option>');
$("select#ME").html("<option>Begriffe werden geladen</option>");
$.post("select_ME.php", {}, function(data){
$("select#ME").removeAttr("disabled");
$("select#ME").html(data);
});
lastSelectedEntry = 0;
lastSelected0 = 0;
lastSelectedEntryTyp = "";
}
```

```
else{
  $("select#H2").html("<option>Begriffe werden geladen</option>");
        $.post("select_H2.php", {id:id}, function(data){
$("select#H2").removeAttr("disabled");
$("select#H2").html(data);
});

$("select#ME").html("<option>Begriffe werden geladen</option>");
$.post("select_ME_from_H1.php", {id:id}, function(data){
$("select#ME").removeAttr("disabled");
$("select#ME").html(data);
});
lastSelectedEntry = id;
lastSelectedO = id;
lastSelectedEntryTyp = "O";
get_Attribute(id);
}
});
```

Hierbei werden auch gleichzeitig über die Funktion „get_Attribute(id)“ die zum gewähltem Begriff dazugehörigen Attribute mittels der Methode „$.post“ vom Server angefordert. Die vom Server zurückgegebenen Daten „data“ werden anschließend in die Formularfelder, wie „input#LANGBEZ“, eingetragen. Bei den Auswahllisten wie „select#DATENTYP“ werden zusätzlich noch die Auswahlmöglichkeiten für den Benutzer festgelegt, wie im folgenden Codeauszug zu sehen ist.

```
function get_Attribute(id){
$.post("get_Attribute.php", {id:id}, function(data){
data = JSON.parse(data);
$("input#LANGBEZ").val(data[0]);
$("input#KURZBEZ").val(data[1]);
$("input#MERKMAL_CLEAR").val(data[2]);
$("input#FELD_CLEAR").val(data[3]);
lastSelectedDATENTYP = data[4];
...
$("input#INKRAFT").val(data[22]);
...
```

Abhähngig davon, ob der ausgewählte Begriff ein Oberbegriff, Merkmal, oder Feld ist, erfolgt eine Anpassung einiger Auswahllisten auf der Seite der Eigenschaftsübersicht.

```
...
if(lastSelectedEntryTyp == "O"){
$("select#DATENTYP").html('<option value="O">O</option>');
}
if(lastSelectedEntryTyp == "M"){
$("select#DATENTYP").html('<option value="M">M</option>');
$("select#PFLICHTFELD").html('<option value="J">J</option>
```

```
<option value="N">N</option>');
for(var i, j = 0; i = PFLICHTFELD.options[j]; j++) {
     if(i.value == lastSelectedPFLICHTFELD) {
       PFLICHTFELD.selectedIndex = j;
         break;
     }
}
$("select#MULTIPLI").html('<option value="J">J</option>
<option value="N">N</option>');
for(var i, j = 0; i = MULTIPLI.options[j]; j++) {
     if(i.value == lastSelectedMULTIPLI) {
         MULTIPLI.selectedIndex = j;
         break;
     }
}
}
if(lastSelectedEntryTyp == "F"){
...
}
});
}
```

Nach der Auswahl des Begriffes hat der Nutzer die Möglichkeit diesen zu löschen. Dies geschieht, indem er auf den Button „Löschen“ klickt. Dieser Vorgang lässt sich für den Client mit der Methode „submit“ nachvollziehen, wie im folgenden Codeauszug dargestellt wird.

```
$("form#delete").submit(function(){
if(lastSelectedEntry == 0 || lastSelectedEntry ==""){
     alert("Fehler! Es wurde kein Eintrag zum L"oschen ausgew"ahlt!");
    }
    else{
delete_Entry(lastSelectedEntry);
}
return false;
});
```

Hier erfolgt vom Client her die Prüfung, ob auch wirklich ein Eintrag zum Löschen ausgewählt wurde und falls dem so ist, wird über „alert“ eine Fehlermeldung ausgegeben. Ansonsten wird die Funktion „delete_Entry“ mit dem zu löschenden Begriff aufgerufen. Bevor die Aufforderung jedoch an den Server geschickt wird, erfolgt eine Sicherheitsabfrage über „confirm“, ob der ausgewählte Begriff auch tatsächlich gelöscht werden soll, wie im nachfolgenden Codeauszug zu sehen ist.

```
function delete_Entry(id){
var agree = confirm("Wollen Sie diesen Eintrag, sowie
alle dazugehoerigen Unterbegriffe, Merkmale und Felder
wirklich loeschen?");
```

```
if (agree){
$.post("delete_Entry.php", {id:id}, function(data){
if(data > 0){
alert("Der Eintrag wurde erfolgreich geloescht!");
window.location.reload();
}
if(data == -1){
alert("Fehler! Der Eintrag wurde nicht geloescht!
Wiederholen Sie bitte den Vorgang!");
}
if(data < -1 || data == 0){
alert("Fehler!");
}

});
}
}
```

Zur Bearbeitung der Eigenschaften eines ausgewählten Begriffs muss der Button „Bearbeiten“ ausgewählt werden. Dadurch werden die Formularfelder über die Methode „removeAttr“ für den Benutzer zum Bearbeiten freigegeben, wie als Bespiel für das Formularfeld „input#LANGBEZ“ ersichtlich ist.

```
$("input#LANGBEZ").removeAttr("disabled");
```

Um die Änderungen auf der Datenbank zu speichern, wird der Button „Speichern“ benutzt. Dieser ruft die Funktion „save_Attribute(id)“ auf, welche den Inhalt der Formularfelder ausliest und diesen zur Verarbeitung an den Server schickt. Nach der Verarbeitung der Daten auf dem Server schickt dieser einen Statuscode zurück an den Client, damit dieser eine Rückmeldung in Form einer Nachricht an den Nutzer übermitteln kann. Welche Arten es an Rückmeldungen gibt, lässt sich im folgenden Codeauszug nachvollziehen.

```
function save_Attribute(id){
var lang = document.getElementsByName("LANGBEZ")[0].value;
var kurz = document.getElementsByName("KURZBEZ")[0].value;
var idME = document.getElementsByName("MERKMAL_CLEAR")[0].value;
var idFE = document.getElementsByName("FELD_CLEAR")[0].value;
var dt = $("select#DATENTYP option:selected").attr('value');
...
var inkraft = document.getElementsByName("INKRAFT")[0].value;

$.post("set_Attributes.php", {id:id,lang:lang,kurz:kurz,idME:idME
,idFE:idFE,dt:dt,bf:bf,pf:pf,mp:mp,fl:fl,ta:ta,zu:zu,se:se,ss:ss,
ro:ro,sek:sek,ve:ve,ab:ab,an:an,zt:zt,it:it,fs:fs,be:be
,inkraft:inkraft},
function(data){
        switch(data){
case "1":
```

```
alert("Der Eintrag wurde erfolgreich geaendert!");
window.location.reload();
break;
case "0":
alert("Fehler! Die eingegebene KURZBEZ wird bereits verwendet!
Tragen Sie bitte eine Andere ein!");
break;
case "-1":
alert("Fehler! Der Eintrag wurde nicht hinzugefuegt!
Wiederholen Sie bitte den Vorgang!");
break;
case "-2":
alert("Fehler! LANGBEZ und KURZBEZ duerfen nicht leer oder 0 sein!");
break;
case "-3":
alert("Fehler! Es wurde ein nicht zulaessiger Datentyp uebertragen!");
break;
case "-4":
alert("Fehler! Bei einem Oberbegriff muss Merkmal_clear
und Feld_clear leer sein!");
break;
case "-5":
alert("Fehler! Bei einem Merkaml darf Merkmal_clear
nicht leer sein und Feld_clear muss leer sein!");
break;
case "-6":
alert("Fehler! Bei einem Feld darf Merkmal_clear
und Feld_clear nicht leer sein!");
break;
case "-7":
alert("Fehler! Bei einem Feld mit Datentyp W muss
Breichsflag gesetzt sein!");
break;
case "-8":
alert("Fehler! Bei einem Merkmal oder Feld muss
Pflichtfeld gesetzt sein!");
break;
case "-9":
alert("Fehler! Bei einem Merkmal oder Feld muss
Multiplizit"at gesetzt sein!");
break;
case "-10":
alert("Fehler! Bei einem Feld mit Datentyp S,
darf FELDLAENGE nicht leer sein und es duerfen nur
natuerliche Zahlen groesser 0 eingetragen werden!");
break;
case "-11":
```

```
alert("Fehler! Die KURZBEZ dieses Feldes ist fehlerhaft!");
break;
case "-12":
alert("Fehler! Bei einem Merkmal darf MERKMAL_CLEAR
nicht leer oder 0 sein!!");
break;
case "-13":
alert("Fehler! Bei einem Feld darf Feld_CLEAR
nicht leer oder 0 sein!!");
break;
default:
alert("Fehler!");
break;
}
});
}
```

4.2.2.2. Erstellen eines neuen Eintrags

Um auf die Seite zum Erstellen eines neuen Eintrags zu gelangen muss auf den Hauptseite der Button „Neuer Eintrag“ geklickt werden. Die Seite unterscheidet sich vom Aufbau nur wenig von der Hauptseite. Unterschiedlich ist, dass es nur zwei Buttons gibt, einer um zur Hauptseite zurückzukehren und einen um den Neuen Eintrag zu speichern.

Die Auswahl, worunter der neue Eintrag angelegt werden soll, erfolgt wie bei der Hauptseite über die verketteten Auswahllisten. Wenn der Nutzer die Stelle gefunden hat, worunter er den neuen Eintrag anlegen möchte, so muss er in der entsprechenden Auswahlliste die Option „Neuer Eintrag als...“ auswählen.

Je nachdem, ob es sich beim neuen Eintrag um einen Oberbegriff, einem Merkmal oder um ein Feld handelt, werden die Formularfelder und Auswahllisten für die Eigenschaften vom Client entsprechend vorbereitet. Als Beispiel sei die Funktion „AttributeO()“ angegeben.

```
/*
* AttributeO bestimmt welche Formularfelder bei einem Oberbegriff
* ausw"ahlbar sind
*/

function AttributeO(){
$("input#KURZBEZ").removeAttr("disabled");
document.getElementsByName("KURZBEZ")[0].value ='';

$("input#MERKMAL_CLEAR").attr("disabled","disabled");
document.getElementsByName("MERKMAL_CLEAR")[0].value ='';
```

```
$("input#FELD_CLEAR").attr("disabled","disabled");
document.getElementsByName("FELD_CLEAR")[0].value ='';

$("select#DATENTYP").empty();
$("select#DATENTYP").html('<option value="O">O</option>');

$("select#BREICHSFLAG").empty();
$("select#BREICHSFLAG").html('<option value="">
Nur bei Feldern mit Datentyp W ausw&auml;hlbar</option>');
$("select#BREICHSFLAG").attr("disabled","disabled");

$("select#PFLICHTFELD").empty();
$("select#PFLICHTFELD").html('<option value="">
Nur bei Feldern und Merkmalen ausw&auml;hlbar</option>');
$("select#PFLICHTFELD").attr("disabled","disabled");

$("select#MULTIPLI").empty();
$("select#MULTIPLI").html('<option value="">
Nur bei Feldern und Merkmalen ausw&auml;hlbar</option>');
$("select#MULTIPLI").attr("disabled","disabled");

$("input#FELDLAENGE").attr("disabled","disabled");
document.getElementsByName("FELDLAENGE")[0].value ='';

document.getElementsByName("SEKTION")[0].value ='MERKMALE';
}
```

Diese Funktion bestimmt beispielsweise, dass nur der Datentyp „O“, welcher für Oberbegriff steht, ausgewählt werden kann und das die Auswahlliste „Bereichsflag“ nicht zur Bearbeitung frei steht.

Der Button „Neuen Eintrag speichern“ wird angeklickt, um den neuen Eintrag speichern zu können. Anschließend wird die Funktion „$(„form#save“).submit(function()“ ausgeführt, welche prüft, unter welchen anderen Begriffen der neue Eintrag eventuell steht. Der nachfolgende Programmcodeauszug ist aufgrund seiner repetitiven Form gekürzt.

```
/*
* Vor dem Speichern wird gepr"uft unter welchen Oberbegriffen
*(H1-H3) und Merkmal der neue Eintrag evtl. steht
*/
$("form#save").submit(function(){

var checkH1 = $("select#H1 option:selected").attr('value');
...
var idME;
if(checkH1 == 0 || checkH1 == -1){
idH1 ="";
}
```

```
else{
idH1 = $("select#H1 option:selected").attr('value');
}
...
if(checkME == -1){
idME ="";
}
else{
if(checkME == 0){
idME = document.getElementsByName("MERKMAL_CLEAR")[0].value;
}
else{
idME = $("select#ME option:selected").attr('value');
}

}

if(checkH1 != 0 && checkH2 != 0 && checkH3 != 0 && checkME != 0
&& checkFE != 0){
alert("Fehler! Es wurde keine Option selektiert,
bei der ein neuer Eintrag eingetragen wird!");
}
else{
var id = document.getElementsByName("KURZBEZ")[0].value;
$.post("get_KURZBEZ.php", {id:id}, function(data){
idCheck = data;
save_New(idH1,idH2,idH3,idME,checkFE);
  });
}
return false;
});
```

Anschließend erfolgt der Aufruf der Funktion „save_New“, welche eine Auswertung der Formularfelder und Auswahllisten der Eigenschaften übernimmt und diese an den Server zur weiteren Verarbeitung schickt und eine Rückmeldung über die Verarbeitung an den Nutzer weiter gibt. Die Funktion ist nahezu identisch vom Aufbau her mit der Funktion „save_Attribute(id)“ von der Hauptseite, weshalb auf den Programmcodeauszug verzichtet wird.

4.2.2.3. Verschieben eines Eintrags

Um auf die Seite zum Verschieben eines Eintrags zu gelangen, muss auf der Hauptseite der Button „Eintrag verschieben“ geklickt werden. Bei dieser Seite entfällt die rechte Hälfte, wo vorher die Formularfelder und Auswahllisten für die Eigenschaften standen, durch eine zweite verkettete Auswahlliste.

Die Auswahl welcher Eintrag verschoben werden soll, erfolgt wie bei der Haupt-

seite über die linken verketteten Auswahllisten. Wenn der Nutzer den Eintrag gefunden hat, den er verschieben möchte, so klickt er auf den Button „Eintrag zum Verschieben auswählen“. Bei der Auswahl des Eintrags sind jedoch noch verschiedene Prüfungen vorzunehmen. So muss ermittelt werden, ob es sich beim zu verschiebenden Eintrag um einen Oberbegriff, ein Merkmal oder um ein Feld handelt. Bei einem Oberbegriff muss zudem die maximale Tiefe möglicher Unterbegriffe ermittelt werden. Dies geschieht beispielsweise für einen H1-Oberbegriff über die Anfrage „get_DepthH1.php“ an den Server, wie im nachfolgenden gekürzten Codeauszug dargestellt wird. Dies ist wichtig, damit die Auswahlmöglichkeiten, worunter der gewählte Begriff verschoben werden kann, beschränkt werden.

```
$("form#selectentry").submit(function(){
var checkH1 = $("select#H1 option:selected").attr('value');
...
var checkFE = $("select#FE option:selected").attr('value');

/*
* depth gibt die maximale Tiefe der m"oglichen Unterbegriffe
* des ausgew"ahlten Eintrags an
* depth = 0 => Eintrag hat keine Unterbegriffe
* depth = 1 => Eintrag besitzt Kindknoten
* depth = 2 => Eintrag besitzt Enkelknoten
* Merkmale und Felder werden zur Ermittlung der Tiefe nicht
* ber"ucksichtigt
*/

depth = -1;
selectedEntryTyp ="";
if(checkH1==0 && checkME==0){
alert("Fehler! Es wurde kein Eintrag selektiert!");
}
else{
if(checkH2 == 0 && checkME==0){
selectedEntry = checkH1;
selectedEntryTyp ="0";
$.post("get_DepthH1.php", {id:selectedEntry}, function(data){
depth = data;
});
}
else{
...
}
if(depth<=1){
$("select#H1").attr("disabled","disabled");
...
            $("select#FE").attr("disabled","disabled");
            $("select#H1move").html("<option>
```

```
            Begriffe werden geladen</option>");
            $.post("select_H1move.php", {}, function(data){
$("select#H1move").removeAttr("disabled");
                $("select#H1move").html(data);
            });
            if(selectedEntryTyp !="0"){
                 $("select#MEmove").html("<option>
                 Begriffe werden geladen</option>");
                 $.post("select_MEmove.php", {}, function(data){
  $("select#MEmove").removeAttr("disabled");
     $("select#MEmove").html(data);
                });
            }
            else{
$("select#MEmove").empty();
$("select#MEmove").attr("disabled","disabled");
$("select#MEmove").html('<option value="-1">
Ein Oberbegriff l&auml;sst sich nicht als Merkmal verschieben
</option>');
$("select#FEmove").empty();
$("select#FEmove").attr("disabled","disabled");
$("select#FEmove").html('<option value="-1">
Ein Oberbegriff l&auml;sst sich nicht als Feld verschieben
</option>');
}
document.getElementById("movebutton").disabled = false;
document.getElementById("selectentrybutton").disabled = true;
document.getElementById("selectresetbutton").disabled = false;
}
else{
alert("Ausgew&auml;hlter Eintrag kann nicht verschoben werden
da die Tiefe der Unterbegriffe zu gross ist! Entfernen oder
Verschieben Sie erst die tiefsten Unterbegriffe!");
/*
* Diese Meldung erscheint nur bei Oberbegriffen,
* welche ein H1 sind und mindestens ein H3 besitzten
*/
}
}
return false;
});
```

Danach kann der Nutzer in der verketteten Auswahlliste, welche sich auf der rechten Seite der Webseite befindet, auswählen worunter er den neuen Begriff verschieben kann. Die Auswahlmöglichkeiten sind jedoch je nach gewähltem Begriff beschränkt. So kann ein Oberbegriff nur verschoben werden, solange die maximale Tiefe aller Oberbegriffe nicht überschritten wird, weil es maximal nur H3-Oberbegriffe geben darf und keine

H4-Oberbegriffe existieren dürfen. Außerdem kann ein Oberbegriff nur als Oberbegriff verschoben werden, jedoch nicht als Merkmal oder Feld, gleiches gilt analog für Merkmale und Felder natürlich auch. Bei Merkmalen gilt zudem, dass kein anderes Merkmal, welches dem Oberbegriff zugeordnet ist, die gleiche Bezeichnung tragen darf, wie das ausgewählte Merkmal. Andernfalls wäre sonst die Eindeutigkeit der Kurzbezeichnungen verletzt. Gleiches gilt analog für Felder wenn sie verschoben werden sollen.

```
$("form#move").submit(function(){
var checkH1 = $("select#H1move option:selected").attr('value');
...
var checkFE = $("select#FEmove option:selected").attr('value');

if(checkH1 != 0 && checkH2 != 0 && checkH3 != 0 && checkME != 0
&& checkFE != 0){
alert("Fehler! Es wurde keine Option gewaehlt, sodass der Eintrag
verschoben werden kann!");
}
else{
if((checkH1 == 0 || checkH2 == 0 || checkH3 == 0)
&& (selectedEntryTyp !="O")){
alert("Fehler! Ein Merkmal oder ein Feld kann nicht als
Oberbegriff verschoben werden!");
}
else{
if(checkH1 == selectedEntry || checkH2 == selectedEntry
|| checkH3 == selectedEntry || checkME == selectedEntry){
alert("Fehler! Sie koennen den ausgewaehlten Eintrag
nicht unter sich selbst verschieben!");
}
else{
if(checkH1 == 0 && selectedEntryTyp =="O"){
$.post("move_H1.php", {id:selectedEntry}, function(data){
if(data > 0){
alert("Der Eintrag wurde erfolgreich verschoben!");
parent.location=('Modell.php');
}
if(data == -1){
alert("Fehler! Der Eintrag wurde nicht verschoben!
Wiederholen Sie bitte den Vorgang!");
}
if(data < -1){
alert("Fehler!");
}
});
}
...
}
```

4.3. Serverseitige Implementierung des Webtools

Die Aufgabe des Servers besteht in erster Linie darin, den aktuellen Stand der SQL-Datenbank wiederzugeben und die vom Nutzer gewünschten Änderungen an dieser vorzunehmen. Dies geschieht durch die „select.class.php“-Skriptdatei, denn diese beinhaltet die Klasse „SelectList“, worunter sich alle Funktionen zum Abruf und Verändern der Daten auf der SQL-Datenbank befinden. Wenn der Client eine Anforderung an den Server stellt, so wird immer die dazugehörige Funktion mit den dazu benötigten Parametern übermittelt.

4.3.1. Verbindungsaufbau zur SQL-Datenbank

Eine dauerhafte Verbindung vom Server zur SQL-Datenbank ist nicht notwendig, dadurch wird diese für jede Anforderung vom Client neu aufgebaut und nach Abschluss wieder getrennt. Die Verbindung der Datenbank erfolgt dabei einmalig über dem Konstrukt „public function __construct()“, wodurch die Funktion „DbConnect()“ aufgerufen wird, wie im nachfolgenden Codeauszug dargestellt.

```
class SelectList
{
    protected $connect;

     public function __construct()
        {
            $this->DbConnect();
        }

        protected function DbConnect()
        {
            include "config.php";
            $this->connect = mysql_connect ($host,$user,$password)
            or die ("keine Verbindung m"oglich. Benutzername oder
            Passwort sind falsch");
            mysql_select_db($dbname,$this->connect) or die
            ("Die Datenbank existiert nicht.");
            return TRUE;
        }
...
}
```

In der „config.php“-Datei stehen dabei alle notwendigen Parameter. Falls sich deren Inhalt ändern soll, so müssen sie nur einmal in der „config.php“-Datei geändert werden und nicht an jeder Stelle in der „select.class.php“-Datei. Unter „$host“ steht der Adressname, unter welchem sich die SQL-Datenbank ansprechen lässt. Der Parameter „$user“ beinhaltet den Benutzernamen, welcher alle nötigen Lese- und Schreibrechte hat und das Recht besitzt Einträge zu löschen. In „$password“ steht das zum „$user“ gehörende Passwort und „$dbname“ ist der Name der Datenbank auf der sich die SQL-Tabelle mit

den Begriffen befindet. In „$dbtable“ steht noch der Name der SQL-Tabelle mit den Begriffen und Parametern wie „$valuecol“, welche die Spaltennamen der SQL-Tabelle beinhalten.

4.3.2. Abruf von Begriffen und Eigenschaften

Damit es dem Nutzer möglich ist, die Begriffe in der verketteten Auswahlliste auch richtig auszuwählen, muss der Client diese vom Server anfordern. Dies geschieht über eine Selektion der Spalten „$labelcol“ und „$valuecol“. Über eine „WHILE“-Schleife wird auf dem Server der HTML-Code für den Client vorbereitet und anschließend an diesen geschickt. Im nachfolgenden Codeauszug sei dies für H1-Oberbegriffe mit der Funktion „ShowH1()“ beschrieben.

```
public function ShowH1()
{
    include "config.php";
    $sql = "SELECT $labelcol,$valuecol FROM $dbtable
    WHERE $heinscol = '' AND $merkmalcol='' AND $datentypcol='0'";
    $res = mysql_query($sql,$this->connect)OR die(mysql_error());
    $H1 = '<option value="0">Option w&auml;hlen</option>';
    while($row = mysql_fetch_array($res))
    {
$H1 .= '<option value="' . $row[$valuecol] . '">' . $row[$labelcol]
. '</option>';
    }
    return $H1;
}
```

Die Anforderung von anderen Oberbegriffen sowie von Merkmalen und Feldern erfolgt ähnlich. Im nachfolgenden Codeauszug sei beispielhaft beschrieben, wie über die Funktion „ShowMEfromH1()“ zu einem gewähltem H1-Oberbegriffe die dazugehörigen Merkmale aus der SQL-Tabelle herausgesucht werden. Die Funktion unterscheidet sich nicht sonderlich von der bereits gezeigten „ShowH1()“ -Funktion. Lediglich die „WHERE“-Klausel in der SQL-Anfrage ist anders. In dieser wird sich nur auf Einträge beschränkt, welche unter dem gewählten H1-Oberbegriff stehen und bei denen es sich um ein Merkmal handelt. Dafür muss die Spalte „H1“ mit der Kurzbezeichnung des H1-Oberbegriffs befüllt sein und „H2“ muss dafür leer sein. Die Kennung, ob es sich um ein Merkmal handelt, lässt sich feststellen, indem die Spalte „Merkmal_Clear“ befüllt ist und „Feld_Clear“ leer ist. Mit den Funktionen „stripslashes“, „strip_tags“ und „mysql_real_escape_string“ erfolgt zudem eine ausreichende Maskierung und Überprüfung der Eingabeparameter, um eine SQL-Injection zu verhindern.

```
public function ShowMEfromH1()
{
    include "config.php";
    $kurzbezid = stripslashes($_POST[id]);
    $kurzbezid = strip_tags($kurzbezid);
```

```
    $kurzbezid = mysql_real_escape_string($kurzbezid);
    $sql = "SELECT $labelcol,$valuecol FROM $dbtable
    WHERE $heinscol = '".$kurzbezid."' AND $hzweicol = ''
    AND $merkmalcol != '' AND $feldcol='';
    $res = mysql_query($sql,$this->connect)OR die(mysql_error());
    $ME = '<option value="0">Option w&auml;hlen</option>';
    while($row = mysql_fetch_array($res))
    {
$ME .= '<option value="' . $row[$valuecol] . '">'
. $row[$labelcol] . '</option>';
    }
    return $ME;
}
```

Bei der Auswahl eines Begriffes werden zudem die Werte der Attribute, welche in den Formularfeldern und Auswahllisten stehen, vom Server angefordert. Dabei werden alle Spalten über den Primärschlüssel „.$kurzbezid." selektiert und in einem Array zur Weiterverarbeitung an den Client gespeichert.

```
public function ShowAttribute()
{
    include "config.php";
    $kurzbezid = stripslashes($_POST['id']);
    $kurzbezid = strip_tags($kurzbezid);
    $kurzbezid = mysql_real_escape_string($kurzbezid);

    $sql = "SELECT $labelcol,$valuecol,...,$feldcol FROM $dbtable
    WHERE $valuecol = '".$kurzbezid."'";
    $res = mysql_query($sql,$this->connect)OR die(mysql_error());
    $result= array();
    while($row = mysql_fetch_array($res))
    {
        $result[] = $row[$labelcol];
        $result[] = $row[$valuecol];
...
        $result[] = $row[$inkraftcol];
    }
    return json_encode($result);
}
```

4.3.3. Verändern und Hinzufügen von Begriffen

Die Funktionen zum Verändern und Hinzufügen eines Begriffs sind vom Ablauf her sehr ähnlich, somit werden diese beiden Aktionen zusammengefasst.

Das Verändern von Begriffen erfolgt durch die Funktion „setAttributes()". Beim Aufruf der Funktion werden zunächst wieder die Eingabeparameter maskiert und

überprüft. Bei der Prüfung wird darauf geachtet, ob es sich um definierte Eingaben handelt. Falls nicht, wird ein Fehlercode mit „RETURN“ und einer Zahl von -2 bis -13 zur Identifizierung des Fehlers für den Client ausgegeben, wie im nachfolgenden Programmcodeauszug zu sehen ist.

```
public function setAttributes()
{
include "config.php";
    $kurzbezid = stripslashes($_POST['id']);
    $kurzbezid = strip_tags($kurzbezid);
    $kurzbezid = mysql_real_escape_string($kurzbezid);

...

if($lang == "" || $kurz == "" || $lang == "0" || $kurz =="0")
{return -2;}
if($dt != "O" && $dt != "M" && $dt != "D" && $dt != "I"
&& $dt != "L" && $dt != "R" && $dt != "S" && $dt != "W")
{return -3;}
if($dt == "O" && ($merkmal != "" || $feld != "")){return -4;}
if($dt == "M" && ($merkmal == "" || $feld != "")){return -5;}
if(($dt == "D" || $dt == "I" || $dt == "L" || $dt == "R"
|| $dt == "S" || $dt == "W") && ($merkmal == "" || $feld == ""))
{return -6;}
if($dt == "W" && ($bf != "J" && $bf != "N")){return -7;}
if(($dt =="M" || $dt == "D" || $dt == "I" || $dt == "L"
|| $dt == "R" || $dt == "S" || $dt == "W")
&& ($pf != "J" && $pf != "N")) {return -8;}
if(($dt =="M" || $dt == "D" || $dt == "I" || $dt == "L"
|| $dt == "R" || $dt == "S" || $dt == "W")
&& ($mp != "J" && $mp != "N")){return -9;}
if($dt == "S"){
if($fl > 0 && (floor($fl) == $fl) && (is_numeric($fl) == 1) ){
//nichts da alles richtig
}
else{
return -10;
}
}
if(($dt == "D" || $dt == "I" || $dt == "L" || $dt == "R"
|| $dt == "S" || $dt == "W") && ($kurz != ($merkmal+'.'+$feld)))
{return -11;}
if($dt == "M" && ($merkmal == "" || $merkmal == "0")){return -12;}
if(($dt == "D" || $dt == "I" || $dt == "L" || $dt == "R"
|| $dt == "S" || $dt == "W") && ($feld == "" || $feld == "0"))
{return -13;}
```

```
...
```

Für den Fall, dass die Kurzbezeichnung geändert wurde, muss geprüft werden, ob die neue Kurzbezeichnung schon von einem anderen Begriff verwendet wird. Wenn keine Fehler vorliegen, erfolgt die Aktualisierung der Logdatei über den Aufruf „$this->writeLog($statement,$kurzbezid);“ und die Ausführung des SQL-Befehls „UPDATE“.

```
...
$check = -1;
/*
* falls die KURZBEZ ge"andert wurde, muss gepr"uft werden,
* ob diese schon verwendet wird
*/
if($kurz != $kurzbezid){
$sqlcheck = "SELECT COUNT($valuecol) as number FROM $dbtable
WHERE $valuecol='".$kurz."'";
    $res = mysql_query($sqlcheck,$this->connect)
    OR die(mysql_error());
    $data = mysql_fetch_assoc($res);
    $check = $data['number'];
    }

if($check > 0){
     /* da die KURZBEZ schon von einem anderen Eintrag
     * verwendet wird passiert nichts weiter
*/ Fehlermeldung wird "uber Modell.php ausgegeben
return 0;
}
    else{
        $statement = "UPDATE";
$this->writeLog($statement,$kurzbezid);

    $sql = "UPDATE $dbtable SET $labelcol = '".$lang."'
    , $valuecol = '".$kurz."' , ... , $feldcol = '".$feld."'
    WHERE $valuecol = '".$kurzbezid."'";

    mysql_query($sql,$this->connect)OR die(mysql_error());
    $check = mysql_affected_rows();
...
```

Falls die Kurzbezeichnung geändert wurde erfolgt noch die Aktualisierung der Kindknoten des veränderten Begriffs.

```
...
    /*
    * "Anderung der Kindknoten falls die KURZBEZ ge"andert wurde
    */
    if($kurz != $kurzbezid){
```

```
if($dt ="O"){
$statement = "UPDATECHILDS";
$this->writeLog($statement,$kurzbezid);
      $sql = "UPDATE $dbtable SET $heinscol = '".$kurz."'
      WHERE $heinscol = '".$kurzbezid."'";
     mysql_query($sql,$this->connect)OR die(mysql_error());
     $sql = "UPDATE $dbtable SET $hzweicol = '".$kurz."'
     WHERE $hzweicol = '".$kurzbezid."'";
     mysql_query($sql,$this->connect)OR die(mysql_error());
     $sql = "UPDATE $dbtable SET $hdreicol = '".$kurz."'
     WHERE $hdreicol = '".$kurzbezid."'";
     mysql_query($sql,$this->connect)OR die(mysql_error());
     }
     if($dt ="M"){
$sql = "SELECT $feldcol FROM $dbtable
WHERE $merkmalcol = '".$kurzbezid."' AND $feldcol !='';
    $res = mysql_query($sql,$this->connect)OR die(mysql_error());
    while($row = mysql_fetch_array($res))
    {
        $feld = $row[$feldcol];
        $newKurz = $merkmal . '.' . $feld;
        $sqlfeld = "UPDATE $dbtable SET $valuecol = '".$newKurz."'
        WHERE $merkmalcol = '".$kurzbezid."' AND $feldcol !='';
         mysql_query($sqlfeld,$this->connect)OR die(mysql_error());
    }
    $statement = "UPDATEMERK";
$this->writeLog($statement,$kurzbezid);
     $sql = "UPDATE $dbtable SET $merkmalcol = '".$kurz."'
     WHERE $merkmalcol = '".$kurzbezid."'";
     mysql_query($sql,$this->connect)OR die(mysql_error());
     }
    }

    return $check;
}
}
```

Beim Hinzufügen eines neuen Begriffs verhält es sich ähnlich, denn es wird lediglich „UPDATE“ durch „INSERT INTO“ ersetzt.

```
public function saveNew()
{
...
$sql = "INSERT INTO $dbtable ($labelcol,$valuecol,...,$feldcol)
VALUES ('".$lang."','".$kurz."',...,'".$feld."')";
...
}
```

4.3.4. Löschen von Begriffen

Das Löschen eines Begriffs sowie aller dazugehörigen Unterbegriffe, Merkmale und Felder erfolgt durch die Ausführung der Funktion „deleteEntry()“. Auch bei dieser Funktion erfolgt zunächst eine Maskierung und Überprüfung der Eingabeparameter sowie eine Aktualisierung der Logdatei. Das Löschen erfolgt anschließend durch den SQL-Befehl „DELETE FROM“, wie im nachfolgenden Codeauszug zu sehen ist.

```
public function deleteEntry()
{
    include "config.php";
$kurzbezid = stripslashes($_POST['id']);
    $kurzbezid = strip_tags($kurzbezid);
    $kurzbezid = mysql_real_escape_string($kurzbezid);

$statement = "DELETE";
$this->writeLog($statement,$kurzbezid);

    $sql = "DELETE FROM $dbtable
    WHERE $valuecol = '".$kurzbezid."' OR $heinscol = '".$kurzbezid."'
    OR $hzweicol = '".$kurzbezid."' OR $hdreicol = '".$kurzbezid."'
    OR $merkmalcol = '".$kurzbezid."'";
    mysql_query($sql,$this->connect)OR die(mysql_error());
return mysql_affected_rows();
}
```

4.3.5. Verschieben von Begriffen

Für das Verschieben von Einträgen gibt es unterschiedliche Funktionen, abhängig davon, ob es sich bei dem zu verschiebenden Begriff um einen Oberbegriff, ein Merkmal oder um ein Feld handelt. Bei den Unterschieden handelt es sich jedoch nur um leichte Abwandlungen beim SQL-Befehls „UPDATE“, somit sei beispielhaft an der Funktion „moveH1()“ erklärt, welche zum Verschieben eines beliebigen Oberbegriffs als ein H1-Oberbegriff genutzt wird. Wie bei allen Funktionen mit Nutzereingaben werden auch hier die Eingabeparameter maskiert und überprüft. Zusätzlich wird die Logdatei mit der Funktion „writeLog“ aktualisiert. Die Änderung an der Datenbank erfolgt dann mittels des SQL-Befehls „UPDATE“, wobei nicht nur der zu verschiebende Begriff aktualisiert wird, sondern auch alle sich daran befindlichen Unterbegriffe, Merkmale und Felder, wie es im nachfolgenden Codeauszug ersichtlich ist.

```
public function moveH1()
{
    include "config.php";
    $kurzbezid = stripslashes($_POST['id']);
    $kurzbezid = strip_tags($kurzbezid);
    $kurzbezid = mysql_real_escape_string($kurzbezid);

    $statement = "MOVE";
```

```
$this->writeLog($statement,$kurzbezid);

    $sql = "UPDATE $dbtable SET $heinscol = '' , $hzweicol = ''
    , $hdreicol = '' WHERE $valuecol = '".$kurzbezid."'";
    mysql_query($sql,$this->connect)OR die(mysql_error());
    $check = mysql_affected_rows();

    /*
* Wenn $kurzbezid vorher ein H2 war
* Zur Ver"anderung der Kindknoten
* (Merkmale und Felder mit eingeschlossen)
*/
    $sql = "UPDATE $dbtable SET $hzweicol = $hdreicol
    ,$heinscol = '".$kurzbezid."' , $hdreicol = ''
    WHERE $hzweicol = '".$kurzbezid."'";
    mysql_query($sql,$this->connect)OR die(mysql_error());

    /*
    * Wenn $kurzbezid vorher ein H3 war
    * Zur Ver"anderung der Merkmale und Felder
*/
    $sql = "UPDATE $dbtable SET $hzweicol = ''
    ,$heinscol = '".$kurzbezid."' , $hdreicol = ''
    WHERE $hdreicol = '".$kurzbezid."'";
    mysql_query($sql,$this->connect)OR die(mysql_error());

    return $check;

    /*
    * Anmerkung
    * Wenn $kurzbezid vorher ein H1 war "andert
    * sich an den allen Untergeordneten Knoten nichts,
    * da der Eintrag wieder als ein H1 verschoben wird
*/
}
```

4.3.6. Aktualisierung der Log-Datei

Abschließend sei noch der Ablauf zur Aktualisierung der Logdatei gezeigt. Die Funktion „writeLog($statement,$kurzbezid)“ wird bei jeder Änderung an der SQL-Tabelle aufgerufen. Der Parameter „$kurzbezid“ ist hierbei wie immer zur Identifizierung des veränderten Eintrags notwendig und der Parameter “$statement‘“, um was für eine Änderung es sich handelt. Im nachfolgenden Codeauszug sei beispielhaft der Ablauf beim Löschen eines Eintrages erklärt. Vor dem Löschen werden alle Spalteninhalte zur zugehörigen „$kurzbezid“ selektiert und in einem Array gespeichert. Anschließend erfolgt die Auswertung des Arrays, dabei wird für jeden geänderten Begriff eine neue Zeile über einen zusammengesetzten String in der Logdatei angelegt. Am Anfang jeder Zeile steht das

Änderungsdatum mit Uhrzeit gefolgt vom Statement, was in diesem Fall ein „DELETE“ ist. Anschließend wird der String mit den Spaltennamen und den jeweiligen Werten vor der Löschung erweitert und am Ende des Strings wird ein Zeilenumbruch mit „\r\n“ angefügt. Zum Schluss erfolgt das Öffnen der Logdaten mit „fopen“, das Schreiben des Strings in die Logdatei mit „fwrite“ und das Schließen der Logdatei mit „fclose“.

```
private function writeLog($statement,$kurzbezid)
{
    include "config.php";
    if($statement =="DELETE"){
$sql = "SELECT $labelcol,$valuecol,...,$feldcol FROM $dbtable
WHERE $valuecol = '".$kurzbezid."' OR $heinscol = '".$kurzbezid."'
OR $hzweicol = '".$kurzbezid."' OR $hdreicol = '".$kurzbezid."'
OR $merkmalcol = '".$kurzbezid."'";
$res = mysql_query($sql,$this->connect)OR die(mysql_error());
while($row = mysql_fetch_array($res))
        {
            $timestamp = time();
            $logstring = date("d.m.Y",$timestamp) . " -- "
            . date("H:i",$timestamp) . " -- ";
            $logstring .= "DELETE -- ";
            $logstring .= "langbez:" . $row[$labelcol] . " -- "
            . "kurzbez:" . $row[$valuecol] . ... . $row[$feldcol]
            ."\r\n";
            $logdatei = fopen("LogDatei.txt", "a+");
            fwrite($logdatei, $logstring);
            fclose($logdatei);
        }
}
...
}
```

5. Evaluation des Webtools

In diesem Kapitel soll erklärt werden, ob die Anforderungen an das Webtool erfüllt wurden oder ob die Implementation von den Anforderungen abweicht und warum dies der Fall ist.

Die Hauptanforderung, dass es mit Hilfe des Webtools möglich sein soll, das Datenmodell der GSBL-Datenbank über einen Browser zu bearbeiten, auch wenn keine Fachkenntnisse zu Datenbanken sowie deren Aufbau und Funktionsweisen beim Anwender vorhanden sind, ist erfüllt. Der Nutzer ruft beispielsweise mit dem „Mozilla Firefox“ die Webseite auf, über welche sich das Webtool öffnen lässt, korrekt wiedergibt und es ermöglicht, darauf zu arbeiten. Der „Mozilla Firefox“ ist als Browser für die Nutzung des Webtools empfohlen. Während der Entwicklung wurde die Lauffähigkeit auf diesem Browser getestet, weshalb dessen Funktionsfähigkeit auf diesem Browser garantiert werden kann. Beim Test mit anderen Internetbrowsern, wie dem „Internet Explorer 11“ hat es sich jedoch gezeigt, dass das Webtool auf diesen nicht vollständig läuft. Nach Rücksprache mit dem Auftraggeber ist dieser Mangel jedoch im Toleranzrahmen der prototypischen Entwicklung und die Lauffähigkeit auf mehreren Browsern ist laut Auftraggeber eine Aufgabe für zukünftige Arbeiten. Des Weiteren ist die Oberfläche des Webtools, wie in den Abbildungen 4.1 bis 4.3 zu sehen ist, relativ einfach gestaltet und lässt sich über „CSS“ in ein vom Auftraggeber gewünschtes Design überführen.

Wie in Abschnitt 4.1 beschrieben, wurde das Schema des Datenmodells erfolgreich von einer Excel-Tabelle über das CSV-Format in eine SQL-Datenbanktabelle überführt. In dieser Tabelle werden künftig alle Veränderungen, welche über das Webtool erfolgen, am Datenmodell gespeichert. Dies gilt jedoch nur für Begriffe, wie Oberbegriffe, Merkmale und Feldern. Bei der Abnahme des Webtools wurde jedoch festgestellt, dass die Sektionen „Lieferanten“, „Zitate“ und „Spezies“ nicht bearbeitet werden können, aufgrund deren gesonderten Aufführung im Datenmodell und es während der Implementation nicht ersichtlich war, dass diese Begriffe nicht berücksichtigt werden. Der Grund dafür liegt an deren Eigenschaften, wie zum Beispiel dem Datentyp. Keiner dieser Begriffe wird als Oberbegriff (mit Datentyp „O“), als Merkmal (mit Datentyp „M“ oder als Feld (mit Datentyp „D“, „I“, L“, „R“, „S“, oder „W“) geführt. Auch hier sieht der Auftraggeber die Einbeziehung von „Lieferanten“, „Zitate“ und „Spezies“ aufgrund ihrer Speziellen Behandlung im Datenmodell in zukünftigen Arbeiten, weshalb diese Sektionen im Rahmen dieser Arbeit nicht mehr mit einbezogen wurden.

Durch die verketteten Auswahllisten wurde eine geeignete Navigation gefunden, die es dem Nutzer ermöglicht, zu sehen, wie die Begriffe hierarchisch mit einander verknüpft sind. Der Nutzer kann zwar nicht die gesamte Begriffshierarchie auf einem Blick erfassen, sondern immer nur einen Zweig. Dies hat jedoch den Vorteil, dass der Nutzer nicht so schnell den Überblick verlieren kann und er nicht über die gesamte

Webseite hinweg nach einem Begriff suchen muss, sondern nur in den Auswahllisten nach einem bestimmten Begriff sucht. Dabei ist es natürlich von Vorteil, wenn der Nutzer zumindest die Hierarchie-Stufe des gesuchten Begriffes kennt. Durch die Nutzung von AJAX wurde außerdem ein dynamisches Laden während der Bearbeitung ermöglicht, weil Unterbegriffe und Eigenschaften eines Begriffes erst bei der Auswahl des Begriffes vom Server angefordert werden. Ein Neuladen der Webseite erfolgt lediglich bei einer erfolgreich durchgeführten Änderung am Datenmodell. Mit der jetzigen Navigation über den verketteten Auswahllisten ist der Auftraggeber aufgrund der wie bereits erwähnten intuitiven Auswahl der Begriffe und der für den Auftraggeber guten Übersicht des Webtools sehr zufrieden.

Die Aufteilung der Funktionen, wozu das Löschen von Begriffen, das Hinzufügen von neuen Begriffen sowie das Verschieben und Bearbeiten von bereits vorhandenen Begriffen gehört, auf drei Seiten wurde vom Auftraggeber sehr positiv aufgenommen. Diese Aufteilung gewährleistet eine einfachere Bedienung des Webtool, denn anstatt alle möglichen Optionen zur Bearbeitung des Datenmodells auf einer Seite bereitzustellen, werden diese auf drei Seiten verteilt und ein Nutzer, welcher mit dem Webtool noch nicht vertraut ist, hat es einfacher zu erkennen, welche Eingaben zur Veränderung des Datenmodells von ihm verlangt werden.

Das Bearbeiten von bereits vorhandenen Begriffen und deren Eigenschaften erfolgt auf der Hauptseite. Um zu gewährleisten, dass nur Änderungen vorgenommen werden, welche auch mit der Definition des Datenmodells konform gehen, erfolgt eine Überprüfung der Nutzereingaben. Wenn Veränderungen vorgenommen werden, so werden diese von der Funktion „setAttributes()“ serverseitig überprüft. Bei dieser Überprüfung werden, wie in Abschnitt 4.3.3 anhand des Programmcodeauszugs zu sehen ist, die Nutzereingaben auf ihre Integrität hin überprüft. Bei Fehlern wird ein Fehlercode an den Client zurückgegeben und von ihm ausgewertet. Je nach Fehlercode wird eine für den Nutzer nachvollziehbare Fehlermeldung ausgegeben, sodass dieser entsprechende Änderungen an seinen Eingaben vornehmen kann. Durch den Aufbau der Fehlerüberprüfung ist es außerdem leicht möglich, weitere Fälle einzubauen oder bereits vorhandene Fälle von Fehlern weiter zu differenzieren, um den Nutzer noch eindeutigere Fehlermeldungen zurückgeben zu können.

Die Anforderung zum Löschen von Begriffen wurde umgesetzt und erfolgt ebenfalls auf der Hauptseite. Wie in Abschnitt 4.3.4 anhand des Programmcodeauszugs zu sehen ist, wird beim Löschen nicht nur der Begriff an sich gelöscht, sondern es werden auch alle dem Begriff zugeordneten Unterbegriffe, Merkmale und Felder mit gelöscht.

Die Anforderung zum Hinzufügen von neuen Begriffen in das Datenmodell besagt, dass der Anwender zunächst über die Navigation den Oberbegriff auswählt, unter dem der neue Begriff stehen soll und anschließend über die Maske die Werte zu dessen Eigenschaften einträgt. Die Erfüllung dieser Anforderung erfolgt durch die Seite „NeuerEintrag.php“, welche sich mittels eines Buttons über die Hauptseite aufrufen lässt. Auch hier erfolgt serverseitig eine Überprüfung der Nutzereingabe, wie es beim Bearbeiten von Begriffen der Fall ist. Bei fehlerhaften Nutzereingaben wird ebenfalls ein Fehlercode an den Client übermittelt, um eine entsprechende Rückmeldung an den

Nutzer zu ermöglichen.

Das Verschieben von bereits vorhandenen Begriffen erfolgt auf der Seite „Eintrag-Verschieben.php“, welche sich über einen Button auf der Hauptseite erreichen lässt. Die Anforderung, dass auch alle Unterbegriffe des zu verschiebenden Begriffs ebenfalls korrekt mit verschoben werden, ist, wie in Abschnitt 4.3.5 gezeigt wurde, erfüllt. Vom Auftraggeber kam noch der Wunsch, dass ein Nutzer auch die Möglichkeit hat die Begriffe in umgekehrter Auswahl zu bestimmen, also das erst der Begriff ausgewählt wird unter dem der zu verschiebende Begriff stehen soll und erst dann der zu verschiebende Begriff ausgewählt wird. Diese umgekehrte Möglichkeit der Begriffsauswahl wurde jedoch nicht implementiert, denn wie in Abschnitt 4.2.2.3 beschrieben gibt es unterschiedliche Fälle, je nachdem ob es sich bei dem zu verschiebenden Begriff um einen Oberbegriff, Merkmal oder Feld handelt. Jeder dieser Fälle wird gesondert behandelt und es erfolgt eine Einschränkung der Auswahlmöglichkeiten, worunter der zu verschiebende Begriff verschoben werden kann. Diese Einschränkung würde es bei einer umgekehrten Auswahlmöglichkeit nicht geben, weshalb der Nutzer erst bei der Bestätigung des Änderungsversuches erfahren würde, ob seine Änderung zulässig ist. Mit dieser Argumentation, dass eine umgekehrte Auswahlmöglichkeit der Begriffe auch eine kompliziertere Bedienung nach sich zieht, wurde auf eine Implementation dieser Auswahlmöglichkeit verzichtet.

Die Forderung einer Log-Datei, in der alle Änderungen an der Datenbank festgehalten werden sollen, wie in Abschnitt 4.3.6 beschrieben, ist erfüllt. Bei dieser Logdatei handelt es sich um eine einfache Textdatei, welche den Zustand der Datenbank vor einer Änderung angibt, um die Rücknahme einer Änderung ermöglichen zu können.

Das Aussehen des Webtools im „Mozilla Firefox“ wurde bereits in den Abbildungen 4.1 bis 4.3 gezeigt. Diese Abbildungen stellen den finalen Stand der prototypischen Umsetzung, welche im Rahmen dieser Arbeit erarbeitet wurde, des Webtools dar.

6. Zusammenfassung und Ausblick

Das Ziel dieser Arbeit war es eine prototypische Umsetzung eines Webtool zu erstellen, mit dem es möglich ist das Datenmodell der GSBL-Datenbank zu bearbeiten. Zum Verständnis wurden im Grundlagenkapitel alle wichtigen Datenbankbegriffe und die Grundlagen zur Funktionsweise des Internets erläutert. Weiterhin wurden alle Technologien, welche bei der Erstellung des Webtools zum Einsatz kamen, erklärt. Außerdem wurde in Kapitel 3 das Datenmodell des GSBLs und der GSBL selbst näher erklärt. Damit der Soll-Stand des Webtools nachvollziehbar ist, wurden zudem die Anforderungen des Auftraggeber an das Webtool klar formuliert.

In Kapitel 4 wurde dargelegt, wie das Webtool sowohl clientseitig als auch serverseitig implementiert wurde. Dazu gehörte die Erklärung, warum die verwendeten Technologien eingesetzt wurden und ein Überblick, wie die prototypische Umsetzung des Webtools erarbeitet wurde. Die Funktionen zum Bearbeiten des Datenmodells wurden auf drei Seiten aufgeteilt, der Hauptseite zum Bearbeiten und Löschen von bereits vorhanden Einträgen, einer Seite zum Anlegen neuer Begriffe und eine Seite zum Verschieben von bereits vorhandenen Begriffen. Die Programmierung dieser Funktionen wurde anhand von beispielhaften Programmcodeauszügen erläutert, wobei stets die client- und serverseitige Programmierung getrennt voneinander behandelt wurde.

Im Zuge der Verifikation wurde verglichen, ob die prototypische Umsetzung des Webtools den in Kapitel 3 gestellten Anforderungen genügt. Es wurde gezeigt, dass alle Funktionen zum Bearbeiten des Datenmodells des GSBLs vorhanden sind, dazu gehören das Bearbeiten, Löschen und Verschieben von bereits vorhandenen Begriffen sowie das Erstellen von neuen Begriffen. Die Verteilung dieser Funktionen auf drei Seiten wurde vom Auftraggeber positiv aufgenommen und mit der Art der Begriffsnavigation über verkettete Auswahllisten und der dynamischen Ladung der Begriffe und ihrer Eigenschaften ist der Auftraggeber vollends zufrieden. Es wurde gezeigt, dass das Webtool bereits vom Auftraggeber genutzt werden kann, um das Datenmodell des GSBLs ordnungsgemäß zu bearbeiten.

Das Webtool ist in seiner aktuellen Version lauffähig und einsatzbereit, jedoch gibt es noch einige Funktionen, welche in zukünftigen Versionen verbessert, beziehungsweise mit eingepflegt werden sollten. Einige dieser Verbesserungen wurden bereits im Zuge der Verifikation erwähnt. Dazu gehören, dass das Webtool auch im „Internet Explorer 11“ lauffähig sein soll. Wichtig ist zudem, dass, wie bereits in Kapital 5 erwähnt, auch andere Sektionen wie Lieferanten, Zitate und Spezies mit einbezogen werden müssen. Außerdem müssten Nachschlagetabellen (Kataloge) und Einheitentabellen (Einheiten-Kataloge) noch implementiert werden. Zudem wurde bei der Abnahme des Webtools von der Fachabteilung bemängelt, dass die Logdatei nicht übersichtlich genug ist, dies sollte zukünftig noch verbessert werden.

Ein weiterer Aspekt ist die Überarbeitung von Änderungen. Wenn eine Änderung vorgenommen wird, so wird das Webtool komplett neu geladen. Um der Anforderung des dynamischen Nachladens von Begriffen noch besser gerecht zu werden, sollte die Webseite nicht komplett nachgeladen, sondern nur der aktuelle Stand der Datenbanktabelle, in der das Datenmodell erfasst wird. Vorgenommene Änderungen sind aktuell endgültig und lassen sich nur manuell durch zur Hilfenahme der Logdatei rückgängig machen. Deshalb sollte zukünftig noch eine Möglichkeit existieren, Änderungen auf Wunsch automatisiert zurückzunehmen. Zudem weiß der Nutzer erst, ob seine vorgenommenen Änderungen gültig sind, wenn er diese speichern möchte. Aus diesem Grund sollte der Client schon bei der Eingabe von Änderungen eine Rückmeldung an den Nutzer geben, ob diese zulässig sind. Dies lässt sich beispielsweise durch das Markieren von Pflichtfeldern und dem Vorschlagen von erlaubten Eingaben realisieren. Des Weiteren dürfen zukünftig einige kritische Änderungen nicht gleich durchgeführt, sondern sollten zunächst nur gesondert markiert werden. Dies ist vom Auftraggeber her gewünscht, weil diese Änderungen erst durch eine zuständige Kontrollgruppe freigegeben werden müssen, bevor sie endgültig in die Datenbank übernommen werden. Eine weitere wichtige Änderung, die am Webtool vorgenommen werden muss, ist die Synchronisation, also die Koordination von parallelen Transaktionen von mehreren Benutzern. Dies war in der ersten prototypischen Umsetzung des Webtools nicht vom Auftraggeber gefordert, jedoch soll zukünftig mehr als nur ein Nutzer gleichzeitig mit dem Webtool arbeiten können, womit die Synchronisation zukünftig noch einprogrammiert werden muss. Zukünftig sollen mehrere Personen mit dem Webtool arbeiten können, weshalb es außerdem notwendig ist, verschiedene Benutzersichten zu implementieren, damit ein Nutzer nur die Bereiche des Datenmodells einsehen und bearbeiten kann, zu denen er auch autorisiert ist. Zur Gewährleistung des Datenschutzes muss zudem eine Validierung in Form eines Log-Ins mit der Angabe von Nutzername und Passwort erfolgen, um den Ausschluss unautorisierter Zugriffe verhindern zu können.

Abschließend muss noch erwähnt werden, dass keine vollständige Lauffähigkeit in allen Webbrowsern und auf allen Systemen garantiert wird, hier muss je nach System das Webtool bei Bedarf angepasst werden.

Literaturverzeichnis

[BMU] BMUB. `http://www.bmub.bund.de/themen/umweltinformation-bildung/umweltinformation/stoffdatenpool-gsbl` Aufgerufen am 20.10.2015.

[Bri09] Brinzarea, B.: *AJAX and PHP building modern web applications*. Packt Pub, Birmingham, UK, 2009.

[Cod70] Codd, E. F.: A relational model of data for large shared data banks. *Communications of the ACM*, Band 13, S. 377–387, 1970.

[GSB] GSBL. `http://www.gsbl.de` Aufgerufen am 20.10.2015.

[HR83] Haerder, T.; Reuter, A.: Principles of transaction-oriented database recovery. *ACM Computing Surveys (CSUR) Surveys Homepage archive*, Band 15, Nr. 4, S. 287–317, 1983.

[Kri08] Kriegel, A.: *SQL bible*. Wiley John Wiley distributor, Hoboken, N.J. Chichester, 2008.

[MyS] MySQL. `https://www.mysql.de` Aufgerufen am 05.11.2015.

[oDBMS75] Data Base Management Systems, S. G. o.: Interim report. fdt. *ACM SIGMOD bulletin*, Band 7, Nr. 2, 1975.

[Pom12] Pomaska, G.: *Webseiten-Programmierung Sprachen, Werkzeuge, Entwicklung*. Springer Vieweg, Wiesbaden, 2012.

[Sal95] Salus, P.: *Casting the net : from ARPANET to Internet and beyond*. Addison-Wesley Pub. Co, Reading, Mass, 1995.

[Sch14] Schicker, E.: *Datenbanken und SQL Eine praxisorientierte Einführung mit Anwendungen in Oracle, SQL Server und MySQL*. Springer Vieweg, Wiesbaden, 2014.

[SSH10] Saake, G.; Sattler, K.-U.; Heuer, A.: *Datenbanken Konzepte und Sprachen*. mitp, Heidelberg München Landsberg Frechen Hamburg, 2010.

[Tan12] Tanenbaum, A.: *Computernetzwerke*. Pearson, München u.a, 2012.

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur mit erlaubten Hilfsmitteln angefertigt habe.

Magdeburg, den 21. Februar 2016

Philipp Müller