Universität Ulm
Abteilung Angewandte Informationsverarbeitung
Abteilung Systematische Botanik und Ökologie


# Entwurf und Umsetzung eines konzeptbasierten Systems zur biologischen Taxonomie

Dissertation

zur Erlangung des Doktorgrades Dr. rer. nat.
der Fakultät Mathematik und Wirtschaftswissenschaften
der Universität Ulm

vorgelegt von
Thorsten Ludwig
aus Senden

Oktober 2005

**Amtierender Dekan:** Prof. Dr. Ulrich Stadtmüller

**Erster Gutachter:** Prof. Dr. Franz Schweiggert

**Zweiter Gutachter:** Prof. Dr. Gerhard Gottsberger

**Tag der Promotion:** 14.12.2005

# Danksagung

Die vorliegende Dissertation entstand als interdisziplinäre Arbeit während meiner Tätigkeit als wissenschaftlicher Angestellter in der Abteilung Spezielle Botanik und Ökologie.

In erster Linie bedanke ich mich bei Herrn Prof. Dr. Franz Schweiggert, der die Betreuung von Seiten der Informatik übernahm. Nicht nur durch seine äußerst kompetente fachliche Unterstützung, sondern auch durch seine menschliche und vor allem motivierende Art trug Herr Prof. Schweiggert sehr zum Gelingen dieser Arbeit bei. Des Weiteren danke ich ihm für die Bereitschaft, den SysTax-Datenbankserver in das Abteilungsnetzwerk einzubinden und dort zu administrieren.

Mein besonderer Dank gilt Herrn Prof. Dr. Gerhard Gottsberger für die Übernahme des Zweitgutachtens und vor allem dafür, dass er mir einen Arbeitsplatz in den Räumlichkeiten der Abteilung Systematische Botanik und Ökologie zur Verfügung stellte.

Bei Herrn Dr. Jürgen Hoppe, Frau Dr. Evelin Boos, Herrn Michael Wiedemann sowie Frau Dr. Sigrun Bopp bedanke ich mich für die konstruktive Zusammenarbeit im Projekt SysTax. Die vielen Diskussionen waren mir eine große Hilfe, nicht nur beim Anfertigen dieser Arbeit, sondern auch bei der praktischen Umsetzung der hier vorgestellten Modelle.

Dank gilt auch den Kolleginnen und Kollegen der Abteilung Spezielle Botanik und Ökologie für die freundliche Aufnahme eines Nichtbiologen im Kreis der Botaniker.

Den Mitarbeiterinnen und Mitarbeitern der Abteilung Angewandte Informationsverarbeitung danke ich für die schnelle und kompetente Hilfe bei Fragestellungen aller Art. Insbesondere für Herrn Dr. Andreas Borchert bedeuteten die vielen Sonderwünsche von „uns Biologen“ einen nicht unerheblichen Mehraufwand bei der Administration des Abteilungsnetzwerkes. Dafür gebührt ihm mein aufrichtiger Dank.

Für das aufmerksame Durchsehen und Korrekturlesen dieser Arbeit bedanke ich mich bei Frau Carina Stiefel und Frau Dr. Sigrun Bopp, die mich auch geduldig in die Geheimnisse der neuen deutschen Rechtschreibung einweihte.

Insbesondere möchte ich an dieser Stelle meinen Eltern danke sagen, die mir die schulische und universitäre Ausbildung so selbstverständlich erscheinen ließen haben und die mich von jeher in jeder Hinsicht unterstützt haben.

Senden, im September 2005 Thorsten Ludwig

# Inhaltsverzeichnis

# 1 Einleitung

## 1.1 Das Projekt SysTax

Ausschlaggebend für die Entwicklung von SysTax war vor etwa 25 Jahren die Idee, ein EDV-System für die Arbeiten der Abteilung Spezielle Botanik[1] an der Universität Ulm einzuführen, insbesondere für die Verwaltung des Herbariums und des Botanischen Gartens Ulm. Alle damals verfügbaren Systeme entsprachen nicht den gestellten Anforderungen (Boos, 1992), sodass eine eigene Datenbank entwickelt werden sollte. Ein erster Prototyp entstand 1988 in Zusammenarbeit mit der Sektion für Angewandte Informationsverarbeitung[2] der Fakultät Mathematik und Wirtschaftswissenschaften im Rahmen der Diplomarbeit von Dietmar Kümmel, bei der vor allem die Frage nach einer geeigneten Datenbanksoftware im Vordergrund stand. Die Entscheidung fiel auf das relationale Datenbankmanagementsystem Oracle, das aufgrund der zu erwartenden Datenmenge und vor allem wegen der Mehrbenutzerfähigkeit innerhalb eines Netzwerkes am geeignetsten erschien.

Dieser Prototyp wurde von Frau Dr. Boos während ihrer Promotion (1989 bis 1992) weiterentwickelt, sodass Daten aus den Bereichen „botanische Taxonomie und Synonymie“, „Literatur“, „Verbreitung“, „Herbarium“ und „botanischer Garten“ verwaltet und miteinander verknüpft werden konnten. Das Besondere an SysTax war, dass nicht nur mehrere Nutzer gleichzeitig mit diesem System arbeiten konnten, sondern dass es von Anfang an „multi-institutionell“ konzipiert war, d. h. Daten mehrerer Institutionen mit Hilfe eines Rechtesystems parallel verwaltetet werden konnten.

Eingesetzt wurde zuerst ein Compaq Portable III mit Oracle 5.1.A, später kam ein IBM PS70 mit Oracle 5.1.B hinzu. Ab 1990 wurde ein Hewlett-Packard-Rechner (HP 9000/825) mit Oracle Version 6 von der Abteilung Theoretische Chemie als Produktivsystem zur Verfügung gestellt. Parallel dazu wurde aus Mitteln der Deutschen Forschungsgesellschaft (kurz: DFG) ein weiterer HP-Rechner als Entwicklungssystem beschafft, der später weiter ausgebaut und ebenfalls als Produktivumgebung eingesetzt wurde. Die Benutzeroberfläche (im Folgenden auch „Masken“ genannt) wurde mittels sql*forms erstellt, Client-Rechner konnten sich über ein extra hierfür modifiziertes Telnet-Programm mit der Datenbank verbinden und die Masken ausführen.

Mitte der neunziger Jahre wurde SysTax weitestgehend durch die Abteilung Spezielle Botanik finanziert, in Bezug auf die Funktionalität erweitert und den Wünschen neu hin-

[1] heute: Abteilung Systematische Botanik und Ökologie

[2] heute: Abteilung für Angewandte Informationsverarbeitung

zukommender Nutzer sowie den sich ändernden Ansprüchen bestehender Nutzer angeglichen. Insbesondere stand die Weiterentwicklung der Herbarverwaltung im Vordergrund, was vor allem die Georeferenzierung der Fundorte, das Leihsystem und die Ausgabe des Herbaretiketts umfasste. Zusätzlich konnte aus den Berufungsmitteln von Herrn Prof. Dr. Gottsberger, der die Leitung der Abteilung Systematische Botanik und Ökologie seit 1993 innehat, ein HP 9000/E55 beschafft werden. Später wurde SysTax auf die Oracle-Version 7 portiert und, finanziell unterstützt durch ein Frauenförderungsprogramm, mit der Umprogrammierung der Masken in Oracle-Forms 4.5 begonnen. Damit stand erstmalig eine grafische Benutzeroberfläche unter Microsoft Windows zur Verfügung. Die Kommunikation zwischen Datenbank und Client erfolgte bei Forms über TCP/IP mit dem Oracle-internen Protokoll Net7. Auf demselben Rechner lief ebenfalls ein Webserver, der zum ersten Mal eine dynamische Präsentation der SysTax-Daten im World Wide Web mittels CGI-Programmen ermöglichte.

Am 01.04.1998 startete das vom Bundesministerium für Bildung und Forschung (kurz: BMBF) finanzierte, auf drei Jahre befristete Projekt „Bundesinformationssystem Genetischer Resourcen“ (kurz: BIG), das es sich zur Aufgabe gemacht hat, biologische, genetische und ökologische sowie ökonomische und geographische Informationen zu Wild- und Kulturpflanzen Deutschlands aus verschiedenen Datenbanken in einem zentralen Portal im Internet abrufbar zu machen. Dazu wurde ein Datenaustauschformat erstellt, das auf XML basiert. Auf der Seite von SysTax wurde die Funktionalität implementiert, Suchanfragen, die vom Portal mittels XML über HTTP an die abzufragenden Datenbanken geschickt werden, zu verarbeiten und die gefundenen Daten als XML-Dokument, ebenfalls über HTTP, zurück an das Portal zu senden.

Im Jahr 2000 initiierte das BMBF unter dem Deutschen Zentrum für Luft- und Raumfahrt als Träger das Projekt „Biodiversität und Global Change“ (kurz: BIOLOG)[3]. Damit wurden nicht nur Forschungsprojekte auf dem Gebiet der Artenvielfalt (Biodiversität), sondern erstmalig auch direkt Informatik-Projekte gefördert, die sich mit Biodiversitätsdaten beschäftigen.

Am 01.04.2000 startete im Rahmen des BIOLOG-Programms das auf drei Jahre befristete Projekt „Entomological Data Information System“ (EDIS)[4], das es sich als Kooperation zwischen mehreren deutschen naturhistorischen Museen und Universitäten zur Aufgabe gemacht hat, digitalisierte Daten über Insekten und andere Arthropoden zusammenzutragen und im Internet zu veröffentlichen. Dies betrifft vor allem die Erfassung der in den Museen vorhandenen Sammlungsbelege[5], die Aufschluss über die Verbreitung von Arten in der Vergangenheit geben können, da Datum und Ort des Fundes durch sie protokolliert sind. Daneben sollen Daten aus der Literatur sowie taxonomische, multimediale und molekulare Daten über diese Tiergruppe gesammelt werden. Die einzelnen

[3] http://www.biolog-online.info (Juni 2005)

[4] http://www.insects-online.de (Juni 2005)

[5] Aufgrund der großen Anzahl an Sammlungsbelegen in den einzelnen Einrichtungen wurde die digitale Erfassung zunächst auf die Typusexemplare beschränkt.

Projektpartner haben beschlossen, SysTax als zentrales Datenbanksystem für die Speicherung und Veröffentlichung ihrer Daten zu verwenden.

Aus Projektmitteln konnte Mitte 2000 ein neuer Server, Modell SUN Enterprise 450, beschafft werden, der dankenswerterweise von der Abteilung für Angewandte Informationsverarbeitung administriert wird. Die Datenbank wurde auf Oracle 8i, die Masken auf Forms 6i portiert.

Im Rahmen des EDIS-Projekts bestand die Notwendigkeit, den Programmteil für die zoologische Taxonomie und Synonymie neu zu entwickeln, die Multimediadatenverwaltung auszubauen und Importschnittstellen zu entwerfen. Entsprechend wurden die Masken und die Webausgaben erweitert, um den Anforderungen der Zoologen Rechnung zu tragen.

Ende des Jahres 2000 wurde die Organisation „Global Biodiversity Information Facility“ (kurz: GBIF)[6] auf Grundlage eines internationalen Vertragswerks gegründet, dem bis Mitte 2005 insgesamt 67 Staaten beigetreten sind, darunter auch Deutschland als Gründungsmitglied. Die Aufgabe von GBIF ist es, weltweit Projekte zur Erfassung und Veröffentlichung von Biodiversitätsdaten zu fördern. Insbesondere lassen sich dadurch Änderungen der globalen Artenvielfalt feststellen und protokollieren, wie sie möglicherweise durch Klimaveränderungen hervorgerufen sein können.

Die unterzeichnenden Staaten haben sich verpflichtet, nationale Biodiversitätsprojekte zu finanzieren, die ihre Daten dem GBIF-Netzwerk zur Verfügung stellen. In Deutschland wurde beschlossen, für jede Organismengruppe so genannte „GBIF-Knoten“ zu bilden: je ein Knoten für die Prokaryonten, die Botanik, die Mykologie (Pilzkunde) und die Vertebrata (Wirbeltiere) sowie zwei Knoten für die Evertebrata (wirbellose Tiere; Evertebrata I und II). Viele EDIS-Projekte wurden mit Ablauf des EDIS-Programms Ende 2002 in einem der beiden Evertebrata-Knoten weitergeführt. Deswegen haben sich die beiden Evertebrata- sowie der Vertebrata-Knoten entschlossen, ihre Daten weiterhin an SysTax zu liefern. Damit ist SysTax seit 2003, finanziert durch das BMBF bis Ende 2005, Datenlieferant an das GBIF-Netzwerk.

Aufgrund der großen zu erwartenden Datenmenge, insbesondere von Bildern, wurde 2003 ein Raid-Array mit etwas mehr als einem Terabyte Netto-Speicherkapazität[7] beschafft. Zusätzlich wurden im Rahmen einer weiteren Promotion die technischen Voraussetzungen geschaffen, um Datenabfragen in XML-Form aus dem GBIF-Netzwerk beantworten sowie selber andere GBIF-Datenquellen abfragen zu können.

## 1.2 Begriffserklärungen

Ganz allgemein beschäftigt sich die *Taxonomie* mit der Einteilung von Objekten bzw. im Fall der Biologie von Organismen in disjunkte Gruppen. Sie untersucht und beschreibt

[6] http://www.gbif.org (Juli 2005)

[7] acht 200 MB Festplatten, konfiguriert als Raid-5 mit einer Hot-Spare-Festplatte

die Unterschiede zwischen diesen Objekten und erforscht die Gründe für diese Unterschiede sowie die sich daraus ergebenden Konsequenzen. Ihr Ziel ist es, ein logisches System zur Gliederung der Objekte zu erstellen. Prinzipiell kann sie in drei sich teilweise überlappende Bereiche unterteilt werden: die Klassifikation, die Nomenklatur und die Identifikation.

Speziell in der Biologie hat die Taxonomie eine besondere Stellung inne. Auf der einen Seite kann jegliche biologische Forschung zur Etablierung oder Verbesserung eines Klassifizierungssystems beitragen, auf der anderen Seite versorgt die Taxonomie die biologische Forschung mit grundlegenden Informationen über die Identität, Merkmale und mögliche Verwandtschaftsbeziehungen von Organismen (Sivarajan, 1991).

Der Begriff *Systematik* wurde bis in die ersten Hälfte des zwanzigsten Jahrhunderts als Oberbegriff zur Taxonomie verwendet. Da aber keine klare Abgrenzung definiert ist, werden beide Ausdrücke heute als gleichberechtigte Synonyme gebraucht (Sivarajan, 1991).

Mit *Klassifikation* wird ein System von disjunkten Kategorien zur Gruppierung von Objekten oder Organismen bezeichnet. Die einzelnen Kategorien definieren sich über die Eigenschaften und Merkmale, die ihre Elemente besitzen - oder umgekehrt: ein bestimmtes Objekt gehört zu einer bestimmten Gruppe, wenn es genau die für diese Kategorie definierte Merkmale besitzt. Folglich können verschiedene Klassifikationssysteme aufgestellt werden, je nachdem, welche Eigenschaften zur Gruppenbildung herangezogen werden und wie diese gewichtet sind.

In der Biologie wird ein Spezialfall der Klassifikation angewendet: die hierarchische Klassifikation. Ausgehend von einer groben Einteilung mit eher allgemeinen Merkmalen wird diese immer weiter untergliedert, wodurch taxonomische Rangstufen entstehen:

- *Reich*, z. B. Tier- oder Pflanzenreich (*Animalia* bzw. *Plantae*)
- *Stamm*, z. B. innerhalb des Tierreichs: Gliederfüßer (*Arthropoda*)
- *Klasse*, z. B. innerhalb von *Arthropoda*: Insekten (*Insecta*)
- *Ordnung*, z. B. innerhalb von *Insecta*: Schrecken (*Orthoptera*)
- *Familie*, z. B. innerhalb von *Orthoptera*: Grillen (*Gryllidae*)
- *Gattung*, z. B. innerhalb von *Gryllidae*: *Acheta Fabricius, 1775*
- *Art*, z. B. innerhalb von *Acheta*: Hausgrille, *Acheta domesticus (Linnaeus, 1758)*

Gegebenenfalls können diese Stufen noch weiter unterteilt sein; eine Ordnung beispielsweise in Unterordnungen und diese wiederum in Überfamilien. Will man sich auf eine Gruppe beziehen, ohne explizit ihren Rang zu nennen, so kann der Begriff *Taxon* (Plural: *Taxa*) verwendet werden.

Natürlich gibt es nicht nur ein „Standardsystem“, sondern eine Vielzahl unterschiedlicher Klassifikationssysteme, die sich durch die für sie relevanten Merkmale und Eigenschaften sowie deren Gewichtung unterscheiden. Ziel der taxonomischen Forschung in der Biologie ist die Bildung eines Systems, das die natürlichen Verwandtschaftsverhältnisse und die Abstammung aller Organismen wiedergibt. Ob dieses Ziel erreicht werden kann oder ob jede Klassifizierung von Lebewesen immer künstlich, also vom Menschen konstruiert ist, wird kontrovers diskutiert. Auf jeden Fall können neue Forschungsmethoden zu neuen Erkenntnissen über die Verwandtschaft von Organismen und somit zu neuen Klassifizierungssystemen führen.

Die *Nomenklatur* beschäftigt sich mit der Richtigkeit der für die einzelnen Kategorien vergebenen Namen. Charakteristisches Merkmal in der Biologie ist die *binominale Nomenklatur*, die erstmalig von Carl von Linné 1753 in seinem Werk *Species plantarum* verwendet wurde und bis heute gültig ist. Glichen vorher die Artnamen eher beschreibenden Sätzen und Phrasen, so setzen sich heute die Namen von Arten aus dem Namen der Gattung (ohne deren Autorenzitat) und einem artspezifischen Anhängsel, dem *Epitheton*, sowie dem Autorenzitat der Art zusammen. Teilweise besteht unter Biologen die Übereinkunft, nur das Epitheton als Artnamen zu betrachten. Aus diesem Grund sei an dieser Stelle darauf hingewiesen, dass in dieser Arbeit unter dem Begriff „Artname“ der vollständige, zusammengesetzte Name, bestehend aus Gattungsnamen, Epitheton und Autorenzitat, gemeint ist.

Prinzipiell bestehen die biologischen Namen aus lateinischen oder griechischen Wortstämmen – oder sind zumindest durch ihre Endung so klingend. Zum einen ist dies historisch bedingt, denn früher, und auch noch im 18. Jahrhundert, war Latein die Sprache der Wissenschaft und somit hat sich die Regel, Organismengruppen lateinische Namen zu geben, bis heute gehalten. Zum anderen hat dies auch ganz pragmatische Gründe. Ein Name sollte eindeutig mit der von ihm repräsentierten Gruppe verknüpft sein. Namen in einer Landessprache entsprechen dem nicht, denn zum einen gibt es sie nur für wenige, schon länger bekannte Tier- und Pfanzenarten, zum anderen werden sie häufig allein von einer kleinen linguistischen oder ethnischen Gruppe gebraucht. Auch kann es selbst innerhalb einer Region für dieselbe Art mehrere umgangssprachliche Namen geben, oder es kann vorkommen, dass verschiedene Arten unter demselben Namen bekannt sind.

Die Regeln, welche die wissenschaftlichen Namen einhalten müssen, sind für Pflanzen, Pilze und Cyanobakterien im *International Code of Botanical Nomenclature* (ICBN, Greuter et al., 2000) und für die Zoologie im *International Code of Zoological Nomenclature* (ICZN, 2000) festgelegt. Daneben wurden noch Nomenklaturcodes für Bakterien und Viren sowie explizit für Kulturpflanzen erstellt. In diesen Regelwerken wird nicht nur definiert, wie ein wissenschaftlicher Name aufgebaut sein muss (also etwa welche Endung für welchen Rang vorgeschrieben ist), sondern unter anderem auch in welcher Form ein neuer Name publiziert werden muss. Eine solche erste Veröffentlichung, in der der wissenschaftliche Name für ein Taxon festgelegt wird, wird auch *Erstbeschreibung* genannt. Bei botanischen Namen ist der Name des Autors dieser Erstbeschreibung ebenfalls fester Bestandteil des Taxonnamens; in der Zoologie gehört er nicht zwingend dazu. Wenn er aber dem Namen angefügt wird, so muss ebenso das Jahr der Publikation der

Erstbeschreibung zitiert werden.

Eine weitere wichtige Vorschrift besteht darin, dass zu jedem neuen Artnamen mindestens ein Sammlungsbeleg in einer öffentlichen Sammlung als so genannter „Typus“ (auch oft mit Typusbeleg oder Typusexemplar bezeichnet) deklariert und in der Erstbeschreibung genannt werden muss. Anhand der Merkmale dieses Typus wird die Art beschrieben. Dies bedeutet unter anderem, dass jedes Individuum mit denselben Merkmalen wie das Typusexemplar auch zu dieser Art gehört.

Trotz dieser Nomenklaturregeln sind die wissenschaftlichen Namen nicht konstant. So war es beispielsweise Ende des 19., Anfang des 20. Jahrhunderts populär, Exkursionen in die Tropen zu unternehmen, um dort möglichst viele neue Tier- und Pfanzenarten zu entdecken. Aufgrund der Vielzahl neuer Arten und einem vergleichsweise eher langsamen Informationsfluss kam es häufig vor, dass dieselbe Art von verschiedenen Forschern unabhängig voneinander beschrieben wurde. Wird später festgestellt, dass zwei Namen eigentlich dasselbe Taxon beschreiben, so wird gemäß den Nomenklaturregeln der ältere der beiden Namen als der *gültige Name* deklariert. Der andere Name verschwindet jedoch nicht, denn es könnte sein, dass in zwischenzeitlich erschienener Literatur dieser Name verwendet worden ist. Er wird in diesem Fall als *Synonym* des gültigen Namens gehandhabt.[8]

Es gibt weitere Bedingungen, die es erforderlich machen, einen Namen zu einem Synonym zu deklarieren. Wie schon erwähnt, kann die biologische Forschung zu einer Modifizierung des Klassifikationssystems führen, etwa indem festgestellt wird, dass eine Art oder ein Teil einer Art gar nicht zu der Gattung gehört, in die sie ursprünglich gestellt wurde, sondern in eine ganz andere. Wird die Gattungszugehörigkeit einer Art berichtigt, so spricht man von einer *Umkombination*. Aufgrund der binominalen Nomenklatur ändert sich bei einer Umkombination der Name der Art: Der Name der neuen Gattung ersetzt den alten Gattungsnamen, das Epitheton aber wird beibehalten[9]. Um anzuzeigen, dass es sich bei einem Artnamen um eine Umkombination handelt, wird das Autorenzitat in runde Klammern gesetzt und, im Fall der Botanik, der Autor, der die Umkombination veröffentlicht hat, dem Artnamen hinzugefügt. Damit erhält man ebenfalls zwei Namen für dieselbe Art. In der Botanik wird die Umkombination zum gültigen Namen und der ursprüngliche Name zu einem Synonym. In der Zoologie hingegen werden Umkombinationen nicht zu den Synonymen gezählt, sondern getrennt aufgelistet; trotzdem wird auch hier der Name der Umkombination als gültiger Name verwendet.[10] Hier zwei Beispiele aus der Zoologie und der Botanik:

[8] Obwohl der Begriff „Synonym“ impliziert, dass beide Namen nun gleichberechtigt verwendet werden könnten, ist dies in der biologischen Nomenklatur nicht der Fall. Ein als Synonym deklarierter Name darf bzw. sollte nicht mehr benutzt werden, sondern anstelle dessen nur noch der gültige Name.

[9] Gegebenenfalls muss die lateinische Endung des Epithetons an das Geschlecht des Gattungsnamens angepasst werden.

[10] Aus datentechnischer Sicht ist diese Unterscheidung nicht relevant, da Umkombinationen wie Synonyme behandelt werden können und somit nur einen anderen „Synonymietyp“ darstellen.

– Die Art *Anser jubatus Spix, 1825* wurde ursprünglich in der Gattung *Anser Brisson, 1760* beschrieben und später in die Gattung *Neochen Oberholser, 1918* gestellt. Der neue Artname lautet jetzt *Neochen jubatus (Spix, 1825)*.

– *Acacia jupunba Willd.* gehörte anfangs zur Gattung *Acacia Mill.* und wurde dann in die Gattung *Abarema Pittier* gestellt. Dadurch ändert sich der Artname: *Abarema jupunba (Willd.) Britton & Killip*.

Prinzipiell unterscheidet man zwischen zwei Arten von Synonymen: den *nomenklatorischen* und den *taxonomischen Synonymen*. Erstere werden in der Botanik häufig auch *homotypische* und in der Zoologie *objektive* Synonyme genannt, letztere werden als *heterotypische* bzw. *subjektive* Synonyme bezeichnet. Bei nomenklatorischen Synonymen erfolgt die Umbenennung einer Art in der Regel durch eine andere Auffassung über die Gattungszugehörigkeit, also durch eine Umkombination. Dadurch besteht meistens kein Zweifel darüber, welcher Name nun gültig und welcher Synonym ist. Wie oben erwähnt, werden in der Zoologie Umkombinationen nicht zu den Synonymen gezählt, sodass es hier nur sehr wenige „echte“ nomenklatorische Synonyme gibt. Taxonomische Synonyme hingegen stellen eine Meinungsäußerung über die Gleichheit zweier Taxa dar, die auf unterschiedlichen Typusbelegen basieren. Dies bedeutet, dass unterschiedliche Ansichten darüber existieren können, ob die Namen nun synonym sind oder nicht.

Da zumindest in der Zoologie die Art der Synonymie – sei es eine Umkombination, ein nomenklatorisches oder ein taxonomisches Synonym – keinen Einfluss auf die Behandlung der Namen und ihrer Synonyme hat, kann der Synonymietyp als einfaches Attribut der Synonymie verstanden werden.

## 1.3 Motivation

Wie eingangs erwähnt, wurde im EDIS-Projekt beschlossen, die SysTax-Datenbank zur Speicherung und späteren Veröffentlichung von zoologischen Sammlungsdaten im Internet zu verwenden.

Obwohl SysTax zu diesem Zeitpunkt vor allem botanisch ausgerichtet war, war es eines der wenigen Datenbanksysteme, welches nicht nur Sammlungsbelege, sondern auch taxonomische Daten und Synonyme, Literaturdaten (insbesondere Erstbeschreibungen) sowie weitere Informationen, wie beispielsweise Verbreitungsangaben oder Bilder verwalten, zusammenführen und diese im Internet präsentieren konnte. Somit bestand nun die Aufgabe darin, die SysTax-Datenbank an die Bedürfnisse der zoologisch ausgerichteten Nutzer anzupassen.

In Bereichen wie beispielsweise Literatur-, Instituts- und Adressenverwaltung waren keine Änderungen notwendig, da diese von der Fachrichtung unabhängig sind.[11] Auch die

[11] Anpassungen an die Bedürfnisse neuer Nutzer, insbesondere durch Erweiterung des Funktionsumfangs der Datenbank, sind dadurch natürlich nicht ausgeschlossen.

Sammlungsverwaltung selbst bedurfte keiner großer Anpassung, denn Grundinformationen wie etwa Sammler, Sammeldatum, Fundort oder Aufbewahrungsort im Museum sind für botanische und zoologische Sammlungsbelege dieselben; sie unterscheiden sich lediglich in wenigen Details: In der Botanik wird häufig explizit gekennzeichnet, ob bei einem Sammlungsbeleg Pflanzenteile dabei sind, die der Fortpflanzung dienen. Eine solche Information wird in der Zoologie hingegen nicht verwendet, wohl aber wird das Entwicklungsstadium des Tieres (z. B. Larve, Puppe oder adultes Insekt) erfasst.

Nach einer Nutzerbefragung war es nicht notwendig, getrennte Benutzeroberflächen für die botanischen und die zoologischen Sammlungsbelege zu entwickeln, sodass nach einer Erweiterung der botanischen Sammlungsverwaltung, die sowohl Datenbanktabellen als auch Masken betraf, diese für die zoologischen Belege mitverwendet werden konnte.

### 1.3.1 Taxonomie

Im Bereich der Taxonomie und Synonymie waren umfangreichere Anpassungen notwendig. Zwar war schon Mitte der neunziger Jahre die zoologische Taxonomie Bestandteil von SysTax, wurde aber wegen mangelnder Nachfrage nicht weiterentwickelt, weshalb die Notwendigkeit einer Neukonzipierung bestand. Zu Beginn des Projektes stellte sich die Frage, ob und inwieweit die botanische Taxonomie- und Synonymieverwaltung für die Zoologie übernommen werden könne. Folgende Gründe sprechen für eine komplette Trennung der zoologischen Taxonomie von der botanischen, auf Tabellen- und auf Maskenebene:

Häufig soll in den Ausgaben, unabhängig ob in den Masken oder im Internet, zwischen zoologischen und botanischen Daten unterschieden werden. Bei gemeinsamer Verwendung der Taxonomietabellen müsste entweder jeder Name eine Kennzeichnung bezüglich seiner Zugehörigkeit zum Tier- oder Pflanzenreich erhalten, oder man müsste eventuell lange Abfragezeiten in Kauf nehmen, um diese Information zu ermitteln, denn aufgrund des hierarchischen Aufbaus der entsprechenden Tabelle müsste bei jeder Abfrage das jeweilige Wurzelelement[12] eines gefundenen Namens ermittelt werden.

Aufgrund der Differenzen zwischen dem botanischen und dem zoologischen Nomenklaturcode unterscheidet sich die Logik und damit wiederum der hierarchische Aufbau der botanischen Taxonomie und Synonymie von dem der zoologischen. Zum Beispiel werden in der Botanik Namen von Unterarten aus dem Namen der Gattung, dem Epitheton gefolgt vom Kürzel „ssp.“ sowie dem Epitheton der Unterart gebildet. In der Zoologie hingegen wird kein solches Kürzel verwendet, wodurch Trinomen entstehen. Auch die Anzahl und Bezeichnung der einzelnen Rangstufen sind verschieden. So benutzt die Zoologie in den höheren Rangstufen mehr Unterstufen, in den unteren Rangstufen hingegen weniger Unterstufen als die Botanik.

Auch in der Synonymie gibt es Unterschiede: In der Zoologie wird zwischen den so genannten objektiven Synonymen und den Umkombinationen unterschieden, während die

[12] hier: entweder Tier oder Pflanze

botanische Synonymie beides unter dem Begriff homotypisches Synonym zusammenfasst. Dies schließt die Verwendung derselben Benutzeroberfläche für die botanische und die zoologische Arbeitsweise aus, denn in diesem Fall müssten entweder botanisch arbeitende Nutzer zwischen homotypischen Synonymen und Umkombinationen unterscheiden oder zoologischen Nutzern würde diese Unterscheidungsmöglichkeit fehlen. Beides könnte sich negativ auf die Akzeptanz von SysTax auswirken. Um beiden Fachbereichen gerecht zu werden, sollten also zweierlei Maskengruppen geschaffen werden.

In der Botanik gab es schon zu Projektbeginn die Möglichkeit, alternative Taxonomien und Synonymien zu speichern, jedoch war das Prinzip der Potential Taxa[13] nicht vollständig implementiert. Dies wurde zwar auch von der Zoologie nicht gefordert, aber die Güte einer biologischen Datenbank wurde schon zu diesem Zeitpunkt an der Verwendbarkeit von Potential Taxa gemessen (Hoppe, persönl. Mitteilung).

Somit wurde beschlossen, für die zoologische Taxonomie und Synonymie ein neues Datenbankenmodell zu entwickeln mit der Vorgabe, dass es das Prinzip der Potential Taxa unterstützen und trotzdem eine Implementierung der darauf basierenden Benutzeroberfläche in Design und Funktionalität im Sinne einer „Corporate Identity“ an die vorhandene botanische Maske angepasst sein sollte.

Ein Ziel der vorliegenden Arbeit war es, dieses Modell und seine Umsetzung in Datenbanktabellen zu entwickeln. Kapitel 2 behandelt die Problematik der verschiedenen Auffassungen eines wissenschaftlichen Namens und stellt das Prinzip der Potential Taxa als mögliche Lösung vor. Kapitel 3 beschäftigt sich zunächst mit den Prinzipien der Modellierung allgemein, zudem werden ein schon existierendes Datenmodell zur Umsetzung der Potential Taxa sowie das für SysTax entworfene Modell werden vorgestellt. Im darauf folgenden Kapitel 4 wird die Realisierung dieses Modell in Datenbanktabellen gezeigt.

### 1.3.2 Multimedia

Eines der erklärten Ziele des EDIS-Projektes ist es, multimediale Daten über die in den Einzelprojekten digitalisierten Sammlungsobjekte der Öffentlichkeit zugänglich zu machen. Durch die Beschränkung auf die Typusbelege der beteiligten Museen konnte die Anzahl der Bilder auf etwa 30000 geschätzt werden, die Schätzung der Anzahl der Tondokumente betrug ca. 8000.

SysTax hatte schon zu Projektbeginn die Möglichkeit integriert, Bilder zu botanischen Taxa zu speichern, nicht aber zu Sammlungsbelegen. Ebenso wenig war die Ablage von Tondateien vorgesehen. Zudem werden in den EDIS-Teilprojekten wesentlich mehr Metadaten zu den einzelnen Multimediaobjekten erfasst, als in SysTax zu diesem Zeitpunkt gespeichert werden konnten. Daraus resultierte zumindest die Notwendigkeit, die bestehende Bildverwaltung zu erweitern.

[13] siehe Kapitel 2

Zu Beginn des EDIS-Projekts wurde die SysTax-Datenbank von der Oracle-Version 7.3.4 auf die Version 8.1.5 portiert. In den Version vor Oracle 8 konnten binäre Daten nur in Tabellenspalten des Datentyps LONG abgelegt werden. Die Verwendung dieses Datentyps ist jedoch umständlich, da er von Seiten Oracles nur rudimentär unterstützt und Daten nur mit gewissem Aufwand manipuliert werden können. Mit Oracle Version 8 wurde der Datentyp BLOB (Binary Large Object) eingeführt, der den Typ LONG ersetzen sollte. Zugleich standen wesentlich mehr Funktionen zur Manipulation von BLOB-Daten als nur Lesen und Schreiben zur Verfügung, wie etwa *append*, *copy*, *compare*, oder *get_size*. LONG-Spalten hingegen werden nur noch aus Gründen der Abwärtskompatibilität unterstützt und sollten durch BLOBs ersetzt werden.

Dies wurde zum Anlass genommen, die Verwaltung von Multimediainformationen neu zu konzipieren. In Kapitel 5 wird ein entsprechendes Datenmodell vorgestellt und die verschiedenen Möglichkeiten der physikalischen Speicherung dieser Daten diskutiert.

### 1.3.3 Schnittstellen

In den meisten an EDIS beteiligten Institutionen wurden schon vor Projektbeginn Literatur-, Sammlungs- und taxonomische Daten mit Hilfe von kleinen Access-Datenbanken oder nicht-normalisierten Excel-Tabellen elektronisch erfasst. Diese Datenbestände sollten auf Wunsch der Nutzer beibehalten und ergänzt werden. Anderen Projektpartnern war es technisch (beispielsweise wegen fehlender Internetanbindung) nicht möglich, die SysTax-Clientsoftware zu verwenden, um damit ihre Daten zu erfassen.

Dies machte es erforderlich, eine vom Internet unabhängige Form der Dateneingabe in SysTax zu schaffen. Dazu wurde ein entsprechendes Datenaustauschformat erstellt und auf der Seite von SysTax entsprechende Routinen implementiert, die diese Daten in die Datenbankstruktur importieren. Ein erster Versuch, diese Schnittstelle basierend auf einem XML-Schema zu formulieren, scheiterte an der mangelnden Akzeptanz seitens der Datenlieferanten. Daraufhin wurde entschieden, tabellarische Textdateien zum Datenaustausch zu verwenden.

Die Notwendigkeit zur Erstellung einer eigenen Schnittstellenbeschreibung ergab sich daraus, dass miteinander verknüpfte Daten aus extrem unterschiedlichen Bereichen wie Sammlungsverwaltung, Literatur und Taxonomie darüber transportiert werden sollten. Natürlich gab und gibt es bereits bestehende Datenaustauschformate, die aber meist auf einen einzigen Bereich spezialisiert und für eine Verknüpfung von Daten aus unterschiedlichen Bereichen nicht konzipiert sind. Häufig definieren sie auch eine eigene, nicht auf XML oder tabellarische Textdateien basierende Datenstruktur. Als Beispiel hierfür sei das Format *Herbarium Information Standards and Protocols for Interchange of Data*[14] erwähnt, das dem Austausch speziell von botanischen Sammlungsbelegen dient.

Zur Erstellung der SysTax-Schnittstellen wurde mit Hilfe des „Verlags für interaktive Medien“, zuständig für die EDV-Koordination innerhalb von EDIS, zunächst analysiert,

[14] http://plantnet.rbgsyd.nsw.gov.au/HISCOM/HISPID/HISPID3/hispidright.html (Mai 2005)

welche Arten von Daten die einzelnen Projekte liefern können. Aus diesen Informationen konnte dann ein Datenaustauschformat erstellt werden, das in der Lage war, sämtliche in den Teilprojekten erhobenen Daten nach SysTax zu transportieren. Die einfache Struktur als tabellarische Textdatei ermöglichte es allen Teilprojekten, ihre Daten selbstständig in das vorgegebene Format zu konvertieren.

In Kapitel 6 werden die Vor- und Nachteile von XML- und tabellarischen Textdateien als Datenaustauschformat diskutiert, insbesondere in Hinblick auf die Darstellung verknüpfter sowie hierarchisch gegliederter Daten. Des Weiteren werden die Schwierigkeiten und möglichen Lösungen bei der Vergabe von schnittstelleninternen Datensatzschlüsseln beschrieben. Auf den konkreten Inhalt des Datenaustauschformats soll nicht näher eingegangen werden, da dieser laufend den Anforderungen der Datenlieferanten angepasst wird.

# 2 Potential Taxa

## 2.1 Verschiedene Sichtweisen

Viele botanische und zoologische Datenbanksysteme gehen von einer vorgegebenen „Standardtaxonomie“ aus. Für einige Aufgabenstellungen, etwa der reinen Sammlungsverwaltung, mag dies zwar ausreichend sein, nicht jedoch für wissenschaftliches Arbeiten auf dem Gebiet der Systematik. Hierfür ist es notwendig, mehrere Sichtweisen über die hierarchische Zuordnung der einzelnen Taxa und mehrere Synonymiebäume gleichzeitig zu verwalten (Berendsohn, 1995). SysTax ist eines der ersten Datenbanksysteme, mit dem es möglich ist, mehrere alternative Taxonomien und Synonymien darzustellen. Realisiert wird dies durch die konsequente Trennung der biologischen Namen von ihrer systematischen Klassifikation.

Obwohl die botanischen wie auch die zoologischen Nomenklaturregeln eine eindeutige Zuordnung der einzelnen Taxa zu ihrem jeweils nächsthöheren Taxon im System vorschreiben, treten in der Praxis immer wieder unterschiedliche Ansichten und Meinungen über ein Taxon und/oder dessen Synonyme auf. Diese Problematik ergibt sich sogar unmittelbar aus dem Code selbst: Hier ist zwar festgelegt, dass die Umschreibung eines Taxon immer den Typus beinhalten muss, nicht aber, dass die Umschreibung selbst konstant zu bleiben hat. Anders ausgedrückt, der Name eines Taxons ändert sich nicht, wenn seine Umschreibung geändert wird, solange sein Typus in dieser Umschreibung enthalten ist:

*„An alteration of the diagnostic characters or of the circumscription of a taxon without the exclusion of the type does not warrant a change of the author citation of the name of the taxon.“* (Greuter et al., 2000, Artikel 47.1)

*„No matter how the boundaries of a taxonomic taxon may vary in the opinion of zoologists the valid name of such a taxon is determined [Art. 23.3] from the name-bearing type(s) considered to belong within those boundaries.“* (ICZN, 2000, Artikel 61.1.1)

Unter der Umschreibung versteht man bei Arten die Gesamtheit aller untergeordneten, infraspezifischen Taxa und die Gesamtheit aller Belegexemplare, welche zur Definition dieser Art beigetragen haben, und bei höheren Taxa die Gesamtheit aller unmittelbar untergeordneten Taxa. Zusätzlich kann noch eine Reihe von Merkmalen und deren Ausprägung zur Umschreibung hinzugenommen werden, welche das Taxon näher charakterisieren.

Die Problematik besteht darin, dass Taxonomen unterschiedlicher Meinung bezüglich der Umschreibung eines Taxons sein können. Meist entstehen solche Unstimmigkeiten

durch die fortschreitende Erkenntnis in der taxonomischen Forschung, sodass die jeweils alte Auffassung in die Synonymie verwiesen wird. Trotzdem existieren solche Meinungsverschiedenheiten manchmal parallel nebeneinander, denn eine allgemeingültige und verbindliche Auflistung aller Taxa und deren Systematik gibt es nicht.

Ein bekanntes Beispiel dieser Problematik aus der Botanik ist die Familie der Liliengewächse, *Liliaceae Juss.* Hier sind sich die Wissenschaftler nicht einig über ihre Umschreibung. Einige sehen *Liliaceae* in einem engeren Zusammenhang und betrachten z. B. *Alliaceae*, *Amaryllidaceae*, *Aphyllanthaceae*, *Hypoxidaceae* usw. als eigenständige Familien. Ein andere Gruppe von Taxonomen fasst all diese Taxa zu einer einzigen, großen Familie *Liliaceae* zusammen (Mabberley, 1997, S. 410f). Deshalb ist nicht immer klar, was genau gemeint ist, wenn jemand die Familie *Liliaceae* erwähnt.

Als mögliche Lösung wird in Berendsohn (1995, 1997) vorgeschlagen, ein Taxon immer zusammen mit der Referenz (Literaturzitat) auf seine Umschreibung anzugeben. Dies kann dadurch geschehen, dass die Referenz zusammen mit der lateinischen Präposition „secundum“ (abgekürzt „sec.“) an den Taxonnamen angehängt wird. Berendsohn nennt dieses Konstrukt aus Taxonnamen und Referenzwerk „Potential Taxon“, eine Bezeichnung, die sich vor allem in englischsprachigen Publikationen durchgesetzt hat. Andere Quellen wie z. B. Koperski et al. (2000) verwenden hierfür den Begriff Taxonym[1]. Durch die Verwendung von Taxonymen anstelle von Taxonnamen können nicht nur verschiedene Sichtweisen eines Taxons kenntlich gemacht werden, sondern auch mit Hilfe der so genannten Konzeptsynonyme die Unterschiede zwischen den einzelnen Konzepten genauer charakterisiert werden[2].

Ein Beispiel aus dem Bereich der Moose soll diese Theorie verdeutlichen (Koperski et al. 2000): Abbildung 2.1 zeigt die unterschiedlichen Auffassungen verschiedener Autoren darüber, was sie unter dem Artnamen *Pottia starckeana* verstehen. Dabei stellt jede Reihe das entsprechende Konzept des jeweiligen Autors dar, die Rechtecke symbolisieren die Umschreibung des Taxons. Sich entsprechende Taxa sind senkrecht übereinander angeordnet. In allen sechs Referenzwerken findet sich eine Art namens *Pottia starckeana*, wobei aber die Schreibweise des Epithetons leicht variiert (*starckeana* und *starkeana*).

Eine taxonomische Übereinstimmung liegt hier bei *Pottia starckeana* sec. Koperski & al. (2000), *Pottia starckeana* sec. Frahm & Frey (1992) und *Pottia starckeana* sec. Corley & al. (1981/1991) vor. Die drei anderen Werke, Ludwig & al. (1996), Chamberlain in Smith (1980) und Mönkemeyer (1927) sehen *Pottia starckeana* in einer umfangreicheren Umschreibung, da hier Taxa eingeschlossen sind, die z. B. bei Koperski & al. (2000) als eigenständige Arten akzeptiert sind. Ähnlich sieht es auch bei *Pottia mutica* aus: *Pottia mutica* sec. Koperski & al. (2000) entspricht z. B. *Pottia starckeana var. brachyoda* sec. Ludwig & al. (1996) und ist in der Umschreibung von *Pottia davalliana* sec. Corley & al. (1981/1991) enthalten.

Ohne die Angabe einer Referenz ist es also nur schwer möglich, einem konkreten Taxon eine Information zuzuordnen. Angenommen, es liege lediglich die Information vor, dass

[1] Im Folgenden werden beide Ausdrücke als gleichwertige Synonyme verwendet.

[2] siehe hierzu auch Abschnitt 2.2

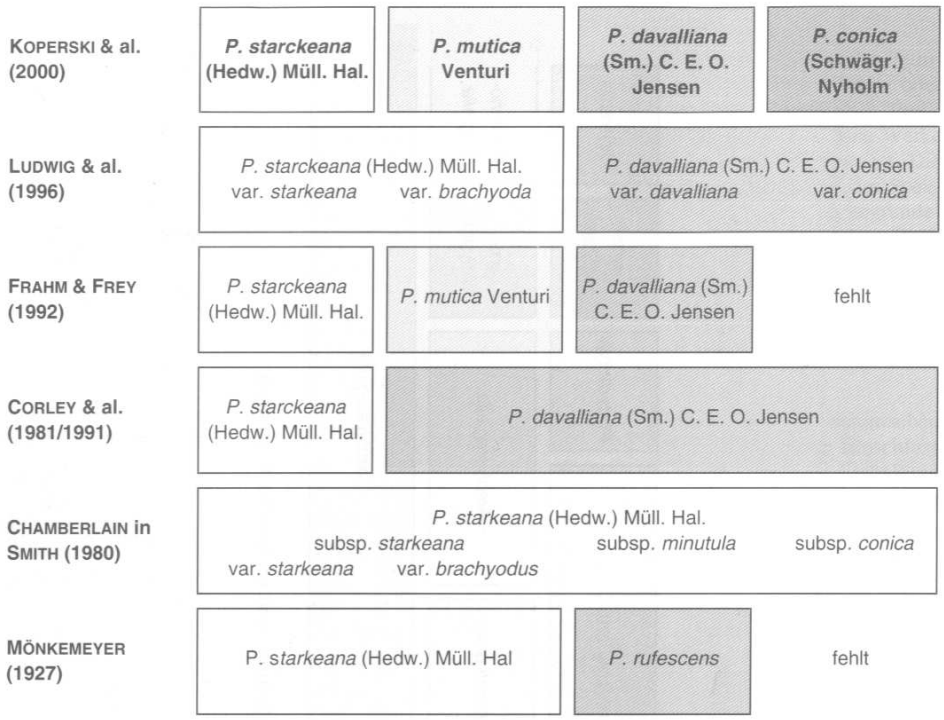


Abbildung 2.1: Vergleich der taxonomischen Konzepte am Beispiel des Verwandtschaftskreises von *Pottia starckeana (Hedw.) Müll. Hal.* (entnommen aus Koperski et al. (2000), S. 14).

*Pottia davalliana* eine bedrohte Art sei und gesetzlich geschützt wäre, ohne die Angabe, dass hier auf das Referenzwerk von CORLEY & al. (1981/91) Bezug genommen werde. Jemand, der nun ein Vertreter dieser Art in der Natur anhand von LUDWIG & al. (1996) als *Pottia mutica* identifiziert, kann somit nicht wissen, dass es sich hierbei ebenfalls um ein bedrohtes Individuum handelt. Anders ausgedrückt, eine Information, die für *Pottia davalliana* sec. CORLEY & al. (1981/1991) gültig ist, besitzt ebenfalls Gültigkeit für *Pottia muttica* sec. KOPERSKI & al. (2000), *Pottia davalliana* sec. KOPERSKI & al. (2000) sowie *Pottia conica* sec. KOPERSKI & al. (2000).

## 2.2 Konzeptsynonyme

Durch diese im vorhergehenden Abschnitt vorgestellten Potential Taxa steht nun ein weiterer Typ einer Synonymiebeziehung zur Verfügung: das so genannten Konzeptsynonym.

Herkömmliche Synonymiebeziehungen beinhalten die Zuordnung von ungültigen Namen zu ihren jeweils akzeptierten, gültigen Namen, wie schon in Abschnitt 1.2 näher dargestellt wurde. Aber auch hier ist es möglich, durch die Verwendung bestimmter

Zusätze, wie z. B. „p. p.“ (für „pro parte“) oder „sensu“, gefolgt von eine Literaturreferenz, konzeptionelle Unterschiede darzustellen. Meist wird jedoch nicht deutlich, worauf sich solche Zusätze beziehen.

Erst die Konzeptsynonymie macht es möglich, verschiedene Umschreibungen eines Taxons miteinander zu vergleichen. Hierfür werden entsprechende Taxonyme unterschiedlicher Referenzwerke in Beziehung zueinander gestellt und somit die einzelnen taxonomischen Konzepte miteinander verglichen (daher der Begriff Konzeptsynonym). Die Namen der in der Konzeptsynonymie involvierten Taxonyme müssen dabei nicht zwingend identisch sein, wie obiges Beispiel zeigt, ebenso wenig wie sie auf derselben systematischen Stufe stehen müssen. So kann es durchaus vorkommen, dass z. B. die Umschreibung einer Art in Referenz A identisch mit der einer Unterart in Referenz B ist.

Die Zuweisung eines Status zu einem Konzeptsynonym erlaubt die nähere Charakterisierung des Grades der Übereinstimmung. Dieser Status kann folgende Werte annehmen: *Kongruenz*, *Inklusion*, *Pro-parte-Inklusion*, *Interferenz* und *Exklusion* (Koperski et al. 2000).

Am häufigsten tritt sicherlich die *Kongruenz* auf. Hierbei stimmen die Umschreibungen beider Taxonyme vollständig überein. Dieser Status wird mit dem Symbol $\widehat{=}$ angegeben:

„Taxonym A $\widehat{=}$ Taxonym B“.

Sind zwei Taxonyme nicht kongruent, so können sich ihre Umschreibungen auf verschiedene Arten überschneiden. Bei der *Inklusion* ist die Umschreibung eines Taxonyms größer als die eines anderen und schließt diese vollständig ein:

„Taxonym A $>$ Taxonym B“.

Der umgekehrte Fall, die Umschreibung von Taxonym A wird vollständig von der Umschreibung des Taxonym B eingeschlossen, wird als *Pro-parte-Inklusion* bezeichnet:

„Taxonym A $<$ Taxonym B“.

Diese beiden Status sind symmetrisch: wenn „Taxonym A $>$ Taxonym B“ ist, dann gilt auch „Taxonym B $<$ Taxonym A“.

Ein weiterer Status der Konzeptsynonymie ist die *Interferenz*:

„Taxonym A $\lessgtr$ Taxonym B“.

Sie liegt dann vor, wenn sich die Umschreibungen zweier Taxonyme nur partiell überschneiden. Es kann aber auch vorkommen, dass bei zwei Taxonymen, obwohl vielleicht vom Namen her identisch, keinerlei Übereinstimmung ihrer Umschreibung vorhanden ist. In diesem Fall spricht man von der *Exklusion*:

„Taxonym A $\neq$ Taxonym B“.

Unklarheiten über den Status der Konzeptsynonyme können durch das Anfügen eines Fragezeichens an das entsprechende Symbol zum Ausdruck gebracht werden (z. B. $\leqslant?$ oder $>?$). Des Weiteren ist es möglich, durch die Angabe weiterer Taxa darzustellen, worin die Unterschiede zwischen den Konzeptsynonymen bestehen.

Abbildung 2.2 zeigt die Synonymieliste für die in Abbildung 2.1 vorgestellten Konzepte von *Pottia starckeana*.

**Pottia starckeana (Hedw.) Müll. Hal.**
Syn. Musc. Frond. 1: 547. 1849 <1>
≡ *Weisia starckeana Hedw.*, Sp. Musc. Frond.: 65. 1801
≡ *Microbryum starckeanum R. H. Zander*

| | |
|---|---|
| ≙ *Pottia starckeana (Hedw.) Müll. Hal.* | sec. Corley & al. (1981/1991) |
| ≙ *Pottia starckeana (Hedw.) Müll. Hal.* | sec. Fram & Frey (1992) |
| ≙ *Pottia starckeana var. starckeana* | sec. Ludwig & al. (1996) |
| ≙ *Pottia starkeana var. starkeana* | sec. Mönkemeyer (1927) |
| Autonym ergänzt | |
| ≙ *Pottia starkeana var. starkeana* | sec. Smith (1980) |
| Eigentlich als typische Varietät der typischen Unterart (*P. starkeana subsp. starkeana var. starkeana*). | |
| < *Pottia starckeana (Hedw.) Müll. Hal.* | sec. Ludwig & al. (1996) |
| incl. *P. mutica* (*P. starckeana var. brachyoda*) | |
| < *Pottia starkeana (Hedw.) Müll. Hal.* | sec. Mönkemeyer (1927) |
| incl. *P. mutica* (*P. starkeana var. brachypoda*) | |
| < *Pottia starkeana (Hedw.) Müll. Hal.* | sec. Smith (1980) |
| incl. *P. mutica* (*P. starkeana subsp. starkeana var. mutica*), *P. davalliana* (*P. starkeana subsp. minutula*) und *P. conica* (*P. starkeana subsp. conica*) | |
| < *Pottia starkeana subsp. starkeana* | sec. Smith (1980) |
| incl. *P. mutica* (*P. starkeana subsp. starckeana var. brachyodus*) | |

Abbildung 2.2: Der Artkomplex von *Pottia starckeana (Hedw.) Müll. Hal.* (zitiert aus Koperski et al. (2000), S. 19).

## 2.3 Potential Taxon = Potential Chaos?

Die Vorteile der Verwendung von Potential Taxa liegen auf der Hand: sie bietet eine einfache Lösung zur parallelen Speicherung von alternativen Taxonomien und alternativen Synonymien. Nicht taxonomische Informationen können dabei einem konkreten Namenskonzept zugeordnet werden. Eine Folge davon ist die Möglichkeit, taxonomische Daten verschiedener Quellen (z. B. Institute, Sammlungen, Museen) getrennt voneinander verwalten zu können, falls dies gewünscht wird, indem für jede Quelle ein eigenes Potential Taxon generiert wird.

Dennoch besitzt das Konzept der Potential Taxa auch Schwachstellen. Häufig werden bei nicht taxonomischen Informationen, wie etwa Verbreitungsangaben, Bestimmungen von

Sammlungsbelegen oder Abbildungen, nur die Taxonnamen angegeben, jedoch keinerlei Aussage darüber gemacht, auf welche Konzepte sich die Namen beziehen.

Soll nun für jede Nennung eines Taxons ein eigenes Potential Taxon erstellt werden? Dann würden möglicherweise viele Potential Taxa eines Namens existieren, über deren taxonomisches Konzept keinerlei Aussage gemacht wird, sondern denen lediglich einige wenige Informationen zugeordnet sind, was sicherlich nicht sinnvoll wäre.

Wird hingegen auf die Erstellung mehrerer Potential Taxa bei der Nennung des jeweiligen Namens in der Literatur verzichtet, so stellt sich die Frage, welchem bereits existierenden Potential Taxon eine Information zugeordnet werden soll. Berendsohn[3] macht hierüber keine Aussagen. In vielen Fällen können solche Daten an das Potential Taxon der Erstbeschreibung[4] des Namens gehängt werden, da sich viele Informationen auch auf diese beziehen.

Während der Entwicklung von SysTax hat sich gezeigt, dass die Bereitstellung eines *neutralen Potential Taxons* mit einer neutralen, undefinierten Quelle („sec. undefined“) eine akzeptable Lösung für dieses Problem bietet: Existiert zu jedem Namen ein neutrales Potential Taxon, so können alle Informationen ohne explizite Konzeptangabe, vor allem die Bestimmung von Sammlungsmaterial oder Freilandfotos, diesem zugeordnet werden.

Ein weiteres Problem stellt sich, wenn zu einem Namen mehrere Potential Taxa mit unterschiedlichen taxonomischen Vorgängern existieren: Wenn nur eine Hierarchie dargestellt werden kann, etwa bei Ausgaben im Internet, welche soll dann bevorzugt werden? Berendsohn schlägt hierzu die von ihm so benannten *preferred taxon views* vor[5]. Diesen liegt eine Präferenzordnung auf der Menge der Quellen zugrunde. Stehen nun mehrere Potential Taxa eines Namens zur Auswahl, so wird stets das mit der gemäß dieser Ordnung größten Gewichtung bevorzugt. Selbstverständlich könnten für verschiedene Ausgaben, Benutzer oder Benutzergruppen unterschiedliche Präferenzordnungen definiert werden.

Prinzipiell erscheint diese Lösung zunächst praktikabel; während des EDIS-Projektes hat sich jedoch gezeigt, dass sich dieser Ansatz für SysTax nur schwer umsetzen lässt. Nicht alle Datenlieferanten erfassen ihre taxonomischen Daten gemäß einer Literaturquelle oder geben bei einer Datenlieferung die Literaturquelle ihrer Taxonomie an. Des Weiteren kommt es relativ selten vor, dass ein Datenlieferant mehr als eine taxonomische Zuordnung für ein bestimmtes Taxon liefert und somit eine Gewichtung der Quellen für jeden einzelnen Datenlieferanten nicht notwendig und auch nicht sinnvoll ist. In SysTax werden die Taxa der verschiedenen Lieferanten zusammengeführt. Hierbei kann es mitunter vorkommen, dass dasselbe Taxon von unterschiedlichen Datenlieferanten verschiedenen taxonomischen Vorgängern zugeordnet ist. Aber eine Gewichtung der auf

[3] vgl. Berendsohn 1995 und 1997

[4] Mit „Erstbeschreibung“ ist hier die erste namentliche Erwähnung des Taxon gemeint; so ist z. B. die „Erstbeschreibung“ einer Umkombination (eine Art wird zu einer anderen Gattung gestellt) die Literaturangabe dieser Revision.

[5] s. Berendsohn (1997), S. 302f.

diese Art und Weise zusammengeführten Quellen und damit eine Gewichtung der Datenlieferanten selbst, kann von Seiten des SysTax-Projekts nicht durchgeführt werden. Eine solche Präferenzordnung müsste nach jedem Import von Taxa durch einen von den EDIS-Projektpartnern ernannten Experten erstellt und laufend gepflegt werden.

Das Konzept des neutralen Potential Taxons kann für diese Problematik als Kompromisslösung angesehen werden. Unter der Voraussetzung, dass zu jedem Namen ein neutrales Potential Taxon existieren muss und jedes neutrale Potential Taxon als Vorgänger ebenfalls ein neutrales Potential Taxon besitzt, kann die dadurch gegebene Hierarchie als „Standardhierarchie“ bei Ausgaben, insbesondere im Internet, verwendet werden. Ein Streitpunkt war die Gestaltung dieser Hierarchie. Auch hierfür wurde ein Kompromiss gefunden: Wird ein neuer Name importiert, so ist der Vorgänger seines neutralen Potential Taxons identisch mit dem in diesem Import angegeben Vorgänger des Namens. Später kann dieser Vorgänger ebenfalls neueren taxonomischen Erkenntnissen angepasst werden.

Durch das Prinzip des neutralen Potential Taxons bietet SysTax sowohl die Vorteile einer Datenbank, die lediglich eine Standardtaxonomie speichert, als auch die einer reinen Potential-Taxa-Datenbank. Damit steht insbesondere sammlungsorientiert arbeitenden Nutzern eine einfache Taxonomieverwaltung zur Verfügung, ohne dass sie sich um verschiedene taxonomische Konzepte kümmern müssen. Sollen hingegen unterschiedliche taxonomische Auffassungen als Ergebnis umfangreicher Literaturrecherchen gespeichert werden, ist dies mit dem Prinzip der Potential Taxa ebenfalls möglich.

# 3 Datenmodelle zu Potential Taxa

## 3.1 Modellierungsarten

Um ein real existierendes Problem in Form einer Datenbank oder, ganz allgemein, in Form eines Programms darzustellen, bedient man sich eines Datenmodells. Unter einem Datenmodell versteht man ein durch Abstraktion erstelltes Abbild der Realität, ohne dass dabei näher auf die Technologie bzw. physische Speicherung eingegangen wird, mit der dieses Modell später umgesetzt werden soll. Datenmodelle können unterschieden werden in *konzeptionelle* Modelle, die sich eher an der Realität orientieren, und *physische* Modelle, die sich eher mit der technischen Umsetzung befassen.

Laut Vossen (1994) stehen bei der Modellbildung folgende Mechanismen der Abstraktion zur Verfügung: Klassifikation, Aggregation, Verallgemeinerung und Spezialisierung.

Die Klassifikation entspricht einer Mengenbildung; real existierende Objekte mit gemeinsamen Eigenschaften werden zu einer Menge bzw. Klasse zusammengefasst. Welche Klasse aus welchen Objekten gebildet wird, hängt von der konkreten Problemstellung ab. Häufig sind mehrere Einteilungen plausibel, wobei die Frage nach der „richtigen“ Einteilung nicht pauschal beantwortet werden kann.

Bei einer Aggregation werden neue Klassen aus schon bestehenden Klassen gebildet, welche die Komponenten der neuen Klassen darstellen. Dies könnte beispielsweise der Fall sein, wenn die Klasse *Flugzeug* aus den schon bestehenden Klassen *Flügel*, *Rumpf*, etc. definiert wird.

Durch Verallgemeinerung können Teilmengenbeziehungen zwischen Elementen verschiedener Klassen festgelegt werden. Die Klasse *Flugzeug* etwa wäre eine solche Verallgemeinerung der Klasse *Segelflugzeug*. Unter einer Spezialisierung hingegen versteht man nicht nur eine Umkehrung der Sichtweise (Klasse *Segelflugzeug* als Spezialfall der Klasse *Flugzeug*), sondern meistens eine disjunkte Zerlegung einer Klasse in Unterklassen.

Zur Beschreibung von Modellen steht eine Vielzahl von Möglichkeiten zur Verfügung, die häufig für spezifische Anwendungsgebiete geeignet sind. In vielen Fällen wird eine grafische Visualisierung bevorzugt, da diese besonders gut geeignet ist, einen schnellen Überblick über das modellierte System zu geben.

### 3.1.1 Entity-Relationship

Speziell zur Darstellung von Datenbankmodellen ist das *Entity-Relationship-Model* weit verbreitet. Darin werden die im Modellbildungsprozess erstellten Klassen (Entitäten)

und ihre Beziehungen zueinander in einem *Entity-Relationship-Diagramm* grafisch dargestellt. Zusätzlich wird die Kardinalität der einzelnen Beziehungen angegeben, welche die Anzahl der an einer Beziehung beteiligten Objekte festlegt. Das ER-Diagramm erfüllt dabei zwei Aufgaben: Während der Konzeptionsphase einer Anwendung ermöglicht es die Kommunikation zwischen Entwickler und Anwender, wobei ausschließlich die Sachlogik, nicht aber die technische Umsetzung im Vordergrund steht. In der Implementierungsphase stellt es dann die Basis für das Design der Datenbank dar.

Realisiert werden ER-Diagramme in relationalen Datenbanken mittels der *Structured Query Language* (kurz *SQL*), einer Standardsprache, mit der Datenbestände in Datenbanken manipuliert und abgefragt, aber auch Datenstrukturen (Tabellen, Ansichten etc.) erstellt und bearbeitet werden können. Spezielle Werkzeuge, so genannte *CASE-Tools*[1], stehen zur Verfügung, um aus einem vorgegebenen ER-Diagramm die entsprechenden SQL-Anweisungen automatisch zu generieren oder, umgekehrt, um aus einer bestehenden relationalen Datenbank das entsprechende ER-Diagramm zu erstellen.

### 3.1.2 Unified Modeling Language

Während der Entity-Relationship-Ansatz vor allem bei der Datenbankentwicklung zur Anwendung kommt, ist die *Unified Modeling Language*, abgekürzt mit *UML*, *„. . . a family of graphical notations, backed by single meta-model, that help in describing and designing software systems, particularly software systems built using the object-oriented (OO) style.“* (Fowler, 2004, S. 1). Als Modellierungssprache definiert sie eine Notation zur Konstruktion, Visualisierung und Dokumentation von Softwaresystemen. In der aktuellen Version 2.0 sind dreizehn Diagrammtypen bekannt (Fowler, 2004, S. 11, alphabetisch sortiert):

- Aktivitätsdiagramm (activity): prozedurales und paralleles Verhalten, Geschäftsprozesse, „Work Flow“
- Anwendungsfalldiagramm (use case): Benutzerinteraktion
- Interaktionsübersichtsdiagramm (interaction overview): Mischung aus Sequenz- und Aktivitätsdiagramm
- Klassendiagramm (class): Klassen, Eigenschaften und ihre Beziehungen
- Kommunikationsdiagramm (communication): Interaktion (vor allem Datenfluss) von Objekten, mit Betonung auf die Verknüpfung
- Komponentendiagramm (component): Struktur und Verknüpfungen von Komponenten

[1] Die Abkürzung *CASE* steht für *Computer Aided Software Engineering*, also die computergestützte Softwareentwicklung. Entsprechende Programme können dem Entwickler bei der Planung, Modellierung, Implementierung und Dokumentation unterstützen.

- Kompositionsstrukturdiagramm (composite structure): hierarchische Zerlegung von komplexen Klassen zur Laufzeit
- Objektdiagramm (object): Instanzen von Klassen zu einem bestimmten Zeitpunkt
- Paketdiagramm (package): hierarchische Struktur (Gruppierung von Klassen zu Paketen) zur Kompilierungszeit
- Sequenzdiagramm (sequence): Interaktion von Objekten, mit Betonung auf die Reihenfolge
- Verteilungsdiagramm (deployment): physikalischer Aufbau des Systems
- Zeitverlaufsdiagramm (timing): Interaktion von Objekten, mit Betonung auf die genaue zeitliche Abfolge
- Zustandsdiagramm (state machine): wie Ereignisse den Zustand eines Objektes ändern

Auch wenn die meisten Anwender UML als eine Ansammlung von Diagrammtypen sehen, der Kern der Unified Modeling Language ist ihr Metamodell, ein Klassendiagramm, welches die Konzepte der Sprache definiert. Ob ein Anwender mit dem Metamodell in Berührung kommt, hängt davon ab, wofür er UML einsetzen will. Dient es lediglich zur Dokumentation eines Softwaresystems, sind Kenntnisse über das Metamodell nicht notwendig. Selbst die strikte Einhaltung der UML-Spezifikationen ist in diesem Fall nicht erforderlich, da neue Elemente oder Elemente aus anderen Diagrammtypen in ein eigenes Diagramm integriert werden können. Soll hingegen ein UML-Modell mittels CASE-Tools in einen Programmcode umgewandelt werden, oder gar UML als Programmiersprache eingesetzt werden, wobei das Modell direkt in den ausführbaren Code kompiliert wird, ist die Einhaltung des UML-Standards sowie die Auseinandersetzung mit dem Metamodell unerlässlich, da dieses die abstrakte Syntax der Sprache definiert.

Die wohl am häufigsten angewandte Diagrammart ist das Klassendiagramm. Zugleich entspricht dieser Diagrammtyp auch am ehesten einem Entity-Relationship-Diagramm, da es die Typen von Objekten, die den Entitäten entsprechen und ihre vielfältigen Beziehungen zueinander beschreibt. Auch die Attribute von Klassen bzw. Entitäten können in einem Klassendiagramm dargestellt werden, ebenso wie die Kardinalität einer Relation, die hier allerdings „Multiplizität“ genannt wird. Darüber hinaus kennt das Klassendiagramm noch weitere Elemente und Eigenschaften, wie etwa Initialwerte, Wertebereiche, Operationen oder Abhängigkeiten.

### 3.1.3 Object Constraint Language

Auch wenn die grafische Darstellung von Modellen meist als leichter verständlich und übersichtlicher gilt als eine Beschreibung in Textform, so hat sie doch einen entscheidenden Nachteil: Nicht alle Eigenschaften, insbesondere Nebenbedingungen, lassen sich grafisch visualisieren (Rumpe, 2004). Aus diesem Grund erlaubt ein UML-Klassendiagramm

die zusätzliche Angabe solcher Bedingungen als Text. Die Sprache, in der die Bedingungen formuliert werden, ist frei wählbar, sei es die natürliche Sprache, eine Programmiersprache oder Ähnliches. Dabei sollte eine formale Sprache vor einer natürlichen bevorzugt werden, um Fehlinterpretationen und Missverständnissen vorzubeugen. Empfohlen, aber nicht vorgeschrieben, wird die *Object Constraint Language*, kurz *OCL*, die auch Bestandteil des UML-Standards ist.

Die OCL ist eine eigens für die Definition und Beschreibung von Bedingungen entwickelte Sprache, die in Form so genannter Invarianten sowie Vor- und Nachbedingungen von Methoden vorliegt. Sie ist zwar an die Sprache der Mathematik, insbesondere die Prädikatenlogik, angelehnt, ihr liegt aber keine Programmiersprache zugrunde.

Gemäß Rumpe (2004) beschreiben *Invarianten* Eigenschaften eines Systems, die zu jedem Zeitpunkt gelten sollen. Eine Bedingung ist eine boolsche Aussage über das System und ist meistens in einen *Kontext* gebettet, welcher alle Klassen-, Methoden- und Attributnamen umfasst, die in der Bedingung genannt werden. Wird eine Bedingung *interpretiert*, also anhand einer konkreten Objektstruktur ausgewertet, so liefert sie als boolscher Ausdruck den Wert *wahr* zurück, wenn die Bedingung für diesen spezielle Fall erfüllt ist, oder *falsch*, wenn sie nicht erfüllt ist.

Die *Vorbedingung* einer Methode muss, wie der Name besagt, *vor* Ausführung einer Methode gelten, damit diese ein plausibles und definiertes Ergebnis liefern kann. Bei Nichteinhaltung der Bedingung kann und darf das Ergebnis, welches die Methode zurückliefert, nicht interpretiert und verwendet werden. Eine *Nachbedingung* beschreibt die Eigenschaften, die *nach* Ausführung der Methode gelten. Das Tupel, bestehend aus Vor- und Nachbedingung, wird als *Methodenspezifikation* bezeichnet.

Seit UML Version 2.0 umfasst OCL auch Definitionen einer *Object Query Language*, mit der es möglich ist, den Status eines Systems, insbesondere Werte von Objekten abzufragen, ohne dabei das System selbst zu verändern. Somit können Abfragen in Bedingungen verwendet werden.

### 3.1.4 Fazit

Die Unified Modeling Language ist vielseitig anwendbar. Nahezu alle Softwaresysteme sollen sich durch ihre Verwendung darstellen lassen, so auch Datenbankanwendungen und das zugrunde liegende Datenbankmodell. Ihre volle Stärke entfaltet sie aber erst in Verbindung mit objektorientierten Programmiersprachen, wie etwa Java oder C++. Auch die meisten CASE-Tools setzen UML-Modelle in eine objektorientierte Sprache um.

CASE-Tools für Datenbankmodelle hingegen arbeiten mit ER-Diagrammen, welche in SQL-Anweisungen oder gleich direkt in ein Datenbankschema übersetzt werden. Umgekehrt lassen sich mit ihrer Hilfe ER-Diagramme aus vorgegebenen Datenbankschemata erstellen.

Keine der im SysTax-Projekt zum Einsatz kommenden Programmiersprachen ist objektorientiert[2]. Auch war zu Projektbeginn kein Werkzeug bekannt, welches UML mit Oracle-Software, insbesondere mit Oracle-Forms und Oracle-PL/SQL, verbindet. Aus diesem Grund werden für die Darstellung der SysTax-Datenbankmodelle Entity-Relationship-Diagramme verwendet. Daneben ist es auch aus Sicht der Nutzer angebracht, die Modelle als ER-Diagramme zu veröffentlichen, da die meisten Projektpartner Erfahrung haben mit Datenbankentwicklung und somit auch mit ER-Diagrammen.

Für ER-Diagramme gibt es mehrere Notationen, die sich vor allem in der Darstellung der Kardinalität, also der Komplexitätsgrade, unterscheiden. Entitäten werden normalerweise als Rechtecke, Relationen als Rauten mit Verbindungslinien zu ihren Entitäten symbolisiert. Die aus dem *Information Engineering*[3] stammende IE-Notation hingegen verzichtet auf das Rautensymbol und stellt binäre Relationen nur durch eine Linie dar. Komplexitätsgrade werden durch Symbole visualisiert, wie sie auch in vielen Programmen zur Anwendung kommen, die Datenbankschemata grafisch darstellen[4]. Mehrstellige, rekursive und *is_a*-Relationen werden auch in der IE-Notation durch eine Raute repräsentiert, die hier mit einem Rechteck umschlossen wird. So genannte *weak entities*, also Entitäten, die nur existieren, wenn andere Entitäten vorhanden sind, werden als doppeltes Rechteck dargestellt. Aus Gründen der Übersichtlichkeit wird diese IE-Notation für die SysTax-Datenmodelle verwendet.

Da die Unified Modeling Language nicht zum Einsatz kommt, wird auch bei der Beschreibung der Integritätsbedingungen auf die Verwendung der Object Constraint Language verzichtet, obwohl diese nicht an UML gebunden ist. Sie erinnert an eine Programmiersprache, während Entity-Relationship-Modelle eher mengenorientiert sind und damit eine Schreibweise aus der relationalen Algebra für die SysTax-Modellbedingungen passender erscheint.

Allgemein werden Integritätsbedingungen als Tupel $(O, B, A, R)$ geschrieben, mit:

O: betroffene Objekte

B: Bedingung

A: Auslöseregel (Wann soll die Bedingungen überprüft werden?)

R: Reaktionsregel (Was soll bei Verletzung der Bedingung getan werden?)

Bei den Bedingungen des SysTax-Modells handelt es sich nicht um Integritätsbedingungen im eigentlichen Sinne[5], sondern um Modellbedingungen, die sich aus der biologischen

[2] wenn man von Perl für die Webausgaben absieht

[3] Das *Information Engineering* stellt Konzepte und Methoden für die integrierte Softwareentwicklung, insbesondere von Informationssystemen, zur Verfügung. Dabei unterstützt es durchgängig den gesamten Software-Entwicklungsprozess, von der Planung, Systemanalyse und -entwurf bis hin zur Implementierung (http://www.v-modell.iabg.de/au251htm/a2-2kome/iem.htm (Mai 2005)).

[4] beispielsweise Microsoft-Access, das bei vielen Projektpartnern im Einsatz ist

[5] Integritätsbedingungen gelten in erster Linie für Realisierungen eines Modells

Taxonomie ergeben. Für ihre Beschreibung werden mathematische Formulierungen verwendet, da sie grafisch nicht im ER-Diagramm darstellbar sind. Im Folgenden werden die Begriffe „Modellbedingung“ und „Integritätsbedingung“ als gleichberechtigte Synonyme eingesetzt.

Bevor das SysTax-Modell beschrieben wird, sollen noch zwei weitere Modellansätze für Potential Taxa, das IOPI-Modell und der Prometheus-Ansatz, vorgestellt werden.

## 3.2 Existierende Modelle

### 3.2.1 Das IOPI-Modell

Ein Datenmodell, welches das Konzept der Potential Taxa umsetzt, wurde im Rahmen eines Projektes der *International Organization for Plant Information*[6] entwickelt und unter dem Namen „IOPI-Modell“ in Berendsohn (1997) veröffentlicht.

Die wesentlichen Kernpunkte dieses Modells sind:

- Potential Taxa werden gebildet aus der Kombination eines Namens und einer Literaturangabe zusammen mit einer „referenced status assignation“. Diese gibt den Status des Taxons in der jeweiligen Quelle an und liefert gegebenenfalls eine Referenz auf den in dieser Quelle gültigen Namen.

- Taxonomische Informationen, die sich aus der Literaturquelle ergeben, wie z. B. die Einordnung in das hierarchische System oder der Status des Taxons als Synonym, werden dem Potential Taxon zugeordnet. Dies ermöglicht eine parallele Speicherung alternativer Taxonomien und Synonymien.

- Alle sonstigen Informationen werden ebenfalls dem Potential Taxon zugeordnet. Beispiele hierfür wären u. a. Bestimmungen von Sammlungsbelegen, molekulare Daten, Multimediadaten oder Verbreitungsangaben. Diese Daten dürfen auch eine andere Quelle als die Referenz des Potential Taxons haben.

- Gewisse nomenklatorische Daten wie die Typusinformation oder das Basionym sollten direkt dem Namen und nicht einem Potential Taxon zugeordnet sein.

Kurz zusammengefasst bedeutet dies, dass eine Literaturquelle einem oder mehreren Namen jeweils genau einen Status zuweist. Durch diesen Status wird insbesondere ausgedrückt, ob das entsprechende Taxon in dieser Quelle akzeptiert, also „gültig“ ist oder nicht. Im letzteren Fall erhält das Taxon noch ein Verweis auf ein anderes, in dieser Quelle akzeptiertes Taxon. Umgekehrt können einem Namen mehrere, auch unterschiedliche Status aus verschiedenen Literaturquellen zugeordnet sein.

[6] http://plantnet.rbgsyd.gov.au/iopi/iopihome.htm (April 2005)

Wissenschaftliche Namen bestehen, wie schon in Abschnitt 1.2 beschrieben, aus bis zu drei Namenselementen[7], gefolgt von einer Autorenangabe. Jeder dieser Namen besitzt einen systematischen Rang, der fest mit ihm verbunden ist und der auch nicht geändert werden kann, ohne einen neuen Namen zu erzeugen.

Die Existenz solcher Binomen (bzw. Trinomen) bedeutet aber, dass schon der Name selbst eine Klassifikation impliziert. So muss eine Art zu genau der Gattung oder einer ihrer infragenerischen Taxa gestellt sein, die im Artnamen genannt wird. Diese Regel muss jede Implementierung des Modells durch geeignete Integritätsbedingungen einhalten.

Das IOPI-Modell ist speziell an die Anforderungen der botanischen Nomenklatur angepasst. Im Gegensatz zur Zoologie ist es in der Botanik möglich, Taxa außerhalb der Systematik oder der Synonymiezuordnung zueinander in Beziehung zu stellen. Dies ist der Fall bei den so genannten Hybriden und Chimären. Es handelt sich hier um Pflanzen, bei denen zwei oder mehrere Arten miteinander gekreuzt oder durch Pfropfung „veredelt“ werden. Dies kommt vor allem bei Nutzpflanzen zur Anwendung, weshalb an dieser Stelle nicht näher auf die Besonderheiten der Modellierung von botanischen Namen eingegangen werden soll, sondern der wissenschaftliche Name als eine einzige „Meta-“ Entität betrachtet wird, auch wenn er im Modell in mehrere Entitäten unterteilt ist.

Ein weiteres Problem stellt die Verwaltung der Autorenangabe dar. In der Botanik folgt das Zitat des Autors oder der Autoren eines Taxons komplizierten Regeln, die in der Zoologie in dieser Form nicht vorkommen.[8] Als Beispiel sei hier die Art *Muscari botryoides (L.) Mill. ex. DC.* erwähnt: Ursprünglich von Linné als *Hyacinthus botryoides L.* beschrieben, wurde diese Art später von Miller in die Gattung *Muscari Mill.* gestellt. Die Information, dass Miller die Art umkombiniert hat, wurde aus einem Werk von de Candolle entnommen, ausgedrückt durch den Terminus „*ex.*“ im Autorenzitat. Aus diesem Grund sei hier nur am Rande vermerkt, dass im IOPI-Modell das Autorenzitat durch die Unterteilung in mehrere Entitäten normalisiert ist.[9] Aber auch diese Entitäten können unter der „Meta-“ Entität des Namens zusammengefasst werden.

Abbildung 3.1 zeigt als Zusammenfassung von Fig. 4 (S. 289) und Fig. 10 (S. 299) in Berendsohn (1997) das Entity-Relationship-Diagramm des Kernstückes des IOPI-Modells, die Verwaltung der Namen und ihre Verknüpfung mit einer Quelle.

Jedem wissenschaftlichen Namen, hier durch die oben erwähnte „Meta-“ Entität *Taxon Name* dargestellt, sind neben den nomenklatorischen Daten wie z. B. seinem Rang (Entität *Rank*) ein oder mehrere Potential Taxa zugeordnet (*Potential Taxon Name*). Jedes Potential Taxon muss mindestens einer „Quelle“ (Entität *Referenced Status Assignment*) zugewiesen sein, ausgedrückt durch die untere der Relationen zwischen den beiden Entitäten.

[7] Taxa ab Gattung aufwärts mit einem, Arten mit zwei sowie Unterarten mit drei Namensbestandteilen

[8] Zusätzlich werden die Autoren meist gemäß international gültigen, standardisierten Listen abgekürzt.

[9] Näheres hierzu siehe Berendsohn (1997), S. 294f.

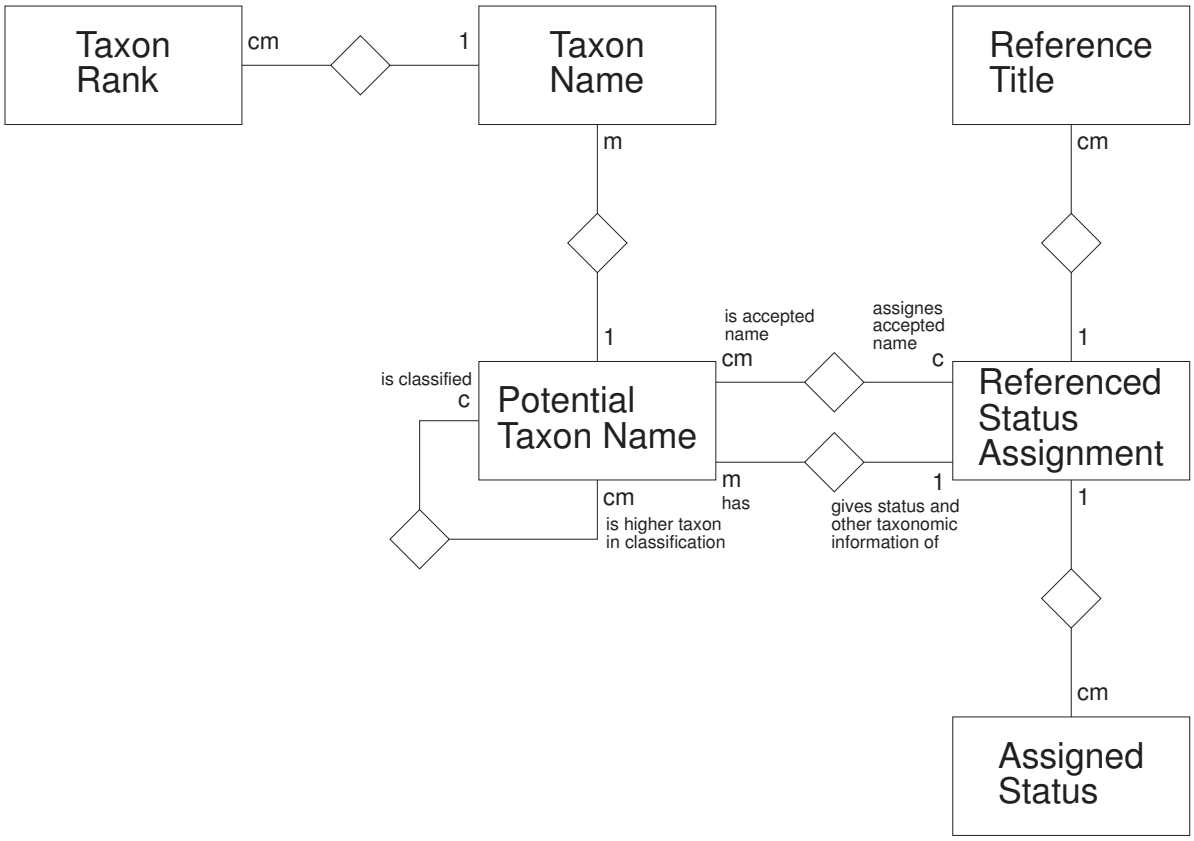


Abbildung 3.1: ER-Diagramm zum Kernstück des IOPI-Modells, entnommen aus Berendsohn (1997)

Durch die Entität *Referenced Status Assignment* wird eine Verknüpfung zwischen der Literatur (*Reference Name*) und dem Status (*Assigned Status*) hergestellt, den ein Potential Taxon in der jeweiligen Literatur besitzt. Durch diese Form der Modellierung, eine Verknüpfung zwischen zwei Entitäten selbst als Entität aufzufassen, ist es möglich, diese mit anderen Entitäten in Beziehung zu setzen.

Eine Quelle ordnet einem Taxon nicht nur einen Status zu, sondern kann auch einen Verweis auf ein anderes Taxon liefern, ausgedrückt durch die obere der beiden Relationen zwischen *Potential Taxon Name* und *Referenced Status Assignment*. Damit können beliebig viele (Synonymie-) Beziehungen zwischen Potential Taxa erstellt werden.

Die Klassifikation der Taxa, also die Gruppierung von Taxa und deren Zuweisung zu Taxa höherer Stufen, wird durch die rekursive Relation der Entität *Potential Taxon Name* mit sich selbst dargestellt. Jedes Potential Taxon kann dabei genau einem anderen höherrangigen Potential Taxon zugeordnet sein, aber auch selber mehrere untergeordnete Potential Taxa haben. Da einem Name mehreren Potential Taxa zugeordnet sein können, lassen sich hiermit mehrere verschiedene taxonomische Hierarchien parallel abbilden.

### 3.2.2 Prometheus-Ansatz

In Berendsohn[10] wird explizit darauf hingewiesen, dass das IOPI-Modell erstellt wurde, um die Ergebnisse des Klassifizierungsprozesses darzustellen, und nicht um direkt diesen Prozess zu unterstützen. Um dies zu bewerkstelligen, ist es nicht nur nötig, eine Referenz auf die Umschreibung eines Potential Taxon zu speichern, sondern auch die Umschreibung selbst, d. h. sämtliche Sammlungsbelege und untergeordnete Taxa. Die größte Schwierigkeit hierbei ist nicht die Verwaltung einer großen Menge von Sammlungsbelegen, sondern deren Verfügbarkeit (Berendsohn, 1995).

Dieser Aussage wird in Pullan et al. (2000) widersprochen. Gemäß diesem Werk sollten sämtliche Daten, die zu einer Klassifikation beitragen, in einer taxonomischen Publikation angegeben sein. Das dort vorgestellte „Prometheus-Modell“ verwendet diesen Ansatz, um einen taxonomisch arbeitenden Biologen bei seiner Arbeit zu unterstützen. Dabei basiert es ebenfalls auf dem Prinzip der Potential Taxa. Im Gegensatz hierzu wird jedoch nicht nur auf die Referenz einer Umschreibung verwiesen, sondern diese wird vollständig angegeben.

Obwohl in SysTax ebenfalls eine Sammlungsverwaltung integriert ist, wurde die Idee des Prometheus-Modells nicht weiterverfolgt, da hierfür – zumindest zur Zeit – kein Bedarf besteht.

### 3.2.3 Das SysTax-Modell

Gemäß der Beschreibung des IOPI-Modells kann ein neues Potential Taxon immer dann gebildet werden, wenn ein Name in einer Quelle „akzeptiert“ ist. Ein Status hingegen, der die Nichtakzeptanz eines Namens ausdrückt, muss immer einem schon vorhandenen Potential Taxon zugewiesen werden. Besitzt ein Name bereits mehrere Potential Taxa, so besteht in diesem Fall die Schwierigkeit darin, zu entscheiden, welchem der Potential Taxa der Status zugewiesen werden soll, sodass es zu einem Synonym wird. Während einem Taxonomen meistens die entsprechenden Angaben vorliegen, anhand derer er eine solche Entscheidung treffen kann, bedeutet es für einen sammlungsorientiert arbeitenden Wissenschaftler, dem es hauptsächlich darauf ankommt, dass ein Beleg unter verschiedenen, zueinander synonymen Namen gefunden werden kann, oft einen Mehraufwand, diese Daten zu ermitteln.

Da die Mehrzahl der mit SysTax arbeitenden Nutzer sammlungsorientiert arbeitet, wurde zu Beginn des Projekts beschlossen, Synonyme nicht entsprechend dem IOPI-Modell zu behandeln. Folglich konnte das IOPI-Modell nicht unmodifiziert in SysTax übernommen werden und so wurde ein eigenes Modell entwickelt. In Grundzügen ist es allerdings an das IOPI-Modell angelehnt: In beiden Ansätzen können aus jedem Namen mehrere Potential Taxa gebildet werden und die Zuordnung zu einem höherrangigem Taxon erfolgt ausschließlich über diese Potential Taxa. In einem wesentlichen Punkt weist es

[10] vgl. Berendsohn (1997), S. 300

allerdings vom IOPI-Modell ab: Jedes Potential Taxon soll im SysTax-Modell genau eine Quelle und damit genau einen Status ('gültig' oder 'ungültig') besitzen. Synonyme der Form „Quelle X sagt, Taxon A sei Synonym von Taxon B" lassen sich damit in einer rekursiven Relation der Potential Taxa verwalten.

Folgenden Anforderungen soll das SysTax-Modell genügen:

- Potential Taxa werden gebildet aus der Verknüpfung eines Namens und einer Quelle, die diesem Namen einen Status zuweist.
- Jeder Name muss genau ein „neutrales Potential Taxon" besitzen, das gebildet wird aus der Verknüpfung des Namens mit der „neutralen Quelle".
- Die Anzahl der Potential Taxa eines Namens ist nicht begrenzt.
- Die Klassifikation, also die Zuordnung eines Namens zu einem anderen Namen höheren Ranges, erfolgt ausschließlich über die Potential Taxa.
- Synonyme werden gebildet durch die Zuordnung zweier Potential Taxa.
- Die Namen der in einer Synonymiebeziehung stehenden Potential Taxa müssen denselben Hauptrang haben.
- Die neutralen Potential Taxa dürfen keine Synonyme besitzen und selber kein Synonym sein.
- Bei regulären Synonymen muss die Quelle der beteiligten Potential Taxa identisch sein.
- Bei Konzeptsynonymen muss die Quelle der beteiligten Potential Taxa verschieden sein.
- Jedes 'ungültige' Potential Taxon muss als reguläres Synonym genau einem anderen Potential Taxon beliebigen Status zugewiesen werden.
- Ein 'gültiges' Potential Taxon darf nicht als Synonym in einer regulären Synonymiebeziehung stehen.
- Für Konzeptsynonyme sind die Status der beteiligten Potential Taxa ohne Relevanz.
- Nicht referenzierte Sekundärinformationen werden dem neutralen Potential Taxon zugeordnet.

Abbildung 3.2 zeigt das Entity-Relationship-Diagramm des SysTax-Modells zur Verwaltung der wissenschaftlichen Namen und ihrer Potential Taxa in der IE-Notation. Aus Gründen der Übersichtlichkeit wird auf die Angabe von Attributen verzichtet. Lediglich die absolut notwendigen Attribute *Rang* ($\mathcal{R}$), *Status* ($\mathcal{S}$) und *Synonymietyp* ($\mathcal{A}$) sind abgebildet. Die Entität $\mathcal{T}$ repräsentiert die Taxonnamen und die Entität $\mathcal{Q}$ die Quellen.

Die Potential Taxa, eigentlich eine mehrwertige (mc-mc-) Verknüpfung zwischen Namen und Quellen, werden in diesem Modell als die schwache Entität $\mathcal{PT}$ und nicht als Relation zwischen der Namen- und der Quellenentität modelliert. Letzteres ist nicht möglich, beziehungsweise nicht eindeutig im ER-Diagramm darstellbar, da die „Entität“ der Potential Taxa in zwei rekursiven Relationen sowohl zu sich selbst in Beziehung steht als auch mit anderen Entitäten außerhalb des taxonomischen Modells verknüpft ist. Als Attribut besitzt $\mathcal{PT}$ den Status ($\mathcal{S}$), welcher die Quelle dem jeweiligen Potential Taxon verleiht.

Die Klassifikation der Potential Taxa wird – analog dem IOPI-Modell – mit der rekursiven Relation $\mathcal{C}$ modelliert, die außer dem ranghöchsten jedes Element, also jedes Potential Taxon aus $\mathcal{PT}$, mit einem „ranghöheren“ Potential Taxon verknüpft. Synonymiebeziehungen zwischen Potential Taxa werden mit der rekursiven Relation $\mathcal{Z}$ dargestellt, welche zusätzlich das Attribut $\mathcal{A}$ (Synonymietyp) besitzt.

Eine Synonymieverknüpfung, insbesondere von Konzeptsynonymen, kann theoretisch ebenfalls eine eigene Quelle besitzen, die unabhängig von der Quelle der beteiligten Potential Taxa ist. Dies wird ausgedrückt durch die Beteiligung der Entität $\mathcal{Q}$ an der Relation $\mathcal{Z}$. Aus obigen Modellanforderungen und aus den biologischen Gegebenheiten folgt, dass die Quelle der Synonymieverknüpfung regulärer Synonyme identisch sein muss mit der Quelle der beteiligten Potential Taxa. Für die konsistente Verwaltung der Konzeptsynonyme ist es notwendig, dass die Quelle der Synonymie identisch ist mit der Quelle des „gültigen“ Potential Taxon, während die Quelle des „ungültigen“ Potential Taxons von dieser verschieden sein muss. Wären die Quellen identisch, so würde es sich um eine reguläre Synonymie und nicht um den Vergleich verschiedener Potential-Taxa-Konzepte. Da jedes Element aus der Relation $\mathcal{Z}$, also jede Synonymiebeziehung, entweder zur Gruppe der regulären Synonymiebeziehungen oder zur Gruppe der Konzeptsynonyme gezählt werden muss, ist die Einbeziehung der Entität $\mathcal{Q}$ in Relation $\mathcal{Z}$ nicht notwendig. Jedoch wird der Vollständigkeit halber die Beteiligung von $\mathcal{Q}$ in Abbildung 3.2 gezeigt und auch in den unten folgenden Bedingungen aufgeführt.

Zur Beschreibung der nicht im ER-Diagramm grafisch darstellbaren Modellbedingungen seien die Attribute als folgende Mengen definiert:

$\mathcal{R}$ := {‘Reich’, ‘Unterreich’, . . . , ‘Form’}: Menge der taxonomischen Ränge,
$\mathcal{S}$ := {‘gültig’, ‘ungültig’}: Menge der Status eines Taxons, und
$\mathcal{A}$ := {‘reguläres Synonym’, ‘Konzeptsynonym’}: Menge der Synonymietypen.

Sei also Entität $\mathcal{T}$ die Menge aller Taxonnamen und bezeichne $r_t$ den Rang eines Namens $t \in \mathcal{T}$. Weiter definiere ‘>’ eine Ordnung auf der Menge $\mathcal{R}$[11], und $h(r)$ eine Funktion,

[11] Reich > Unterreich > . . . > Art > . . . > Form

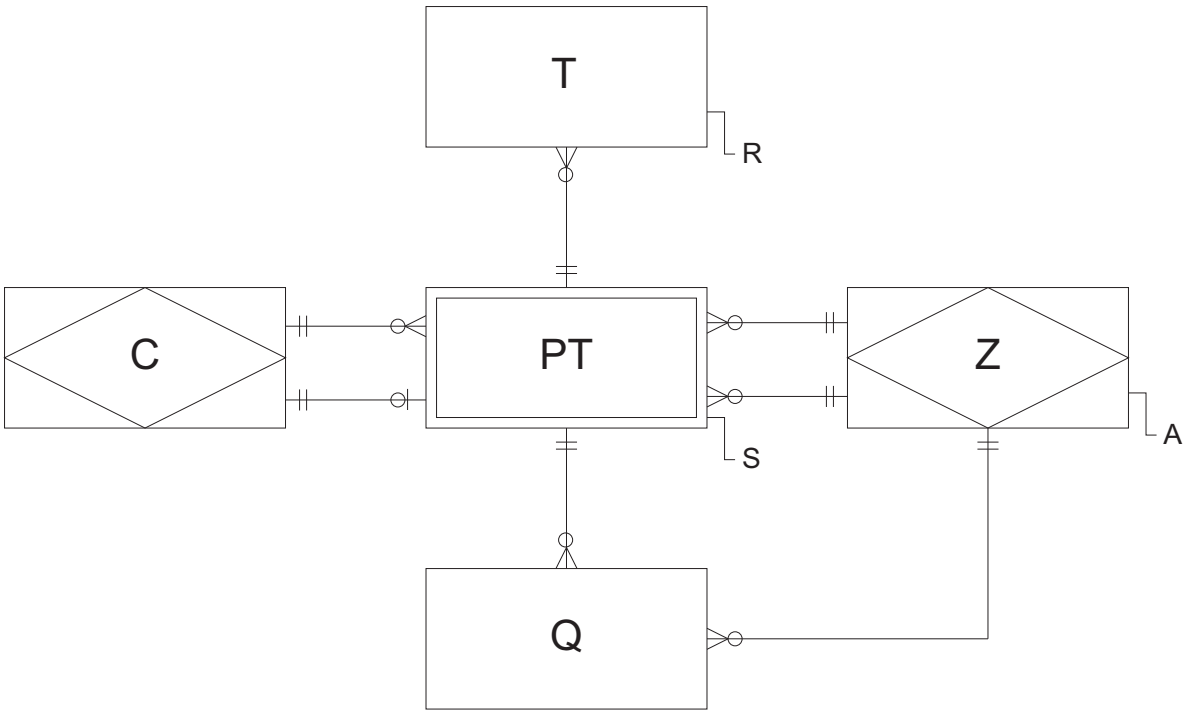


Abbildung 3.2: ER-Diagramm des SysTax-Modells

die jedem taxonomischen Rang seinen jeweiligen Hauptrang zuweist:

$$\begin{aligned} h(\text{‘Reich’}) &= h(\text{‘Unterreich’}) = \text{‘Reich’}, \\ h(\text{‘Stamm’}) &= h(\text{‘Unterstamm’}) = h(\text{‘Überklasse’}) = \text{‘Stamm’}, \\ &\dots \\ h(\text{‘Art’}) &= h(\text{‘Unterart’}) = \dots = h(\text{‘Form’}) = \text{‘Art’}. \end{aligned}$$

Dann ist die schwache Entität $\mathcal{PT}$ (Menge der Potential Taxa) sowie die Relationen $\mathcal{C}$ (Hierarchie der Potential Taxa) und $\mathcal{Z}$ (Synonymiebeziehungen zwischen Potential Taxa) definiert als

$$\begin{aligned} \mathcal{PT} &:= \{p = (t, q, s) \mid t \in \mathcal{T}, q \in \mathcal{Q}, s \in \mathcal{S}\}, \\ \mathcal{C} &:= \left\{c = (p, p') = \left((t, q, s), (t', q', s')\right) \mid p, p' \in \mathcal{PT}, t \neq t'\right\}, \text{ und} \\ \mathcal{Z} &:= \{z = (p, p', \varphi, a) \mid p', p \in \mathcal{PT}, p' \neq p, \varphi \in \mathcal{Q}, a \in \mathcal{A}\}. \end{aligned}$$

In der Definition von $\mathcal{C}$ steht $p'$ für das Potential Taxons des taxonomischen Vorgängers des Potential Taxons $p$, während in der Relation $\mathcal{Z}$ der Synonymiebeziehungen $p$ dem Synonym und $p'$ seinem Synonymievorgänger (meistens, aber nicht notwendigerweise das gültige Potential Taxon) entspricht.

Seien weiterhin mit $\mathcal{RZ}$ die Menge der regulären Synonymieverknüpfungen und mit $\mathcal{KS}$ die Menge der Konzeptsynonyme zwei disjunkte Teilmengen von $\mathcal{Z}$ bezeichnet und durch

$$\mathcal{RZ} := \{z = ((t,q,s),(t',q',s'),\varphi,a) \in \mathcal{Z} \mid a = \text{‘reguläres Synonym’}, t \neq t', q = q' = \varphi, s = \text{‘ungültig’}, s' \in \mathcal{S} \text{ beliebig}\}$$
$$\mathcal{KS} := \{z = ((t,q,s),(t',q',s'),\varphi,a) \in \mathcal{Z} \mid a = \text{‘Konzeptsynonym’}, t \neq t', q \neq q', s, s' \in \mathcal{S} \text{ beliebig}\}$$

definiert. Dann müssen für das in Abbildung 3.2 gezeigte Datenmodell folgende Bedingungen erfüllt sein:

- Existenz eines „Neutralen Potential Taxons“:
  Sei $nq \in \mathcal{Q}$ die „neutrale Quelle“ und $\mathcal{NPT} := \{(t,q,s) \in \mathcal{PT} \mid q = nq, s = \text{‘gültig’}\}$ die Menge der „neutralen Potential Taxa“. Dann muss gelten:
  Für alle $t \in \mathcal{T}$ existiert genau ein $npt \in \mathcal{NPT}$.
  Weiter muss für alle $npt \in \mathcal{NPT}$ und $p \in \mathcal{PT}$ beliebig gelten:
  $(npt, p, \varphi, a) \notin \mathcal{Z}$ und $(p, npt, \varphi, a) \notin \mathcal{Z}$ (für $\varphi, a$ beliebig)[12].

- Für alle $p = (t,q,s) \in \mathcal{PT}$ mit $r_t \neq$ ‘Reich’ muss es ein $p' \in \mathcal{PT}$ geben, sodass $(p,p') \in \mathcal{C}$ existiert.
  Jedes Potential Taxon, welches nicht den höchsten Rang besitzt, muss einem taxonomischen Vorgänger zugeordnet werden.

- Für alle $c = ((t,q,s),(t',q',s')) \in \mathcal{C}$ muss gelten: $r_{t'} > r_t$.
  Das bedeutet, dass der Rang des Vorgängers eines Potential Taxons in der taxonomischen Hierarchie höher sein muss als der Rang des Potential Taxons selbst.

- Für alle $c = ((t,q,s),(t',q',s')) \in \mathcal{C}$ muss gelten: $r_t \neq h(r_t) \Rightarrow h(r_t) = h(r_{t'})$
  Falls der Rang eines Potential Taxons nicht schon ein Hauptrang ist, so muss sein Vorgänger denselben Hauptrang besitzen. Damit ist gewährleistet, dass die Hauptstufen innerhalb einer Hierarchie besetzt sind. So muss etwa eine Untergattung einer Gattung zugewiesen werden und darf z. B. nicht an eine Familie gehängt werden.

- Für alle $c = ((t,q,s),(t',q',s')) \in \mathcal{C}$ mit $r_t =$ Art muss gelten: $h(r_{t'}) =$ Gattung.
  Der Hauptrang des Vorgängers einer Art muss immer eine Gattung sein, der Rang des Vorgängers von Potential Taxa auf allen anderen Stufen ist entsprechend den Modellbedingungen beliebig.

- Für alle $c = ((t,q,s),(t',q',s')) \in \mathcal{C}$ mit $q$ beliebig muss gelten: $q' \in \{nq, q\}$
  Damit muss der Vorgänger eines neutralen Potential Taxon ebenfalls ein neutrales Potential Taxon sein. Der Vorgänger jedes anderen Potential Taxons muss entweder ein neutrales Potential Taxon sein oder aus derselben Quelle stammen wie das Potential Taxon selbst.

[12] d.h. das neutrale Potential Taxon darf weder Synonym sein noch Synonyme besitzen

- Für Taxa, deren Namen sich aus den Namen mehrerer Taxa mit unterschiedlichen Rängen zusammensetzen, also vor allem Bi- und Trinomen auf Artebene, muss gelten: Die durch die Relation $\mathcal{C}$ definierte Hierarchie darf dem zusammengesetzten Namen nicht widersprechen, da es sich bei diesem Namen um eine „grammatikalische Regel“ und nicht um eine Klassifikation im Sinne der Potential Taxa handelt.

- Für alle $z \in \mathcal{Z}$ muss gelten: $z \in \mathcal{RZ}$ oder $z \in \mathcal{KS}$, d. h. $\mathcal{RZ} \cup \mathcal{KS} = \mathcal{Z}$.

  $\Rightarrow$ Für alle $z \in \mathcal{Z}$ mit $a =$ ‘reguläres Synonym’ gilt somit: $z \in \mathcal{RZ}$.
  D. h. es existiert kein reguläres Synonym mit $q' \neq q$, $q' \neq \varphi$ oder $\varphi \neq q$.

- Für alle $z \in \mathcal{Z}$ muss gelten: $h(r_t) = h(r'_t)$.
  Dies bedeutet, dass nur Taxa auf derselben Hauptstufe miteinander synonymisiert werden dürfen.

- Für alle $rz = \big((t, q, s), (t', q, s'), q, a\big) \in \mathcal{RZ}$ mit $s' =$ ‘ungültig’ muss gelten: Es existiert ein $rz' = \big((t', q, s'), (t'', q, s''), q, a\big) \in \mathcal{RZ}$ mit $t'' \neq t'$ und $t'' \neq t$ sowie $s'' \in \mathcal{S}$ beliebig.
  Damit wird ein Synonymiebaum rekursiv aufgebaut, bei dem ein ungültiges Potential Taxon Synonym eines anderen ungültigen Potential Taxons sein kann, während das oberste Potential Taxon der Hierarchie hingegen immer gültig ist. Zusammen mit der Definition der Menge $\mathcal{RZ}$ ist sichergestellt, dass das oberste Potential Taxon auch das einzige Potential Taxon mit Status ‘gültig’ innerhalb eines Baumes ist.

- Für alle $p = (t, q, s) \in \mathcal{PT}$ mit $s =$ ‘ungültig’ muss eine endliche Folge von Potential Taxa $p_1 = (t_1, q_1, s_1), \ldots, p_n = (t_n, q_n, s_n) \in \mathcal{PT}$ existieren mit:
  $t \neq t_1 \neq \cdots \neq t_n$,
  $q = q_1 = \cdots = q_n$,
  $s = s_1 = \cdots = s_{n-1} =$ ‘ungültig’ und $s_n =$ ‘gültig’,
  sowie $(p, p_1, q, a), (p_1, p_2, q, a), \ldots, (p_{n-1}, p_n, q, a) \in \mathcal{PT}$.
  Damit ist sichergestellt, dass zu jedem ungültigen Potential Taxon ein gültiges Potential Taxon aus derselben Quelle existiert und beide innerhalb eines Synonymiebaumes miteinander verknüpft sind, entweder direkt (falls $n = 1$) oder über eine Reihe weiterer ungültiger Potential Taxa mit derselben Quelle (Fall $n > 1$).

- Für alle $rz_1 = (p_1, p'_1, q_1, a) \in \mathcal{RZ}$ und $rz_2 = (p_2, p'_2, q_2, a) \in \mathcal{RZ}$ mit $p_1 = p_2$ und $q_1 = q_2$ muss gelten: $p'_1 = p'_2$.
  D. h. ein ungültiges Potential Taxon darf nicht gleichzeitig Synonym zweier verschiedener Potential Taxa derselben Quelle sein.

- Für alle $ks \in \mathcal{KS}$ muss gelten: $\varphi = q$.
  Durch Weglassen dieser Einschränkung könnten Fälle behandelt werden, in denen eine Quelle A angibt, Taxon 1 sec. B stehe mit Taxon 2 sec. C in einer Konzeptsynonymiebeziehung. Gemäß aktueller taxonomischer Literatur und der Meinung von Biologen kommt dieser Fall nicht vor. Nebeneffekt dieser Bedingung ist, dass

die Verknüpfung zwischen $\mathcal{Z}$ und $\mathcal{Q}$ weggelassen werden kann, da die Quelle des Vorgängers des Konzeptsynonyms[13] auch die Quelle der Konzeptsynonymie selbst ist. Zusammen mit der Bedingung $q \neq q'$ in der Definition der Menge $\mathcal{KS}$ wird durch diese Einschränkung verhindert, dass mehrstufige Konzeptsynonymiebäume gebildet werden können, die ebenfalls laut Meinung der Biologen nicht vorkommen.

[13] Im Gegensatz zur regulären Synonymie ist die Konzeptsynonymie eine ungerichtete Beziehung zwischen zwei Potential Taxa.

# 4 Realisierung des SysTax-Modells

## 4.1 Realisierung des SysTax-Modells – Namen und Potential Taxa

Dieser Abschnitt beschäftigt sich mit der Umsetzung jenes Teils des SysTax-Modells aus Abschnitt 3.2.3, der sowohl für die biologischen Namen als auch für die Potential Taxa und ihrer Klassifikation zuständig ist. Es werden zwei Möglichkeiten zur Realisierung der Namensverwaltung vorgestellt und ihre Vor- und Nachteile diskutiert.

### 4.1.1 Potential Taxa und ihre Quellen

Für das Datenmodell ist es ohne Belang, mit welcher Art von Quelle ein Name verknüpft wird, um ein Potential Taxon zu bilden. Da sowohl Literaturzitate als auch Internetadressen oder Meinungen von Arbeitsgemeinschaften, Instituten und Einzelpersonen als Quelle dienen können, wird die Menge $\mathcal{Q}$ des Modells in die disjunkten Teilmengen 'Literatur', 'Url', 'Institut ' und 'Person' zerlegt. Diese Teilmengen werden bei der Umsetzung in SysTax als eigenständige Bereichen verwaltet, hinter denen – besonders im Fall der Literatur – teils komplexe Datenmodelle mit mehreren Entitäten und Relationen stehen. Auf diese Modelle und ihre Umsetzung soll hier nicht näher eingegangen werden.[1] Wichtig in diesem Zusammenhang ist lediglich, dass jedes Element der einzelnen Teilmengen durch einen Schlüssel eindeutig identifiziert werden kann. Eine Realisierung der Menge $\mathcal{Q}$ in Datenbanktabellen stellt somit auch eine Realisierung einer „*is_a*"-Beziehung zwischen $\mathcal{Q}$ und den Bereichen 'Literatur', 'Url', 'Institut' und 'Person' dar.

| SOI |
|---|
| SoiId |
| LiteraturId |
| UrlId |
| InstitutId |
| PersonId |

Abbildung 4.1: Definition der Tabelle *SOI*

[1] Für nähere Informationen hierzu siehe: Homepage des SysTax-Projektes, http://www.biologie.uni-ulm.de/systax.

Abbildung 4.1 zeigt die Definition der Tabelle *SOI* (Abkürzung für „source of information“) als mögliche Umsetzung der Menge $\mathcal{Q}$ des Modells. Ihren Primärschlüssel beinhaltet die Spalte *SoiId*, die restlichen Spalten stellen die Fremdschlüssel aus den oben genannten Bereichen dar.

Die schwache Entität $\mathcal{PT}$ und die Relation $\mathcal{C}$ des SysTax-Modells werden in einer einzelnen Tabelle *PTAXON* umgesetzt, deren Definition in Abbildung 4.2 gezeigt wird.

| PTAXON |
|---|
| ZooId |
| TaxId |
| SoiId |
| VZooId |
| StatusId |

Abbildung 4.2: Definition der Tabelle *PTAXON*

Spalte *TaxId* beinhaltet als Fremdschlüssel den für jeden Namen (aus der Menge $\mathcal{T}$ des Modells) eindeutigen Schlüssel; siehe hierzu auch die nachfolgenden Abschnitte. Die Realisierung der Beziehung zwischen den Potential Taxa und ihren Quellen erfolgt durch den Fremdschlüssel *SoiId*.

In der Tabelle *PTAXON* wird also einem Namen, repräsentiert durch seinen Schlüssel *TaxId*, eine Quelle zugeordnet, die bestimmt ist durch ihre *SoiId*. Jeder Eintrag dieser Tabelle entspricht also genau einem Potential Taxon. Aus dem Modell und seinen Integritätsbedingungen folgt, dass jede Kombination von *TaxId* und *SoiId* nur einmal vorkommen darf. Somit ist das Tupel (*TaxId*, *SoiId*) für jede Zeile aus *PTAXON* eindeutig und könnte als ihr Primärschlüssel betrachtet werden. Um Komplikationen zu vermeiden, die mehrspaltige Schlüssel mit sich bringen, und aus Gründen der Übersichtlichkeit wird ein weiteres Attribut *ZooId* als Primärschlüssel der Tabelle *PTAXON* eingeführt, das jedes Potential Taxon eindeutig beschreibt.

Das Attribut $\mathcal{S}$ der Relation $\mathcal{PT}$ des SysTax-Modells, welches ausdrückt, ob ein Name in einer Quelle akzeptiert ist oder nicht, wird mit Spalte *StatusId* umgesetzt, die als Fremdschlüssel eine Verknüpfung mit der Tabelle herstellt, in der die Namen der verschiedenen Status abgelegt sind.

Die Klassifikation des Potential Taxons, also seine Zuweisung zu einem anderen Potential Taxon höheren Ranges, erfolgt durch die Spalte *VZooId*, die als Fremdschlüssel die *ZooId* des Vorgängers jedes Potential Taxons enthält. Durch sie wird die rekursive Relation $\mathcal{C}$ der Potential Taxa ($\mathcal{PT}$) realisiert.

Bei jeder Implementierung von Anwendungen, die nicht ausschließlich Daten auslesen, sondern auch Datensätze einfügen oder verändern, muss dafür Sorge getragen werden, dass die in Abschnitt 3.2.3 vorgestellten Integritätsbedingungen bei keinem Eintrag in Tabelle *PTAXON* verletzt werden.

### 4.1.2 Namen - Methode 1

Eine Möglichkeit der Umsetzung der Menge $\mathcal{T}$ der wissenschaftlichen Namen ist es, jeden Namen, unabhängig von seinem Rang, als eigenständiges Element zu betrachten. Eine Tabelle *ZNAME*, vorgestellt in Abbildung 4.3, besteht dann aus den vier Spalten *TaxId*, *RangId*, *Name* und *Autor*, wobei die erste den Primärschlüssel, die zweite einen Verweis auf den Rang des Namens und die letzten beiden Spalten den Namen selbst und das Autorenzitat beinhalten.

| ZNAME |
|---|
| TaxId |
| RangId |
| Name |
| Autor |

Abbildung 4.3: Definition der Tabelle *ZNAME*, 1. Methode

In der Botanik gehört das Autorenzitat zum Taxonnamen dazu, in der Zoologie ist es kein Namensbestandteil und kann theoretisch auch weggelassen werden, obwohl dies von den zoologischen Nomenklaturregeln nicht empfohlen wird. Trotz der Existenz so genannter Homonyme[2] ist es nicht sinnvoll, das Autorenzitat etwa mittels einer mc-mc-Relation vom Namen zu trennen, denn auch Homonyme repräsentieren unterschiedliche Taxa. Um dennoch die Trennung zwischen Namen und Autor zu gewährleisten, werden unterschiedliche Tabellenspalten verwendet.

Dieses Autorenfeld selbst ist nicht normalisiert, denn es enthält eine Unterstruktur: Ein Autorenzitat setzt sich in der Zoologie aus einem oder mehreren Personennamen zusammen, auf die eine Jahreszahl folgt. Eine Person kann dabei, entweder alleine oder zusammen mit anderen, mehreren Taxa als Autor dienen, möglicherweise sogar mit derselben Jahreszahl. Bei Namen auf Artebene kann das komplette Autorenzitat in runde Klammern gesetzt sein, falls die Art umkombiniert wurde. In der Botanik wird in diesem Fall zusätzlich der Autor, der die Umkombination durchgeführt hat, dem Autorenzitat hinzugefügt.

Unabhängig davon kann in der Zoologie der Autor bzw. die Autoren und/oder die Jahreszahl mit eckigen Klammern versehen sein, um anzuzeigen, dass dieses Zitat aus Sekundärquellen erschlossen wurde. In der Botanik wird dieser Sachverhalt dadurch dargestellt, dass der Autor der Sekundärliteratur zusammen mit dem Kürzel „*ex*“ ebenfalls dem Autorenzitat hinzugefügt wird.[3]

[2] Homonyme sind gleiche Namen mit demselben taxonomischen Vorgänger, aber mit unterschiedlichen Autorenzitaten. Diese dürfen gemäß den Nomenklaturregeln nicht auftreten, lassen sich aber unter manchen Umständen nicht vermeiden, was insbesondere bei der Speicherung älterer Namen vorkommen kann.

[3] s. auch das Beispiel für einen solchen Artnamen auf Seite 27.

Theoretisch könnte also das Autorenzitat eines Taxons in mehrere Entitäten und Relationen aufgeteilt werden. Dass dies nicht nur umständlich und für ein effektives Arbeiten mit der Datenbank eher hinderlich ist, sondern sogar zu Widersprüchen und Uneindeutigkeiten führen kann, wird in Boos[4] gezeigt.

Struktur und Inhalt der Tabelle *RANG*, welche die Rangstufen verwaltet und somit eine Umsetzung der Attributmenge $\mathcal{R}$ des SysTax-Modells darstellt, zeigt Tab. 4.1.

| RangId | HStufe | Level | Name |
|---|---|---|---|
| 1 | 1 | 0 | Reich |
| 2 | 1 | 1 | Unterreich |
| 3 | 2 | 0 | Stamm |
| 4 | 2 | 1 | Unterstamm |
| 5 | 2 | 2 | Überklasse |
| 6 | 3 | 0 | Klasse |
| 7 | 3 | 1 | Unterklasse |
| 8 | 3 | 2 | Kohorte |
| 9 | 3 | 3 | Überordnung |
| 10 | 4 | 0 | Ordnung |
| 11 | 4 | 1 | Unterordnung |
| 12 | 4 | 2 | Überfamilie |

| RangId | HStufe | Level | Name |
|---|---|---|---|
| 13 | 5 | 0 | Familie |
| 14 | 5 | 1 | Unterfamilie |
| 15 | 5 | 2 | Tribus |
| 16 | 6 | 0 | Gattung |
| 17 | 6 | 1 | Untergattung |
| 18 | 6 | 2 | Sektion |
| 19 | 6 | 3 | Untersektion |
| 20 | 7 | 0 | Art |
| 21 | 7 | 1 | Unterart |
| 22 | 7 | 2 | Varietät |
| 23 | 7 | 3 | Form |

Tabelle 4.1: Inhalt von Tabelle *RANG*

Spalte *HStufe* realisiert die Funktion $h(r)$ aus Abschnitt 3.2.3, welche die einzelnen Ränge zu Hauptstufen gruppiert. Folglich haben Ränge auf derselben Hauptstufe denselben Wert in ihrer Spalte *HStufe*. Spalte *Level* gibt an, auf welcher Ebene innerhalb einer Hauptstufe der jeweilige Rang angesiedelt ist. Der Wert '0' steht hierbei für die Hauptstufe selbst, der Wert '1' für die erste Unterstufe, usw.

Ein erstes Beispiel soll das Zusammenspiel der Tabellen *ZNAME* und *PTAXON* veranschaulichen.[5] Die Klasse der Insekten, *Insecta*, enthält neben weiteren Ordnungen die Ordnung der Schrecken, *Orthoptera*. Diese wiederum ist unterteilt in mehrere Unterordnungen, wovon eine den Namen *Caelifera* trägt. Diese ist selber in mehrere Überfamilien aufgeteilt, zwei davon sind beispielsweise *Pamphagoidea* und *Acridoidea*. Zur ersteren gehört unter anderem die Familie *Pyrgomorphidae*, zur zweiten die Familie *Lentulidae*.

*Insecta* selbst gehört zum Stamm der Gliederfüßer, *Arthropoda*, dieser wiederum zu den Vielzellern, *Metazoa*, eines der Unterreiche des Tierreichs, *Animalia*. Wie diese Struktur in *ZNAME* aussehen könnte, zeigt Tabelle 4.2.

[4] 1992, Seite 33ff.

[5] Die Daten dieses und aller weiteren Beispiele sind direkt der SysTax-Datenbank entnommen (Stand Februar 2005) und spiegeln den momentanen Forschungs- und Erfassungsstand der jeweiligen Datenlieferanten wider. Insbesondere wird das Autorenzitat auch nur bei den Taxa angegeben, die in der Datenbank zusammen mit ihrem Autor abgespeichert sind. Die Verifizierung bzw. Ergänzung des Autorenzitates kann an dieser Stelle nicht erfolgen, da es vor allem bei älteren Namen in verschiedenen Quellen oft unterschiedlich angegeben ist und somit zu einem Gegenstand der taxonomischen Forschung wird.

| TaxId | RangId | Name | Autor |
|---|---|---|---|
| 1 | 1 | Animalia | |
| 2 | 2 | Metazoa | |
| 3 | 3 | Arthropoda | |
| 4 | 6 | Insecta | |
| 5 | 10 | Orthoptera | |
| 6 | 11 | Caelifera | |
| 7 | 12 | Pamphagoidea | |
| 8 | 12 | Acridoidea | |
| 9 | 13 | Pyrgomorphidae | |
| 10 | 13 | Lentulidae | |

Tabelle 4.2: Beispiel für Tabelle *ZNAME*

Da es sich bei dieser Hierarchie um die (in der SysTax-Datenbank eingegebene) Standardtaxonomie handelt, kann für deren Quelle die neutrale Quelle ($SoiId = 1$) herangezogen werden. *PTAXON* könnte dann wie in Tabelle 4.3 gezeigt aussehen.[6]

| ZooId | TaxId | SoiId | VZooId | *(Name)* |
|---|---|---|---|---|
| 101 | 1 | 1 | | *(Animalia)* |
| 102 | 2 | 1 | 101 | *(Metazoa)* |
| 103 | 3 | 1 | 102 | *(Arthropoda)* |
| 104 | 4 | 1 | 103 | *(Insecta)* |
| 105 | 5 | 1 | 104 | *(Orthoptera)* |
| 106 | 6 | 1 | 105 | *(Caelifera)* |
| 107 | 7 | 1 | 106 | *(Pamphagoidea)* |
| 108 | 8 | 1 | 106 | *(Acridoidea)* |
| 109 | 9 | 1 | 107 | *(Pyrgomorphidae)* |
| 110 | 10 | 1 | 108 | *(Lentulidae)* |

Tabelle 4.3: Beispiel für Tabelle *PTAXON*

Angenommen, eine andere Quelle, etwa mit $SoiId = 2$, verzichte auf die Aufteilung von *Caelifera* und stelle die beiden Familien direkt zur Unterordnung. Damit würde diese Quelle die Unterordnung *Caelifera* in einem neuen Kontext sehen und ein neues Potential Taxon dieses Namens erzeugen. Genauso müsste je ein neues Potential Taxon für die beiden Familien erstellt werden, da diese eine neue, alternative Klassifizierung erhalten. Ausgedrückt in den Tabellen hätte dies zur Folge, dass *PTAXON* um die in Tabelle 4.4 dargestellten Zeilen erweitert werden müsste.

Solange nur Taxa auf Gattungsebene oder höher betrachtet werden, wäre diese Form der Umsetzung ideal. Namen auf Artebene jedoch setzen sich aus den Namen verschiedener Taxa unterschiedlicher Stufen zusammen. So wird ein Artname gebildet aus dem Namen

[6] Zur Veranschaulichung wird Spalte *StatusId* weggelassen, für Spalte *ZooId* einen anderen Wertebereich als für *TaxId* gewählt und der Name des jeweiligen Taxons der Zeile angefügt.

| ZooId | TaxId | SoiId | VZooId | *(Name)* |
|---|---|---|---|---|
| 120 | 6 | 2 | 105 | *(Caelifera)* |
| 121 | 9 | 2 | 120 | *(Pyrgomorphidae)* |
| 122 | 10 | 2 | 120 | *(Lentulidae)* |

Tabelle 4.4: Erweiterung der Tabelle *PTAXON*

seiner Gattung ohne deren Autorenzitat und dem Epitheton sowie dem Autorenzitat der Art. Für dieses Binomen ist es gleichgültig, ob die Art direkt der Gattung unterstellt ist oder einem anderen Taxon unterhalb des Gattungsrangs, denn ausschlaggebend für seinen Namen ist nur die Gattung. Unterarten werden sogar aus drei Namen gebildet: aus dem Artnamen (bestehend aus dem Gattungsnamen und dem Epitheton) ohne Autor und dem zweiten Epitheton sowie dessen Autorenzitat.[7]

Obwohl der Name dieser Taxa bereits ihre Klassifikation zumindest teilweise beinhaltet, handelt es sich nicht um eine Hierarchie im Sinne der Potential Taxa mit der Möglichkeit, alternative Hierarchien zu bilden, sondern um eine „grammatikalische“ Regel, welche der Name zu befolgen hat. Jede Klassifikation von Taxa auf Artebene muss die durch den Namen des Taxons vorgegebene Hierarchie einhalten. Dabei muss durch entsprechende Prüfroutinen bei der Implementierung gewährleistet werden, dass bei keinem Potential Taxon auf Artebene diese Regel verletzt wird.

Um obiges Beispiel fortzuführen, wird die Familie *Pyrgomorphidae* weiter unterteilt, unter anderem in die Unterfamilie *Pyrgomorphinae*, welche den Tribus *Pyrgomorphini* enthält. Dieser besitzt mehrere Gattungen, z. B. *Protanita Kevan, 1962* und *Pyrgomorpha Serville, 1838*. Letztere wiederum enthält mehrere Arten, unter anderem *Pyrgomorpha conica (Olivier, 1791)*. Diese Art ist weiter unterteil und enthält u. a. die Unterart *Pyrgomorpha conica tereticornis (Brulle, 1840)*.

Eine weitere Art, *Pyrgomorpha granulata Stal, 1875* ist nicht der Gattung direkt, sondern einer ihrer Untergattungen, *Phymelloides*, zugeordnet. Jedes Potential Taxon dieser Untergattung muss zwingend der Gattung *Pyrgomorpha* zugeordnet sein. Würde eines ihrer Potential Taxa einer anderen Gattung, etwa *Protanita Kevan, 1962*, zugeordnet sein, entstünde ein Widerspruch in den Namen aller Arten, die der Untergattung unterstellt sind. Es wäre also nicht eindeutig, ob der Artname *Pyrgomorpha granulata* oder „*Protanita granulata*“ lautet.

Zu klären bleibt noch, wie Bi- und Trinomen in der Spalte *Name* von Tabelle *ZNAME* abgelegt werden sollen. In Frage kämen vollständige, zusammengesetzte Namen oder nur der für die jeweilige Stufe relevante Namensbestandteil (also für Arten nur das Epitheton ohne Gattung, für Unterarten nur die Unterart ohne die Art etc.). Beide

[7] Namen, die einen Rang unterhalb der Unterart belegen, bestehen dann sogar aus vier oder mehr Namensbestandteilen. Obwohl in der aktuellen Fassung der zoologischen Nomenklaturregeln (ICZN, 2000) solche Namen nicht mehr zulässig sind, existieren vor allem historische Sammlungsbelege, für die solche Namen berücksichtigt werden müssen.

Vorgehensweisen haben Vor- und Nachteile. Abgesehen davon, dass bei der Speicherung des kompletten Namens die erste Normalform verletzt wird, sind bei dieser Methode an den Stellen, an denen nur ein Namensbestandteil benötigt wird[8], komplexe Zeichenkettenoperationen erforderlich, um diesen Teil aus dem zusammengesetzten Namen zu extrahieren. Sie hat aber den Vorteil, dass bei der Ausgabe, bei der normalerweise der vollständige Name benötigt wird, dieser schon in einer Tabellenspalte vorliegt und nicht erst zeitaufwändig mittels mehrerer Abfragen aus der Hierarchie zusammengestellt werden muss. Vorgreifend auf die Schlussfolgerung in Abschnitt 4.1.6 sei hier angemerkt, dass keine der beiden Möglichkeiten hinsichtlich ihrer Konsequenzen auf die Implementierung einer Anwendung untersucht wurde. Als Kompromiss könnte die Einführung einer weiteren, redundanten Spalte, etwa *ZusNam*, in Tabelle *ZNAME* dienen, welche den zusammengesetzten Namen enthält. Die Tabellen 4.5 und 4.6 zeigen diesen Fall für die oben aufgeführte Beispielart (nur SysTax-Standardtaxonomie, $SoiId = 1$).

| TaxId | RangId | Name | Autor | ZusNam |
|---|---|---|---|---|
| 11 | 14 | Pyrgomorphinae | | |
| 12 | 15 | Pyrgomorphini | | |
| 13 | 16 | Protanita | Kevan, 1962 | |
| 14 | 16 | Pyrgomorpha | Serville, 1838 | |
| 15 | 20 | conica | (Olivier, 1791) | Pyrgomorpha conica (Olivier, 1791) |
| 16 | 21 | tereticornis | (Brulle, 1840) | Pyrgomorpha conica tereticornis (Brulle, 1840) |
| 17 | 17 | Phymelloides | | |
| 18 | 20 | granulata | Stal, 1875 | Pyrgomorpha granulata Stal, 1875 |

Tabelle 4.5: Beispiel für Tabelle *ZNAME*

| ZooId | TaxId | SoiId | VZooId |
|---|---|---|---|
| 130 | 11 | 1 | 109 |
| 131 | 12 | 1 | 130 |
| 132 | 13 | 1 | 131 |
| 133 | 14 | 1 | 131 |
| 134 | 15 | 1 | 133 |
| 135 | 16 | 1 | 134 |
| 136 | 17 | 1 | 133 |
| 137 | 18 | 1 | 136 |

Tabelle 4.6: Beispiel für Tabelle *PTAXON*

[8] etwa für Plausibilitätsprüfungen bei der Eingabe

### 4.1.3 Nachteile - Methode 1

Biologische Namen auf Unterstufenebene werden häufig zusammen mit ihrer jeweiligen Hauptstufe und eventuell dazwischen liegenden Unterstufen dargestellt. Dies ist nicht nur auf Artebene der Fall, sondern auch bei höheren Taxa, vor allem bei Gattungen und Familien. Aufgrund der hierarchischen Struktur ist für die Anzeige eines auf diese Weise zusammengesetzten Namens eine vorher nicht festgelegte Anzahl von Abfragen der Tabellen *PTAXON* und *ZNAME* notwendig, je nachdem, auf welcher Unterstufenebene sich der Name befindet und wie viele Unterstufen zwischen dem Namen und seiner Hauptstufe belegt sind. Für einen Namen auf der ersten Unterstufe ist jeweils genau eine Abfrage von *PTAXON* und *ZNAME* erforderlich, für die zweite Unterstufe entweder jeweils eine oder zwei Abfragen der Tabellen, auf der dritten Unterstufe entweder jeweils eine, zwei oder drei Abfragen, usw.

Eine Implementierung des Modells müsste hier entweder auf eine Schleife zurückgreifen, die so lange den nächsten Vorgänger heraussucht, bis die Hauptstufe erreicht ist, oder es müsste eine hierarchische SQL-Abfrage herangezogen werden. Beide Vorgehensweisen sind in Bezug auf die Abfragezeit nicht effizient. Bei der Darstellung eines einzelnen Namens mag das zwar nicht ins Gewicht fallen, soll jedoch eine Liste von Taxa ausgegeben werden oder sollen Taxa mittels eines regulären Ausdrucks über die zusammengesetzten Namen selektiert werden, so addieren sich die Verzögerungen bei den einzelnen Namen.

Eine Lösungsmöglichkeit zur Vermeidung langer Wartezeiten ist, analog der Vorgehensweise beim Artnamen, die Zwischenspeicherung des zusammengesetzten Namens in einer der beiden Tabellen. Aber im Gegensatz zum Artnamen handelt es sich bei den höheren Taxa nicht um eine grammatikalische Regel, sondern tatsächlich um eine Hierarchie, die durch die Potential Taxa bestimmt ist. Für ein Taxon kann es durchaus zwei oder noch mehr verschiedene zusammengesetzte Namen geben, je nachdem, welche Unterstufen zwischen seinen Potential Taxa und seiner Hauptstufe liegen. Eine Speicherung in Tabelle *ZNAME* ist folglich nicht möglich, sondern er muss als Eigenschaft des Potential Taxons in Tabelle *PTAXON* abgelegt werden. Damit nicht für Artnamen und Namen auf höheren Stufe unterschiedliche Tabellen abgefragt werden müssen, empfiehlt es sich, auch den zusammengesetzten Artnamen nicht in *ZNAME*, sondern ebenfalls in *PTAXON* zu verwalten.[9]

Zusätzlich zum zusammengesetzten Namen eines Taxons soll häufig auch sein Vorgänger auf der nächsthöheren Hauptstufe angezeigt werden. Um diesen zu ermitteln, ist ebenfalls eine vorher nicht bestimmte Anzahl rekursiver Abfragen der Tabelle *PTAXON* notwendig, die abhängig vom jeweiligen Rang des Taxons ist. Man könnte die Zahl der Abfragen auf eine reduzieren, wenn die *ZooId* des Vorgängers auf der nächsten Hauptstufe jedem Potential Taxon als zusätzliches Attribut angeführt mit, etwa in einer neuen Spalte namens *HVZooId*.

Das Zusammenwirken der hier vorgestellten Neuerungen zeigen die Tabellen 4.7 und

[9] Um die Integritätsbedingungen des Modells zu wahren, müssen diese zusammengesetzten Artnamen aller Potential Taxa einer einzelnen Art identisch sein.

4.8 anhand der Namen, die bereits in den vorangegangenen Beispieltabellen vorgestellt wurden.

| TaxId | RangId | Name | Autor |
|---|---|---|---|
| 1 | 1 | Animalia | |
| 2 | 2 | Metazoa | |
| 3 | 3 | Arthropoda | |
| 4 | 6 | Insecta | |
| 5 | 10 | Orthoptera | |
| 6 | 11 | Caelifera | |
| 7 | 12 | Pamphagoidea | |
| 8 | 12 | Acridoidea | |
| 9 | 13 | Pyrgomorphidae | |
| 10 | 13 | Lentulidae | |
| 11 | 14 | Pyrgomorphinae | |
| 12 | 15 | Pyrgomorphini | |
| 13 | 16 | Protanita | Kevan, 1962 |
| 14 | 16 | Pyrgomorpha | Serville, 1838 |
| 15 | 20 | conica | (Olivier, 1791) |
| 16 | 21 | tereticornis | (Brulle, 1840) |
| 17 | 17 | Phymelloides | |
| 18 | 20 | granulata | Stal, 1875 |

Tabelle 4.7: Beispiel für Tabelle *ZNAME*

Aufgrund der Integritätsbedingung des Modells, dass die Quelle des Vorgängers jedes Potential Taxons entweder die neutralen Quelle selbst oder mit der Quelle des jeweiligen Potential Taxon identisch sein muss, sind die beiden neuen Attribute *HVZooId* und *ZusNam* für jeden Eintrag in Tabelle *PTAXON* eindeutig bestimmt. Beide Spalten stellen eine Denormalisierung der Tabelle dar und können als solche zu Inkonsistenzen führen. In einem solchen Fall, in dem der Eintrag in einem der beiden Felder der durch die *VZooId* vorgegebenen Hierarchie widerspricht, sollte der Hierarchie der Vorzug gegeben und *HVZooId* bzw. *ZusNam* korrigiert werden, da diese beiden Werte lediglich aus der Hierarchie abgeleitete und somit von dieser abhängige Eigenschaften darstellen. Bei jeder Implementierung des Modells muss dafür Sorge getragen werden, dass solche Inkonsistenzen beim Erstellen und vor allem beim Ändern von Einträgen in Tabelle *PTAXON* nicht auftreten.

### 4.1.4 Namen - Methode 2

Die in den vorangegangenen Abschnitten vorgestellte Methode der Namensverwaltung betrachtet jeden Namen, egal welchen Ranges, als eigenständigen Eintrag in der Namenstabelle. Da biologische Namen, deren Rang nicht dem einer Hauptstufe entspricht, häufig zusammen mit den Namen ihrer Vorgänger als zusammengesetzte Namen dargestellt werden, ist es möglich, diese als elementar zu betrachten. Ein solcher Ansatz wird

| ZooId | TaxId | SoiId | VZooId | HVZooId | ZusNam |
|---|---|---|---|---|---|
| 101 | 1 | 1 | | | Animalia |
| 102 | 2 | 1 | 101 | | Animalia Metazoa |
| 103 | 3 | 1 | 102 | 102 | Arthropoda |
| 104 | 4 | 1 | 103 | 103 | Insecta |
| 105 | 5 | 1 | 104 | 104 | Orthoptera |
| 106 | 6 | 1 | 105 | 104 | Orthoptera Caelifera |
| 107 | 7 | 1 | 106 | 104 | Orthoptera Caelifera Pamphagoidea |
| 108 | 8 | 1 | 106 | 104 | Orthoptera Caelifera Acridoidea |
| 109 | 9 | 1 | 107 | 107 | Pyrgomorphidae |
| 110 | 10 | 1 | 108 | 108 | Lentulidae |
| 120 | 6 | 2 | 105 | 104 | Orthoptera Caelifera |
| 121 | 9 | 2 | 120 | 120 | Pyrgomorphidae |
| 122 | 10 | 2 | 120 | 120 | Lentulidae |
| 130 | 11 | 1 | 109 | 107 | Pyrgomorphidae subfam. Pyrgomorphinae |
| 131 | 12 | 1 | 130 | 107 | Pyrgomorphidae subfam. Pyrgomorphinae tr. Pyrgomorphini |
| 132 | 13 | 1 | 131 | 131 | Protanita Kevan, 1962 |
| 133 | 14 | 1 | 131 | 131 | Pyrgomorpha Serville, 1838 |
| 134 | 15 | 1 | 133 | 133 | Pyrgomorpha conica (Olivier, 1791) |
| 135 | 16 | 1 | 134 | 133 | Pyrgomorpha conica tereticornis (Brulle, 1840) |
| 136 | 17 | 1 | 133 | 131 | Pyrgomorpha subgen. Phymelloides |
| 137 | 18 | 1 | 136 | 136 | Pyrgomorpha granulata Stal, 1875 |

Tabelle 4.8: Beispiel für Tabelle *PTAXON*

schon von Boos (1992) vorgeschlagen und entspricht der Struktur, die in SysTax bereits in den für die Botanik konzipierten Programmteilen verwendet wird. Eine Zeile in der Tabelle der wissenschaftlichen Namen enthält hierbei den Namen der Hauptstufe mit Autorenzitat *und* den Namen der ersten Unterstufe, wiederum mit Autorenzitat, *und* den Namen der zweiten Unterstufe, etc. Trotz dass hier der zusammengesetzte Name als elementar betrachtet wird, ist es sinnvoll, die einzelnen Namensbestandteile in getrennten Tabellenfeldern zu verwalten, um im Bedarfsfall direkt auf sie zugreifen zu können. Die Anzahl der hierarchischen Stufen reduziert sich durch diese Form der Namensverwaltung von insgesamt 23 Rängen auf die sieben Hauptstufenränge. Abbildung 4.4 zeigt die Definition der Namenstabelle.

Spalte *TaxId* enthält den Primärschlüssel der Tabelle, in Spalte *HStufe* wird die Nummer der Hauptstufe abgelegt, auf der sich der jeweilige Name befindet. Diese Spalte bezieht sich auf die weiter oben vorgestellte Tabelle *RANG* (s. Tab. 4.1). $N_0$ beinhaltet den Namen der Hauptstufe, $A_0$ ihr Autorenzitat, Spalte $N_1$ den Namen der ersten Unterstufe usw. Für die Autorenzitate in den Spalten $A_0$ bis $A_4$ gelten dieselben Überlegungen wie für die Autoren in Abschnitt 4.1.2, wobei die Unterstruktur des Autorenzitats nicht weiter differenziert wird. Der vollständige, zusammengesetzte Namen könnte in einer weiteren Tabellenspalte, etwas *ZusNam*, abgelegt werden. Der Übersichtlichkeit wegen wird in den folgenden Beispielen auf die Darstellung dieser Spalte verzichtet.

| ZNAME |
|---|
| TaxId |
| HStufe |
| $N_0$ |
| $Autor_0$ |
| $N_1$ |
| $Autor_1$ |
| $N_2$ |
| $Autor_2$ |
| $N_3$ |
| $Autor_3$ |
| $N_4$ |
| $Autor_4$ |

Abbildung 4.4: Definition der Tabelle *ZNAME*, 2. Methode

Durch diese Form der Speicherung ergeben sich natürlich Mehrfachnennungen: die Hauptstufe ohne Unterstufen, die Hauptstufe mit erster Unterstufe, etc. Tabelle 4.9 zeigt, wie das Beispiel aus Tab. 4.2 in dieser Struktur aussehen könnte. Aus Gründen der Übersichtlichkeit werden die Autorenspalten ($A_0$, ..., $A_4$) weggelassen.

| TaxId | HStufe | $N_0$ | $N_1$ | $N_2$ | $N_3$ | $N_4$ |
|---|---|---|---|---|---|---|
| 1 | 1 | Animalia | | | | |
| 2 | 1 | Animalia | Metazoa | | | |
| 3 | 2 | Arthropoda | | | | |
| 4 | 3 | Insecta | | | | |
| 5 | 4 | Orthoptera | | | | |
| 6 | 4 | Orthoptera | Caelifera | | | |
| 7 | 4 | Orthoptera | Caelifera | | Pamphagoidea | |
| 8 | 4 | Orthoptera | Caelifera | | Acridoidea | |
| 9 | 5 | Pyrgomorphidae | | | | |
| 10 | 5 | Lentulidae | | | | |

Tabelle 4.9: Beispiel für Tabelle *ZNAME*

Als Folge dieser Methode der Namensverwaltung müssen weitere Integritätsbedingungen für Tabelle *ZNAME* eingehalten werden:

- Für alle $t \in ZNAME$ muss gelten: $t.N_0 \neq$ NULL.

- Für alle $t \in ZNAME$ mit $t.N_i \neq$ NULL und $t.N_{i+1} = \cdots = t.N_4 =$ NULL (für ein $i : 1 \leq i \leq 4$) muss genau ein $t' \in ZNAME$ existieren mit:

$$
\begin{aligned}
t'.HStufe &= t.HStufe, \\
t'.N_0 &= t.N_0, t'.A_0 = t.A_0, \\
&\dots \\
t'.N_{i-1} &= t.N_{i-1}, t'.A_{i-1} = t.A_{i-1}, \text{ und} \\
t'.N_i &= \cdots = t'.N_4 = \text{NULL}.
\end{aligned}
$$

Durch $t'$ ist damit der direkte Vorgänger des Namens $t$ auf derselben Hauptstufe definiert. Wenn also ein Eintrag mit $N_3 \neq$ NULL, $N_0$, $N_1$, $N_2$ beliebig, sowie $N_4 =$ NULL existiert, beispielsweise $TaxId = 7$ oder 8 in Tabelle 4.9, so muss es auch einen Eintrag mit denselben Werten in $N_0$, $N_1$, $N_2$ geben, für den $N_3 = N_4 =$ NULL gelten muss. In diesem Fall wäre dies die Zeile mit $TaxId = 6$.

Die Tabelle *PTAXON* der Potential Taxa hat bei dieser Form der Namensspeicherung dieselbe Gestalt wie in Abbildung 4.2. Im Gegensatz zur Vorgehensweise, die in den vorangegangenen Abschnitten beschrieben wurde, wird jetzt als Vorgänger eines Potential Taxons immer ein Potential Taxon auf einer höheren Ebene gewählt, selbst dann, wenn der Name tatsächlich auf Unterstufenebene liegt. Für das obige Beispiel bedeutet dies, dass der Vorgänger der Unterordnung *Orthoptera subord. Caelifera* ($TaxId = 6$) identisch ist mit dem Vorgänger der Ordnung *Orthoptera* ($TaxId = 5$) – zumindest innerhalb derselben Quelle. Mit den Vorgaben aus Abschnitt 4.1.2, dass die beiden Familien *Pyrgomorphidae* und *Lentulidae* in der Standardtaxonomie den Überfamilien *Pamphagoidea* bzw. *Acridoidea*, in einer weiteren Quelle jeweils der Unterordnung *Caelifera* zugeordnet sind, könnte *PTAXON* wie in Tabelle 4.10 aussehen.

| ZooId | TaxId | SoiId | VZooId |
|---|---|---|---|
| 101 | 1 | 1 | |
| 102 | 2 | 1 | |
| 103 | 3 | 1 | 102 |
| 104 | 4 | 1 | 103 |
| 105 | 5 | 1 | 104 |
| 106 | 6 | 1 | 104 |
| 107 | 7 | 1 | 104 |
| 108 | 8 | 1 | 104 |
| 109 | 9 | 1 | 107 |
| 110 | 10 | 1 | 108 |
| 120 | 6 | 2 | 104 |
| 121 | 9 | 2 | 120 |
| 122 | 10 | 2 | 120 |

Tabelle 4.10: Beispiel für Tabelle *PTAXON*

Es bleibt die Frage zu klären, ob unter der Voraussetzung, dass es für einen Name $t \in ZNAME$ auf Unterstufenebene ein Potential Taxon aus einer bestimmten Quelle gibt, auch für seinen direkten Vorgänger $t'$ (wie oben definiert) ein Potential Taxon mit derselben Quelle geben muss.

Zur Verdeutlichung dieser Problematik, wird die Namenstabelle aus obigen Beispiel um die Unterfamilie *Pyrgomorphinae* der Familie *Pyrgomorphidae* erweitert, wie in Tab. 4.11 gezeigt. Existiert nun zu diesem Eintrag ($TaxId = 11$) ein Potential Taxon der Quelle mit $SoiId = 3$, muss es dann ein Potential Taxon der Familie *Pyrgomorphidae* ($TaxId = 9$) mit $SoiId = 3$ geben? Ein solches Potential Taxon muss immer dann existieren, wenn der Vorgänger des „untergeordneten“ Potential Taxons, in diesem Fall von *Pyrgomorphidae subfam. Pyrgomorphinae*, nicht identisch ist mit dem Vorgänger seines Neutralen Potential Taxons.

Wenn also die *VZooId* eines Potential Taxons mit $(TaxId, SoiId)$=$(11, 3)$ identisch ist mit der *VZooId* des Potential Taxons mit $(TaxId, SoiId) = (11, 1)$, so braucht ein Potential Taxon mit $(TaxId, SoiId)$=$(9, 3)$ nicht existieren. Ist dies nicht der Fall, so ist die Existenz dieses Potential Taxons zwingend notwendig. Tabelle 4.12 zeigt eine in diesem Sinne zulässige Erweiterung von *PTAXON*.

| TaxId | HStufe | $N_0$ | $N_1$ | $N_2$ | $N_3$ | $N_4$ |
|---|---|---|---|---|---|---|
| 11 | 5 | Pyrgomorphidae | Pyrgomorphinae | | | |

Tabelle 4.11: Erweiterung von *ZNAME*

| ZooId | TaxId | SoiId | VZooId |
|---|---|---|---|
| 123 | 11 | 1 | 107 |
| 124 | 11 | 3 | 105 |
| 125 | 9 | 3 | 105 |
| 126 | 11 | 5 | 107 |

Tabelle 4.12: Erweiterung von *PTAXON*

Anders formuliert, die Einträge in Tabelle *PTAXON* müssen neben den Bedingungen, die sich aus dem Datenmodell ergeben, noch folgende Integritätsbedingungen einhalten:

Sei $t \in ZNAME$ ein Taxon auf Unterstufenebene sowie $t' \in ZNAME$ sein direkter Vorgänger wie weiter oben definiert, d. h. $t.N_i \neq \text{NULL}$ und $t.N_{i+1} = \cdots = t.N_4 = \text{NULL}$ (für ein $i : 1 \leq i \leq 4$), sowie

$$\begin{aligned} t'.HStufe &= t.HStufe, \\ t'.N_0 &= t.N_0, t'.A_0 = t.A_0, \\ &\cdots \\ t'.N_{i-1} &= t.N_{i-1}, t'.A_{i-1} = t.A_{i-1}, \text{ und} \\ t'.N_i &= \cdots = t'.N_4 = \text{NULL}. \end{aligned}$$

- Für jedes $pt, pt' \in PTAXON$ mit $pt.TaxId = t.TaxId$, $pt'.TaxId = t'.TaxId$ sowie $pt.SoiId = pt'.SoiId$ muss gelten:
  $pt.VZooId = pt'.VZooId$.
- Sei mit $npt \in PTAXON$: $npt.TaxId = t.TaxId$ und $ntp.SoiId = 1$ das Neutrale Potential Taxon des Namens $t$ bezeichnet. Existiert ein weiteres Potential Taxon $pt \in PTAXON \neq npt$ des Namens $t$ mit $pt.VZooId \neq npt.VZooId$, so muss ein Potential Taxon $pt'$ des Namens $t'$ existieren mit: $pt'.SoiId = pt.SoiId$.[10]

Für Namen auf Artebene ergibt sich aufgrund ihrer Eigenschaft als Binomen eine weitere Bedingung: Der Namensbestandteil $N_0$ der Vorgänger aller Potential Taxa eines Namens auf Artstufe muss identisch sein mir dem Gattungsnamen ihres zusammengesetzten Namens. Dies kann durch zwei weitere Integritätsbedingungen erreicht werden:

Seien zwei Namen auf Artebene $t, t' \in ZNAME$ wie oben definiert. Sei weiter mit $\mathcal{T}_v(t) \subset ZNAME$ die Menge der Namen der Vorgänger aller Potential Taxa des Namens $t$ bezeichnet:

$$\begin{aligned} \mathcal{T}_v(t) := \{t'' \in ZNAME \mid \exists pt, pt'' \in PTAXON\colon\ & pt''.TaxId = t''.TaxId, \\ & pt.TaxId = t.TaxId, \text{ sowie} \\ & pt''.ZooId = pt.VZooId\}. \end{aligned}$$

Dann müssen folgende Bedingungen erfüllt sein:

- Für alle $t_1, t_2 \in \mathcal{T}_v(t)$ muss gelten: $t_1.N_0 = t_2.N_0$ und $t_1.A_0 = t_2.A_0$.
- Für alle $t_1 \in \mathcal{T}_v(t)$, $t_2 \in \mathcal{T}_v(t')$ muss gelten: $t_1.N_0 = t_2.N_0$ und $t_1.A_0 = t_2.A_0$.

Tabellen 4.13 und 4.14 zeigen ein Beispiel für die Verwaltung von Artnamen in dieser Struktur, wobei die Spalten der Autorenzitate aus Gründen der Übersichtlichkeit weggelassen werden. Bei Namen auf Artebene enthält Spalte $N_0$ das Binomen, also Gattung und Epitheton.

| TaxId | HStufe | $N_0$ | $N_1$ | $N_2$ | $N_3$ | $N_4$ |
|---|---|---|---|---|---|---|
| 12 | 5 | Pyrgomorphidae | Pyrgomorphinae | Pyrgomorphini | | |
| 13 | 6 | Protanita | | | | |
| 14 | 6 | Pyrgomorpha | | | | |
| 15 | 7 | Pyrgomorpha conica | | | | |
| 16 | 7 | Pyrgomorpha conica | tereticornis | | | |
| 17 | 6 | Pyrgomorpha | Phymelloides | | | |
| 18 | 7 | Pyrgomorpha granulata | | | | |

Tabelle 4.13: Beispiel für Tabelle *ZNAME*

| ZooId | TaxId | SoiId | VZooId |
|---|---|---|---|
| 131 | 12 | 1 | 107 |
| 132 | 13 | 1 | 131 |
| 133 | 14 | 1 | 131 |
| 134 | 15 | 1 | 133 |
| 135 | 16 | 1 | 133 |
| 136 | 17 | 1 | 131 |
| 137 | 18 | 1 | 136 |

Tabelle 4.14: Beispiel für Tabelle *PTAXON*

Ein Potential Taxon beispielsweise, welches die Unterart *Pyrgomorpha conica tereticornis* ($TaxId = 16$) einem Taxon auf Gattungsebene zuweist, dessen Namensbestandteil $N_0$ nicht identisch ist mit „Pyrgomorpha“, etwa mit $TaxId = 13$, ist nach den Integritätsbedingungen nicht zulässig.

### 4.1.5 Nachteile - Methode 2

Ein Nachteil dieser Methode ist die Entstehung von Redundanzen und Mehrfacheinträge von Namen. Wenn es einen Namen mit $N_1 \neq$ NULL gibt, so muss es auch einen in $N_0$ und $A_0$ identischen Eintrag mit $N_1 =$ NULL geben, und beide müssen innerhalb derselben Quelle demselben Vorgänger zugewiesen sein. Die sich daraus ergebenden möglichen Inkonsistenzen des Datenbestandes lassen sich nur durch die strikte Einhaltung weiterer Integritätsbedingungen vermeiden, die über die vom Modell vorgegebenen Bedingungen hinausgehen.

Durch die „Ausflachung“ der Hierarchie ergibt sich ein zweiter Nachteil: Der Vorgänger eines Namens auf derselben Hauptstufe wird nicht, wie in der ersten Methode[11], durch einen Schlüssel festgelegt, sondern muss durch Wertevergleich von Spalten ermittelt werden:

[10] Zusammen mit der ersten Bedingung ergibt sich für diese Potential Taxa: $pt'.VZooId = pt.VZooId$.

[11] vgl. Abschnitt 4.1.2

Sei etwa $t \in ZNAME$ ein Name mit $t.N_3 \neq \text{NULL}$ und $t.N_4 = \text{NULL}$. Dann wird sein unmittelbarer Vorgänger $t'$ bestimmt durch: $t'.N_0 = t.N_0$, $t'.A_0 = t.A_0$, $t'.N_1 = t.N_1$, $t'.A_1 = t.A_1$, $t'.N_2 = t.N_2$, $t'.A_2 = t.A_2$ und $t'.N_3 = t'.N_4 = \text{NULL}$.

Natürlich könnte die *TaxId* des Namens des Vorgängers innerhalb einer Hauptstufe in Tabelle *ZNAME* redundant mitgeführt werden. Da diese Information aber nur an wenigen Stellen benötigt wird, ist dieser Aufwand nicht lohnenswert.

### 4.1.6 Konklusion

Bei der zweiten Methode steht der zusammengesetzte Name sofort zur Verfügung, genauso wie der Vorgänger auf der nächsthöheren Hauptstufe. Beide Werte müssen in der ersten Methode erst umständlich ermittelt werden. Möchte man dieses für die erste Methode durch Einführung weiterer, redundanter Spalten ändern, so geschieht dies auf Kosten des großen Vorteils dieser Methode, der Redundanzfreiheit. Damit müssten auch bei der ersten Methode weitere Integritätsbedingungen eingehalten werden, um die Konsistenz des Datenbestandes zu gewährleisten. Der Aufwand bei beiden Methoden wäre dann in etwa gleich.

Die modifizierte erste Methode bietet gegenüber der zweiten Methode dennoch einen Vorteil, indem sie die taxonomische Hierarchie als Kette von Schlüsselwerten verwaltet, die bei etwaigen Inkonsistenzen als „oberste Instanz“ herangezogen werden kann, um die redundanten Spaltenwerte neu zu generieren.

Aufgrund der bereits vorliegenden Erfahrungswerte wurde im SysTax-Projekt die zweite Methode bevorzugt, die den im botanischen Programmteil verwendeten Ansatz aufgreift. Zusätzlich konnten Routinen, insbesondere Prüfprogramme, aus der Botanik übernommen werden, womit die Entwicklungszeit entschieden verkürzt werden konnte. Da bei der Erweiterung von SysTax auf die Zoologie terminliche Vorgaben in Bezug auf die Fertigstellung der einzelnen Programmteile gemacht wurden, war dies der maßgebliche Vorteil der zweiten Methode.

## 4.2 Realisierung des SysTax-Modells – Synonymie

Die Umsetzung der Synonymieverwaltung kann unabhängig von der Methode der Namensspeicherung betrachtet werden, da ausschließlich Potential Taxa miteinander synonymisiert werden können.

Solange die Namen von Potential Taxa, die in einer Synonymiebeziehung miteinander verknüpft sind, auf derselben Hauptstufe liegen, ist der tatsächliche Rang des Namens ohne Belang, genauso wie die Vorgänger der Potential Taxa in der taxonomischen Hierarchie. Wenn also in diesem Abschnitt von einem „Vorgänger“ die Rede ist, so ist immer der Vorgänger im Baum der Synonymieverknüpfungen, nicht aber der taxonomische Vorgänger des Potential Taxons gemeint. Aus diesem Grund wird auch in den folgenden Beispielen die Spalte *VZooId* der Tabelle *PTAXON* weggelassen.

Abbildung 4.5 zeigt nochmals zur Wiederholung die Definition der Tabelle *PTAXON*. Wie beschrieben, beinhaltet Spalte *StatusId* den Schlüssel einer Tabelle, in welcher die Namen aller Status gespeichert sind, die die Potential Taxa annehmen können. Für die Darstellung der Synonymieverwaltung ist lediglich die Unterscheidung relevant, ob ein Name in einer Quelle akzeptiert, also „gültig“, oder nicht akzeptiert bzw. „ungültig“ ist. Im Folgenden wird der Status eines Potential Taxons nicht weiter differenziert und in den Beispielen in Spalte *StatusId* der Wert *'yes'* angegeben für Potential Taxa, die in ihrer Quelle akzeptiert sind, und *'no'* für diejenigen Potential Taxa, die in ihrer Quelle als ungültig deklariert werden.

| PTAXON |
|---|
| ZooId |
| TaxId |
| SoiId |
| VZooId |
| StatusId |

Abbildung 4.5: Definition der Tabelle *PTAXON*

Eine Umsetzung der Relation $\mathcal{Z}$ der Synonymieverknüpfungen aus dem SysTax-Modell (Abschnitt 3.2.3) könnte wie in Abbildung 4.6 gezeigt aussehen. Primärschlüssel ist die Spalte *SynoId*. Der Schlüssel des „ungültigen“ Potential Taxons einer Synonymieverknüpfung wird in Spalte *ZooId* abgelegt und der Schlüssel seines Vorgängers in Spalte *GZooId*. Die Bezeichnung für die der verschiedenen Typen von Synonymiebeziehungen (z. B. subjektives, objektives Synonym oder Umkombination für die reguläre Synonymie; Kongruenz, Inklusion, usw. für Konzeptsynonyme) sind in einer weiteren, hier nicht aufgeführten Tabelle abgelegt, deren Schlüssel *SynoArtId* als Realisierung des Attributs $\mathcal{A}$ im SysTax-Modell der Tabelle *ZSYNO* mitgegeben wird. Da es zur Veranschaulichung der Synonymieverwaltung lediglich relevant ist, ob es sich um ein „reguläres“ Synonym oder um ein Konzeptsynonym handelt, wird im Folgenden die Spalte *SynoArtId* den Wert *'R'* für reguläre Synonyme und *'K'* für Konzeptsynonyme annehmen. Wie schon im Abschnitt 3.2.3 gezeigt wurde, könnte eine Spalte *SoiId*, welche den Schlüssel der Quelle der Synonymiebeziehung enthält, auch weggelassen werden, der Vollständigkeit halber wird sie aber hier und in den weiteren Beispielen mitgeführt.

| ZSYNO |
|---|
| SynoId |
| GZooId |
| ZooId |
| SynoArtId |
| SoiId |

Abbildung 4.6: Definition der Tabelle *ZSYNO*

Abbildung 4.7 zeigt ein erstes Beispiel für reguläre Synonyme. In der Praxis werden solch gegensätzliche Synonymiebäume nur selten vorkommen, sind aber theoretisch möglich. Aus diesem Grund werden sie hier dargestellt, es muss aber mit fiktiven Taxa und Quellen gearbeitet werden.

Quelle X (*SoiId* 10):

*Taxon A*
- *Taxon C*
  - *Taxon D*
- *Taxon B*

Quelle Y (*SoiId* 20):

*Taxon B*
- *Taxon A*
- *Taxon D*

Abbildung 4.7: Beispiel für reguläre Synonymie

Tabellen 4.15 und 4.16 zeigen, wie diese beiden Synonymiebäume in Tabellen umgesetzt werden können. Als *TaxId* wird hier keine Zahl, sondern ein Kürzel des „Taxonnamens“ verwendet.

| ZooId | TaxId | SoiId | StatusId |
|---|---|---|---|
| 1 | TaxA | 1 | yes |
| 2 | TaxB | 1 | yes |
| 3 | TaxC | 1 | yes |
| 4 | TaxD | 1 | yes |
| 5 | TaxA | 10 | yes |
| 6 | TaxA | 20 | no |
| 7 | TaxB | 10 | no |
| 8 | TaxB | 20 | yes |
| 9 | TaxC | 10 | no |
| 10 | TaxD | 10 | no |
| 11 | TaxD | 20 | no |

Tabelle 4.15: Beispiel für Tabelle *PTAXON*

| SynoId | GZooId | ZooId | SynoArtId | SoiId |
|---|---|---|---|---|
| 1 | 5 | 9 | R | 10 |
| 2 | 9 | 10 | R | 10 |
| 3 | 5 | 7 | R | 10 |
| 4 | 8 | 6 | R | 20 |
| 5 | 8 | 11 | R | 20 |

Tabelle 4.16: Beispiel für Tabelle *ZSYNO*

Häufig sollen alle Synonyme eines Potential Taxons ausgegeben werden. Ist dieses selbst Synonym, so muss zuerst das höchste, gültige Potential Taxon der Hierarchie gefunden werden, um dann alle seine Synonyme und rekursiv deren Synonyme ausgeben zu können. Um im obigen Beispiel ausgehend von *Taxon D sec. X* das gültige Potential Taxon zu ermitteln, würde zuerst sein Vorgänger *Taxon C sec. X* und dann dessen Vorgänger *Taxon A sec. X* gefunden werden. Anschließend muss getestet werden, ob dieses Potential Taxon nicht ebenfalls in Spalte *ZooId* der Tabelle *ZSYNO* (mit *SynoArtId 'R'*) vorkommt und somit selber noch Synonym ist. Alternativ müsste für jeden Synonymvorgänger in Tabelle *PTAXON* seine *StatusId* geprüft werden. Entspricht sie dem Wert *'yes'*, so hat man das „oberste" gültige Potential Taxon gefunden, andernfalls muss die Suche fortgesetzt werden. Dies führt zu einer Verdoppelung der Anzahl der Abfragen, was ebenfalls, bei entsprechender Tiefe des Synonymbaumes, zu Wartezeiten bei der Ausgabe führen kann.

In einer relationalen Datenbank kann unter Umständen die Überprüfung, ob ein Wert Element einer Menge ist, schneller sein als der Test, ob eine Menge einen bestimmten Wert *nicht* enthält. Folglich bietet es sich an, das oberste Potential Taxon einer Synonymiehierarchie ebenfalls in Tabelle *ZSYNO* zu speichern, wobei sein Schlüssel in Spalte *ZooId* abgelegt wird und *GZooId* leer bleibt. Tabelle 4.17 zeigt eine diesbezügliche Erweiterung der Tabelle 4.16.

| SynoId | GZooId | ZooId | SynoArtId | SoiId |
|---|---|---|---|---|
| 6 | | 5 | R | 10 |
| 7 | | 8 | R | 20 |

Tabelle 4.17: Erweiterung der Tabelle 4.16

Diese Vorgehensweise stellt zwar einen Bruch in der Logik des Modells bzw. dessen Umsetzung dar, da in Spalte *ZooId* ausschließlich die *ZooId* des „ungültigen" Potential Taxons stehen sollte. Die Suche nach dem obersten, gültigen Potential Taxon kann dadurch aber mittels einer einfachen Rekursion oder Iteration auf der Tabelle *ZSYNO* erfolgen, mit der Abbruchbedingung *GZooId* = NULL.

Bei der regulären Synonymie werden stets Potential Taxa aus derselben Quelle miteinander verknüpft, wie auch im obigen Beispiel zu erkennen ist. Bei der Konzeptsynonymie hingegen werden die Konzepte der Potential Taxa verschiedener Quellen miteinander verglichen. Behauptet beispielsweise Quelle *X*, ihr *Taxon A* sei kongruent zu *Taxon A* aus Quelle *Y* und interferiere mit *Taxon D sec. Y*, so zeigt Tabelle 4.18, wie dies in Tabellenform aussehen könnte.[12]

Bei der Konzeptsynonymie handelt es sich im Grunde nicht, wie bei der regulären Synonymie, um eine gerichtete Verknüpfung zwischen zwei Potential Taxa, sondern um eine

[12] Wobei hier, wie oben schon erwähnt, keine Unterscheidung zwischen den verschiedenen Arten der Konzeptsynoymie gemacht werden soll.

| SynoId | GZooId | ZooId | SynoArtId | SoiId |
|---|---|---|---|---|
| 8 | 5 | 6 | K | 10 |
| 9 | 5 | 11 | K | 10 |

Tabelle 4.18: Beispiel für Konzeptsynonyme in Tabelle *ZSYNO*

ungerichtete Verknüpfung. Von der Logik her müsste eine solche Verknüpfung kommutativ sein: Wenn *Taxon A sec. X* kongruent zu *Taxon A sec. Y* ist, so ist auch *Taxon A sec. Y* kongruent zu *Taxon A sec. X*. Es sollte aber nur das gespeichert werden, was die jeweilige Quelle aussagt. Auch wenn sich diese Regel eher an den Benutzer richtet, der Daten eingibt, muss sie auch von Implementationen des Modells berücksichtigt werden, denn es sollte keine automatische Erzeugung von Zeilen mit umgedrehter *GZooId* und *ZooId* erfolgen. In diesem Beispiel dürften die Zeilen

| SynoId | GZooId | ZooId | SynoArtId | SoiId |
|---|---|---|---|---|
| 10 | 6 | 5 | K | 20 |
| 11 | 11 | 5 | K | 20 |

nicht automatisch beim Einfügen der Zeilen mit *SynoId* 8 und 9 (Tab. 4.18) generiert werden.

Genauso wie bei der Namensverwaltung und der Verwaltung der Potential Taxa gilt auch hier, dass für jedes Programm, welches Einträge in Tabelle *ZSYNO* einfügen, editieren oder löschen darf, sichergestellt werden muss, dass die Regeln des Modells aus Abschnitt 3.2.3 jederzeit eingehalten werden.

# 5 Multimedia-Daten und Informationscontainer

## 5.1 Einleitung

Mit dem Fortschritt in der Computertechnologie sind nicht nur die Möglichkeiten von Datenbankmanagementsystemen gestiegen, sondern auch die Anforderungen der Benutzer an diese. Waren zu den Anfangszeiten von SysTax vielfach noch reine Textanwendungen in Gebrauch und Abfrageergebnisse in Listenform ausreichend, so ist heute aus Gründen der besseren Vermarktung zusätzlich zur Funktionalität eines Programms eine optisch ansprechende Präsentation der Daten nicht nur erwünscht, sondern sogar erforderlich. Dabei spielen multimediale Inhalte eine große Rolle. So wollen z. B. viele Institute neben der Verwaltung ihrer Sammlungsobjekte in Datenbanken diese auch für ein breites Publikum im Internet verfügbar machen. Hierzu gehören neben reinen Inventarlisten und Detailinformationen zu den einzelnen Belegen auch Fotografien der Objekte (möglicherweise sogar aus verschiedenen Blickwinkeln oder in verschiedenen Vergrößerungen) oder Scans der Objektetiketten. Außerdem können Multimediadaten zu zoologischen Sammlungsbelegen auch in Form von Tonaufnahmen vorliegen, etwa der Gesang eines Vogels oder das Zirpen von Grillen.

Eine der Anforderungen an SysTax war folglich, dass die Speicherung multimediale Inhalte und deren Zuordnung zu Taxa und Sammlungsbelegen möglich sein sollte. Diese Aussage musste natürlich noch konkretisiert und ein Anforderungskatalog erstellt werden, bevor mit der Erstellung eines Datenbankmodells begonnen werden konnte.

Zur Definition des Begriffs „Multimedia“ schreibt die Internet-Enzyklopädie „Wikipedia“[1]:

*„Multimedia is the use of several different media to convey information (text, audio, graphics, animation, video, and interactivity). Multimedia also refers to computer media. As the information is presented in various formats, multimedia enhances user experience and helps grasping information better and faster. Presenting information in various formats is nothing new to human beings, but multimedia generally implies presenting information in various digital formats. Although it is also used in visual arts to describe works created using more than one medium. [...]“*

Damit wird unter dem Ausdruck „Multimedia“ die Darstellung und Verknüpfung von Informationen unterschiedlicher Formate verstanden.

[1] http://en.wikipedia.org/wiki/Main_Page

Natürlich ist nicht nur die reine Speicherung von multimedialen Inhalten interessant. Genauso wichtig ist auch die Zuordnung dieser Inhalte zu anderen Datensätzen. Für eine biologische Datenbank wie SysTax bedeutet dies, dass z. B. eine oder mehrere Pflanzen und/oder Tiere auf einem Foto abgebildet sein können, auf Tonaufnahmen können ebenfalls einzelne oder auch mehrere Individuen zu hören sein. Ganze Bücher mit Erstbeschreibungen können als gescannte Abbildungen vorliegen. Dabei können sich mehrere Seiten auf ein Taxon oder auch eine Seite auf mehrere Taxa beziehen.

Konkret soll SysTax den folgenden Anforderungen genügen:

- Speicherung von Bildern, Tondateien, Texten und anderen binären Daten (z. B. PDF-Dokumente, Videos)
- Speicherung von spezifischen Parametern entsprechend den Datentypen (z. B. Auflösung in Pixel bei Bildern oder Länge in Sekunden bei Tondateien) und von allgemeinen Parametern (wie z. B. Dateiname oder Copyright-Vermerke)
- Bei Bildern sollen nicht nur die möglicherweise großformatigen Originale gespeichert werden können, sondern auch für die Internetanzeige optimierte Versionen und kleine „Thumbnails“.
- Multimediaobjekte (insbes. Bilder) können als Quellenangabe ein Literaturzitat besitzen.
- Multimediale Daten sollen einem oder mehreren Datensätzen aus folgenden Gebieten gleichzeitig zugeordnet werden können:
  - einem oder mehreren botanischen Taxa
  - einem oder mehreren zoologischen Taxa
  - einem oder mehreren Sammlungsbelegen und/oder
  - einer oder mehreren Akzessionen botanischer Gärten
- Datensätzen aus den oben genannten Gebieten sollen mehrere Multimediadaten zugeordnet werden können.
- Eine oder mehrere solcher Verknüpfungen sollen zu einer „Informationseinheit“ zusammengefasst und mit dieser Einheit sollen Merkmale in Form von hierarchisch gegliederten Stichworten (hierarchischer Thesaurus) zugeordnet werden können.
- Es soll die Möglichkeit bestehend, zu vorhandenen „Informationseinheiten“ auf möglichst einfache Weise weitere Datensätze hinzuzufügen.

## 5.2 Modellierung und Realisierung

Ausgehend von diesen Anforderungen kann nun ein Datenbankmodell erstellt werden. Dieses Modell kann in zwei Teile gegliedert werden: die Verwaltung der Multimediaobjekte und die Verwaltung der Verknüpfungen dieser Objekte mit anderen Datensätzen.

### 5.2.1 Verwaltung der Multimediaobjekte

Aus den ersten beiden Bedingungen des Anforderungskatalog ergibt sich, dass Multimediaobjekte zwei Qualitäten von Attributen besitzen können: Attribute, die jedem Objekt gemein sind und solche, die vom jeweiligen Objekttyp abhängen.

Zu Ersteren gehören unter anderem Informationen wie Titel, Beschreibung, Bemerkung, Autor, Copyright, Format bzw. MIME content type[2], Größe (in Bytes), Lese- und Schreibberechtigungen sowie eine Quellenangabe. Diese lässt sich weiter untergliedern: Als Quelle kann eine Person, ein Institut, eine URL oder eine Literaturstelle dienen.

Bilder und Tondateien besitzen zusätzlich noch spezifische Attribute. Bei einem Bildobjekt wären dies die Auflösung in Pixel, der Bildtyp[3], eine web-optimierte, kleinere Version und ein Thumbnail. Zu Tonobjekten gehören Informationen wie z. B. die Länge der Aufnahme, der verwendete Rekorder sowie Mikrofon, Bassfilter, Licht- und Temperaturverhältnisse zum Zeitpunkt der Aufnahme[4] usw.

Bei einer solchen Konstellation bietet es sich an, die Objekte als eine Entität (hier mit *OBJEKT* bezeichnet) zu modellieren, welche die allgemeingültigen Parameter als Attribute besitzt. Die einzelnen Objekttypen mit ihren spezifischen Parametern inklusive der eigentlichen binären Objektdaten werden als separate Entitäten dargestellt, die mittels einer „is_a“-Beziehung mit der allgemeinen Entität verknüpft sind. Damit steht die Entität *OBJEKT* stellvertretend für alle Binärobjekte für Relationen mit anderen Entitäten zur Verfügung. Das entsprechende Entity-Relationship-Diagramm zeigt Abbildung 5.1.

Die spezifischen Parameter der Bild- und Tondateien werden durch die beiden Entitäten *Bild* und *Ton* repräsentiert, die eigentlichen binären Daten (inklusive der „Thumbnails“ der Bilder) durch die beiden Entitäten *Bild-Binärdaten* und *Ton-Binärdaten*. Die Entität *Text* modelliert die Textdateien und die Entität *andere Binärdaten* repräsentiert ganz allgemein binäre Daten, die nicht in die Kategorien „Bild“ und „Ton“ passen.

Bei einer Realisierung des Modells kann somit jedes Binärobjekt unabhängig von seinem Typ mittels eines eindeutigen Schlüssels (Primärschlüssel der Tabelle, die die Entität *OBJEKT* umsetzt) identifiziert und angesprochen werden.

### 5.2.2 Verwaltung der Verknüpfungen

An dieser Stelle soll noch einmal der Anforderungskatalog aus Abschnitt 5.1 kurz wiederholt werden: Ein oder mehrere Multimediaobjekte, ein oder mehrere botanische und/oder

[2] MIME steht für Multipurpose Internet Mail Extensions; ein Protokoll zur Unterstützung des Transfers von Nicht-Text-Dokumenten in textbasierten Protokollen wie E-Mail oder HTTP. Der „content type“ spezifiziert den genauen Typ der versendeten Daten [http://de.wikipedia.org (Dezember 2003)].

[3] z. B. Dia, Mikroskopaufnahme oder eine eingescannte Abbildung

[4] da diese Einfluss auf die Tiergeräusche haben können

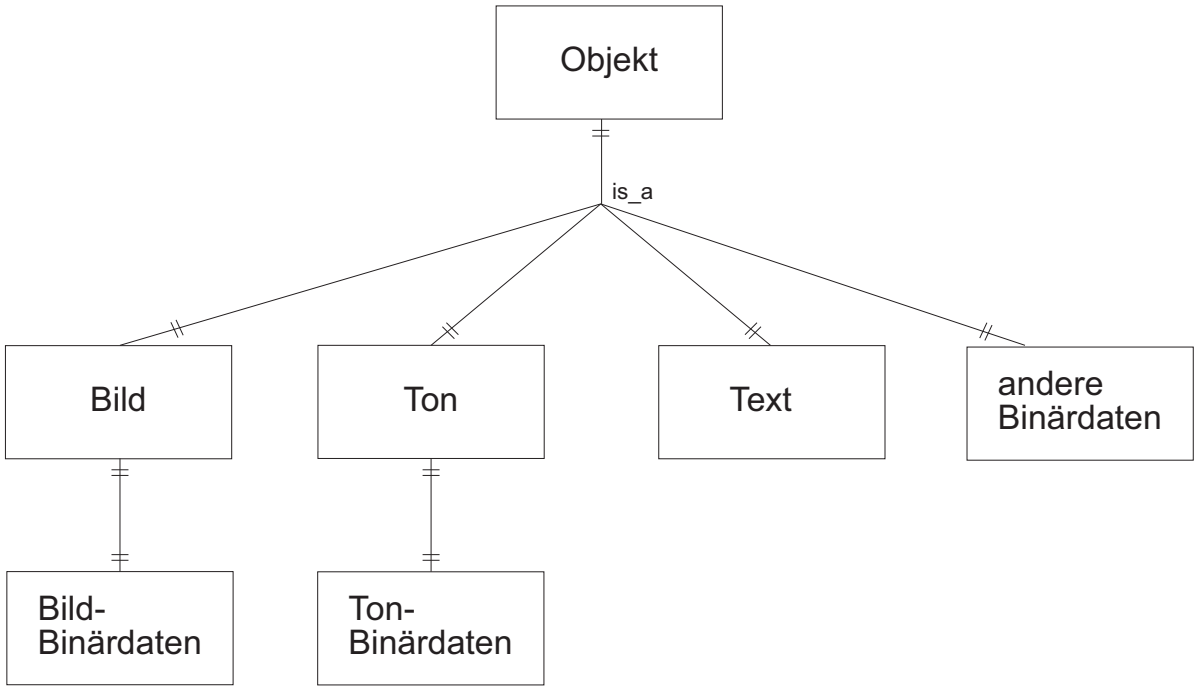


Abbildung 5.1: ER-Diagramm zur Objektverwaltung

zoologische Taxa und/oder Sammlungsbelege und/oder Gartenakzessionen sollen einander zugeordnet werden, sodass auf möglichst einfache Weise weitere Datensätze einer solchen Verknüpfung hinzugefügt werden können.

Eine solche Relation kann als eine Art „Informationscontainer“ angesehen werden, in dem alle Datensätze zusammengefasst werden, die thematisch zusammengehören. Beispielsweise könnte ein solcher „Container“ ein Bild von einer Biene bei der Blütenbestäubung, ein Text zur Erklärung dieses Bildes, die Taxonnamen des abgebildeten Insektes und der Pflanze sowie Stichworte zur Bestäubung beinhalten. Die Reihenfolge, in der die einzelnen Datensätze diesem „Container“ hinzugefügt werden, spielt dabei keine Rolle. So kann es zum Beispiel durchaus vorkommen, dass sich nur zwei verschiedene Taxa in einem solchen „Container“ befinden und ein Multimediaobjekt - wenn überhaupt - erst später hinzugefügt wird.

Die wohl am häufigsten vorkommende Verknüpfungsart ist eine c-mc-Beziehung zwischen den Multimediaobjekten und den Taxa bzw. den Sammlungsbelegen oder Gartenakzessionen; d. h. dass ein botanisches oder zoologisches Taxon oder ein Beleg ein oder mehrere Objekte besitzt. Folglich wäre es sinnvoll, die Objekten direkt mit den Taxa bzw. Belegen in Relation zu bringen. Die oben erwähnten „Informationscontainer“ könnten dann als eigene Entität modelliert werden, die mit den anderen beteiligten Entitäten in Beziehung steht. Dies wäre für die reine Zuordnung von Objekten zu Taxa und Belegen die Lösung, die die wenigsten Benutzereingaben erfordern würde.

Abbildung 5.2 zeigt, wie ein entsprechendes Datenmodell aussehen könnte. Die vier

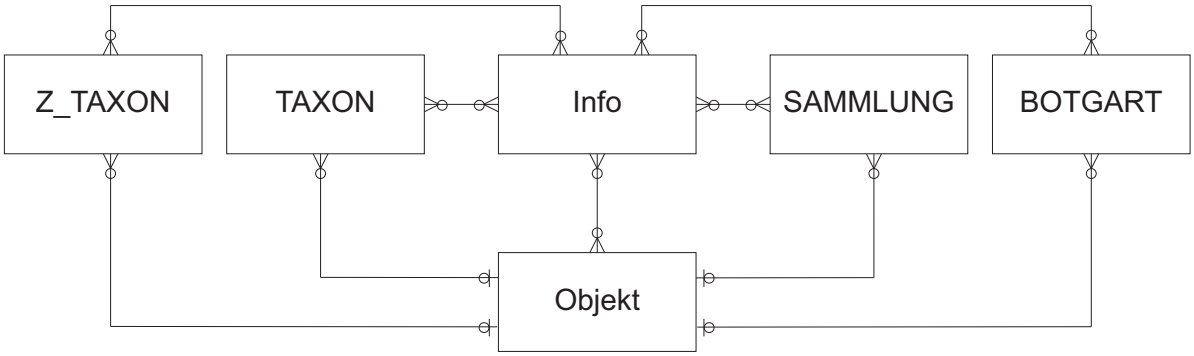


Abbildung 5.2: mögliches ER-Modell für die Verknüpfungsverwaltung

„Entitäten“ *Z_TAXON* (zoologische Taxonomie), *B_TAXON*, (botanische Taxonomie), *SAMMLUNG* (Sammlungsbelege) und *BOTGART* (Botanischer Garten) stehen hier als Platzhalter für die einzelnen Bereiche und sind nicht als Entität im eigentlichen Sinne zu verstehen.[5] Zusätzlich müsste noch die Integritätsbedingung erfüllt sein, dass jedes Element der Entität *Objekt* nur an einer der fünf möglichen Relationen beteiligt sein darf, also dass es entweder nur einem zoologischen, einem botanischen Taxon, einem Sammlungsbeleg, einer Gartenakzession *oder* einem oder mehreren „Informationscontainern“ zugeordnet sein darf.

Dieses Modell bringt aber in seiner Umsetzung einige Probleme mit sich. So ist es nicht ohne weiteres möglich, zu einer schon bestehenden Verknüpfung zwischen Objekt und Taxon (bzw. zwischen Objekt und Beleg) weitere Taxa (bzw. Belege) hinzuzufügen. Hierbei müsste die vorhandene Beziehung aus der Relationstabelle gelöscht und in den neu zu erstellenden „Informationscontainer“ eingefügt werden. Erst dann kann das zweite Taxon (bzw. der zweite Beleg) hinzugefügt werden.

Es stellt sich weiter die Frage, was mit dem „Informationscontainer“ geschehen soll, wenn beispielsweise eine von zwei Verknüpfungen zu Taxa gelöscht wird. Der „Container“ könnte entweder gelöscht und die Multimediaobjekte in die Relation mit dem verbleibenden Taxon gestellt werden, oder er könnte bestehen bleiben und damit zu einer zweiten, parallelen Realisierung der c-mc-Beziehung zwischen den Multimediaobjekten und den Taxa werden. Beides kann zur Verwirrung der Benutzer beitragen, die ihre Daten dann möglicherweise an zwei verschiedenen Stellen zu suchen haben. Sicherlich lassen sich einige dieser Probleme umgehen, indem in einer Implementierung des Modells entsprechende Mechanismen zur Benutzerführung und Automatisierung geschaffen werden.

Für SysTax wurde jedoch ein anderer Modellansatz gewählt: Die „Informationscontainer“ werden als zentrale Entität modelliert, die mit den anderen Bereichen (botanische und zoologische Taxonomie, Sammlungsbelege, Gartenakzessionen und Objekte) jeweils

[5] Tatsächlich ist die „Entität“ *Z_TAXON* sogar identisch mit der Entität $\mathcal{PT}$ aus Abschnitt 3.2.3, da, wie bereits aus Kapitel 2 bekannt, Informationen nur mit Potential Taxa verknüpft werden dürfen.

in einer mc-mc-Relation verknüpft ist. Die direkten c-mc-Relationen zwischen den Objekten und der Taxonomie bzw. den Belegen werden hier weggelassen. Abbildung 5.3 zeigt das Entity-Relationship-Diagramm dieser Variante. Analog zum vorangegangenem Modell stehen auch hier die „Entitäten“ *Z_TAXON*, *B_TAXON*, *SAMMLUNG* und *BOTGART* als Platzhalter für die einzelnen Bereiche.

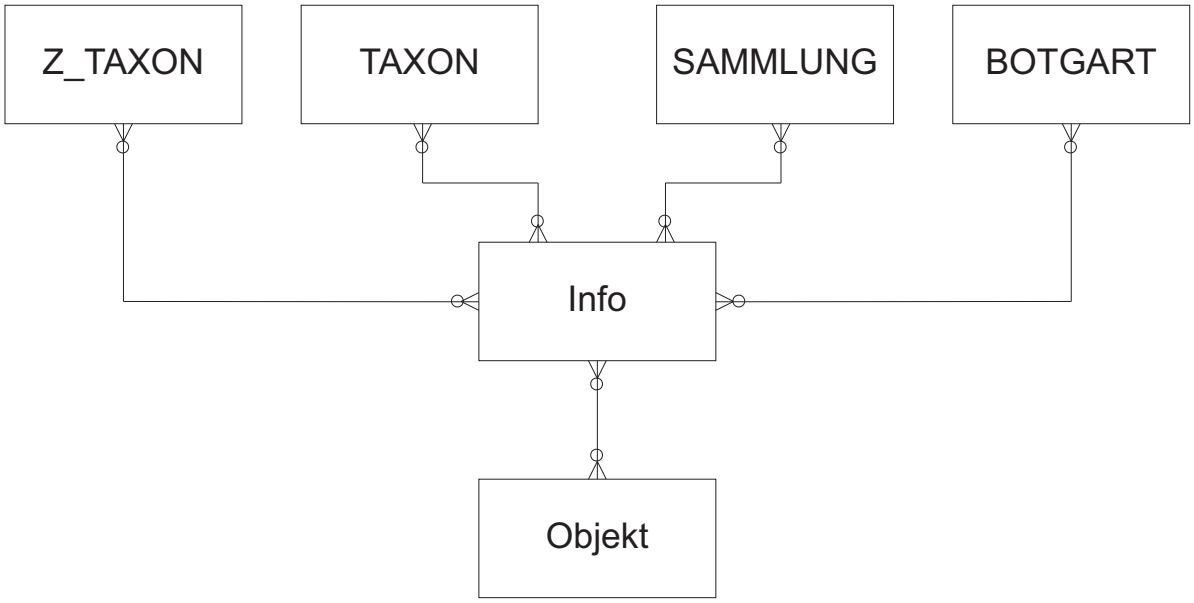


Abbildung 5.3: ER-Diagramm für die „Informationscontainer“

Als Konsequenz daraus wird nicht nur die multiple Zuordnung von Taxa, Belegen und Objekten mittelbar über einen „Informationscontainer“ modelliert, sondern dies gilt für *jede* Zuordnung, also auch dann, wenn z. B. ein Beleg nur ein einziges Objekt besitzt.

Bei einer Realisierung dieses Modells muss nicht nur ein einziger Eintrag in einer Tabelle erstellt werden wie bei einer direkten c-mc-Relation zwischen Objekt und Taxon bzw. Beleg, sondern es muss jeweils ein Eintrag in drei verschiedenen Tabellen vorhanden sein. Soll beispielsweise ein Bild einem Sammlungsbeleg zugeordnet werden, so ist sowohl der „Container“ zu erstellen, also ein Eintrag in Tabelle *INFO*, als auch jeweils ein Eintrag in den Tabellen *INFO_OBJ*[6] und *INFO_HERB*[7]. Der Mehraufwand bei der Eingabe wird durch automatisiertes Erstellen von Einträgen und geeignete Benutzerführung in den SysTax-Eingabemasken so gering wie möglich gehalten.

Natürlich ergeben sich für den Benutzer einer entsprechenden Anwendung mehrere Möglichkeiten der Verwaltung seiner Taxa/Belege und ihrer Multimediaobjekte. So könnte beispielsweise jede Verknüpfung zwischen Beleg und Objekt in einem eigenständigen „Informationscontainer“ abgelegt werden. Weiter ist es möglich, sämtliche Objekte zu

[6] Realisierung der mc-mc-Beziehung zwischen *INFO* und *OBJEKT*

[7] Realisierung der mc-mc-Beziehung zwischen *INFO* und *HERB*

einem Beleg in einem einzigen „Container“ zusammenzufassen. Zwischen diesen beiden Extremen können beliebige Abstufungen existieren.

Die „richtige“ Vorgehensweise gibt es nicht. An dieser Stelle kann lediglich die Empfehlung gegeben werden, nur die informationstechnisch zusammengehörigen Objekte zu einem „Container“ zusammenzuführen. Existieren etwa drei Fotografien eines Beleges aus verschiedenen Perspektiven und eine elektronenmikroskopische Aufnahme, so wäre es nicht sinnvoll, alle vier Bilder in einem einzigen „Informationscontainer“ mit dem Objekt zu verknüpfen, wohl aber die drei Fotografien, da sie thematisch zusammengehören. Eine solche Entscheidung liegt aber in der Verantwortung des Benutzers.

## 5.3 Physikalische Speicherung der Multimediadaten

Ebenso wie andere Datenbankmanagementsysteme, stellt auch Oracle für Tabellenspalten den Datentyp „Binary Large Object“ (abgekürzt: BLOB) zur Verfügung, womit binäre Daten beliebiger Größe[8] gespeichert werden können. Dabei kann ausgewählt werden, ob die Daten in der Datenbank selbst abgelegt werden (Datentyp *BLOB*) oder ob lediglich ein Verweis auf eine Datei außerhalb der Datenbank im Dateisystem verwaltet werden soll (Datentyp *BFILE*).

Tabellen werden in Oracle in einem so genannten *Tablespace* verwaltet. Dieses wiederum stellt physikalisch eine einzige große Datei auf der Festplatte dar. Werden nun BLOBs direkt in der Datenbank gespeichert, so kann diese Datei schnell sehr groß werden und an die vom Betriebssystem vorgegebenen Grenzen stoßen. Bei einer Auslagerung ins Dateisystem hingegen werden nur der Dateiname und das Verzeichnis der Binärdatei in der Datenbank verwaltet. Hierbei wird jedoch nicht die Existenz der Datei überprüft; für die Konsistenz der Daten muss manuell gesorgt werden. Sollen viele Datensätzen gespeichert werden, so kann die Anzahl der Dateien in einem Verzeichnis sehr groß werden, sodass auch hier betriebssystembedingte Grenzen erreicht werden können, sofern nicht Mechanismen zur Verwendung mehrerer Verzeichnisse implementiert werden. Der Vorteil bei der Verwendung des Datentyps *BFILE* anstelle von *BLOB* ist die Möglichkeit, die Binärdaten bei Webabfragen direkt ansprechen zu können und sie nicht erst aus der Datenbank abrufen zu müssen.

Für Datenbankabfragen, welche den binären Inhalt von *BLOBs* oder *BFILEs* zurückliefern, ist die Wahl des Datentyps unerheblich. Beide können auf dieselbe Weise angesprochen werden. Die einzige Ausnahme besteht in der für die Programmierung der SysTax-Clients verwendeten Oracle-Forms-Version, mit der es nicht möglich ist, auf Inhalte von *BFILE*-Spalten zuzugreifen. Dieses Problem kann behelfsmäßig umgangen werden, indem bei einer Abfrage der Inhalt eines *BFILEs* zuerst in einem internen *BLOB*

[8] Die maximale Größe unterliegt lediglich physikalischen Grenzen wie z. B. Speicherplatz oder die vom Betriebssystem abhängige maximale Dateigröße.

zwischengespeichert wird, welches dann der Abfrage zur Verfügung steht.[9] Um doppelte Datenhaltung zu vermeiden, empfiehlt es sich, diese „Cache-BLOBs“ in regelmäßigen Abständen zu leeren.

In einer Webabfrage bietet die Verwaltung der Daten in *BFILEs* klare Vorteile gegenüber der Verwendung von internen *BLOBs*. Bei der internen Speicherung muss ein vom Webserver aufgerufenes CGI-Skript zur Ausgabe eines Multimediaobjekts den vollständigen Inhalt des entsprechenden BLOBs erst aus der Datenbank extrahieren, was unter Umständen zu Verzögerungen bei der Anzeige führen kann. Bei der Speicherung in Spalten des Typs *BFILE* benötigt das Skript lediglich den Pfad und den Dateinamen aus der Datenbank und kann damit die Binärdaten, je nach Server-Konfiguration, direkt ansprechen, indem z. B. bei Bildern aus diesen Informationen direkt das *src*-Attribut des <*img*>-Elements erstellt wird.

Die Frage nach dem Abrufen von Multimedia-Inhalten ist somit geklärt. Es bleibt offen, wie die Binärdaten vom Computer des Benutzers in die Datenbank gelangen. Zwei mögliche Wege sollen hier diskutiert werden: der Datenimport unter Verwendung von Schnittstellen[10] und die direkte Eingabe mittels der in Oracle-Forms entwickelten SysTax-Clients.

Bei Verwendung der Benutzeroberfläche besteht die Möglichkeit, die Binärdaten entweder sofort vom Clientrechner in die Datenbank zu laden, oder aber nur die Parameter abzuspeichern und die Binärdateien zu einem späteren Zeitpunkt in geeigneter Form nachzuliefern.

Wird die Option des sofortigen Hochladens der Binärdatei gewählt, so müssen einige Schwierigkeiten umgangen werden, die durch die Benutzung von Oracle-Forms entstehen. So können z. B. Bilder nur in einem so genannten „Image-Object“ geöffnet und zur Anzeige gebracht werden, dessen Format[11] während der Entwicklung fest voreingestellt werden muss. Dieses „Image-Object“ kann man zwar einem internen *BLOB* zuordnen und auf diese Weise auch Bilder in die Datenbank laden, dabei wird aber von Oracle-Forms immer eine Konvertierung in das voreingestellte Format durchgeführt. Dies ist natürlich weder im Sinne des Benutzers noch von SysTax gewünscht, denn Binärdaten sollen als Original in die Datenbank transferiert werden. Damit muss ein anderer Weg gefunden werden, die Bilder, Tondateien usw. vom Clientrechner auf den Server zu übertragen.

Es gibt sicherlich mehrere Möglichkeiten zur Übertragung von Dateien. So wäre z. B. die Verwendung des *File Transfer Protocol* (FTP) eine davon. Hierbei sollte aber ein Mechanismus eingesetzt wird, der garantiert, dass keine Dateien überschrieben werden, wenn verschiedene Benutzer unabhängig voneinander dieselben Dateinamen vergeben.

Dem SysTax-Projekt wurde vom „Verlag für interaktive Medien“[12] ein kleines Kom-

[9] In späteren Forms-Versionen ist dieses Problem behoben, aber aus Gründen der Abwärtskompatibilität muss es hier berücksichtigt werden.

[10] siehe hierzu auch Kapitel 6

[11] JPEG, GIF oder TIFF

[12] zuständig für die EDV-Koordination des EDIS-Projektes

mandozeilen-Programm zur Verfügung gestellt, das als Parameter den Tabellen- und Spaltennamen, den Dateinamen sowie die Oracle-interne *RowID*[13] erhält und die angegebene Datei direkt in das durch diese *RowId* referenzierte BLOB-Feld lädt.

Werden Multimediadaten in internen *BLOB*-Feldern gespeichert, so müssen nur noch die Metadaten, wie etwa Dateigröße, Höhe und Breite in Pixel bei Bildern, Länge in Sekunden bei Tönen, usw., aus den Binärdaten extrahiert und eventuell bei Bildern web-optimierte Versionen und Thumbnails erstellt werden (siehe unten).

Bei Verwendung von *BFILE*-Spalten kommt noch ein weiterer Schritt hinzu: Das soeben neu erstellte interne *BLOB* muss in eine Datei exportiert und das *BFILE* entsprechend initialisiert werden. Im Anschluss kann das als Cache-Speicher verwendete interne *BLOB* gelöscht werden und der Speichervorgang ist damit abgeschlossen.

Entscheidet sich der Benutzer hingegen für das Nachliefern der Dateien, werden nicht nur die Objektattribute gespeichert. Vielmehr müssen zusätzlich geeignete Identifizierungsmerkmale archiviert werden, damit eine Zuordnung der Binärdaten zu den schon gespeicherten Parametern möglich ist. Hierfür existiert eine Tabelle, welche den Primärschlüssel des Objektdatensatzes, den originalen Dateiname und die *UserId* des Benutzers festhält. Unter der Voraussetzung, dass sich dieser Original-Dateiname nach dem Abspeichern der Objektattribute nicht ändert, ist damit die eindeutige Zuordnung gewährleistet. Sobald sich die Binärdateien physikalisch auf dem Server befinden, ergibt sich eine zum reinen Datenimport analoge Situation.

Bei einem Datenimport werden zusätzlich zu den Binärdateien noch die Parameter als Tabelle mitgeliefert. Diese gilt es mittels geeigneter Routinen zuerst zu importieren. Im Anschluss können die Binärobjekte importiert und mit diesen neu erstellten Einträgen in der Tabelle *OBJEKT* verknüpft werden.

Ungeachtet dessen, auf welche Weise die Objektattribute in die Datenbank aufgenommen wurden, der Import der Binärdateien selbst unterscheidet sich nur hinsichtlich der Ablage in *BLOB*- oder *BFILE*-Spalten. Bei Verwendung von internen *BLOBs* muss die Datei in das entsprechende Tabellenfeld eingelesen werden, wofür Oracle spezielle Routinen zur Verfügung stellt. Je nach Dateigröße kann dieser Vorgang einige Sekunden bis mehrere Minuten dauern.

Sollen die Binärdaten mittels *BFILEs* im Dateisystem verwaltet werden, sind mehrere Schritte durchzuführen. Zuerst muss ein eindeutiger Dateiname[14] kreiert und die Datei entsprechend umbenannt werden, denn die von den Benutzern vergebene Namen müssen nicht eindeutig sein. Gleichzeitig wird sie dabei auch schon in das Verzeichnis kopiert oder verschoben, in dem die Binärobjekte abgelegt sind. Jetzt kann die entsprechende *BFILE*-Spalte mit den Werten „Verzeichnis“ und „Dateiname“ initialisiert werden. Je nachdem, ob die Datei bei diesem Vorgang kopiert oder verschoben wird, dauert dieser

[13] Jeder Datensatz erhält einen von der Datenbank generierten, systemweit eindeutigen Schlüssel, auf den lesend zugegriffen werden kann.

[14] Als eindeutiger Dateiname kann z. B. der für alle Multimediaobjekte eindeutige Primärschlüssel der Tabelle *OBJEKT* herangezogen werden.

nur Bruchteile von Sekunden (beim Verschieben innerhalb desselben Dateisystems) bis hin zu mehreren Minuten (beim Kopieren, abhängig von der Dateigröße).

Speziell für die Be- und Verarbeitung von Multimediadaten stellt Oracle ab Version 8 spezielle Methoden zur Verfügung („Oracle *inter*Media“), um z. B. Daten in verschiedene Dateiformate zu konvertieren oder um Parameter zu extrahieren. Diese Funktionalität wird für SysTax benutzt, um bei Bildern und Tondateien nicht mitgelieferte Metadaten zu ermitteln. Dies wären im Einzelnen der Mimetype, die Länge in Sekunden bei Tondateien und die Auflösung in Pixel bei Bildern. Zusätzlich wird die kleinere, weboptimierte Version eines Bildes erzeugt, sofern diese nicht ebenfalls vom Benutzer geliefert wurde. Abschließend wird damit ein kleines Thumbnail für die Anzeige im Internet generiert. Diese Konvertierungen sind natürlich nur für die vom *inter*Media-Paket unterstützten Formate möglich, bei nicht unterstützten Dateiformaten wird der Benutzer durch eine Fehlermeldung benachrichtigt.

Um die beschriebenen Funktionen benutzen zu können, müssen die Binärdaten in eine spezielle, interne Struktur geladen werden, einen objektrelationalen, abstrakten Datentyp[15]. Befinden sich die Daten bereits in einem internen *BLOB*, so kann dieses direkt als Quelle für diese Struktur dienen, bei Verwendung von *BFILEs* zur Speicherung der Binärdaten müssen diese erst in ein *BLOB* geladen werden. Damit relativiert sich der mögliche Geschwindigkeitsvorteil der *BFILE*-Variante gegenüber der internen Speicherung wieder.

Allein wegen der großen zu erwartenden Datenmenge[16] wurde für SysTax die Speicherung im Dateisystem gewählt, trotz der Nachteile durch die Verwendung von *BFILE*-Spalten mit Oracle-Forms. Aus denselben Gründen werden auch die weboptimierten Versionen der Bilder in *BFILE*-Spalten verwaltet.

[15] Bei Bildern wäre dies der *ORDImage-Type*, bei Tönen der *ORDSound-Type*.

[16] geschätzt (2003): ca. 30000 Bilder mit ca. 150 GB; ca. 7000 Tondateien mit ca. 33 GB
Tatsächliche Datenmenge, Stand Mai 2005: 50500 Bilder mit insgesamt ca. 168 GB, 7779 Tondateien mit insgesamt ca. 33 GB.

# 6 Import-Schnittstelle

## 6.1 Definition des Begriffs Schnittstelle

Schnittstellen können in einer Vielzahl von Formen auftreten und sind nicht auf den Bereich der Informatik beschränkt, sie kommen aber vor allem dort zur Anwendung. Eine allgemeine Definition gibt die Internet-Enzyklopädie Wikipedia: *„Eine Schnittstelle (Interface) ist ein Teil eines Systems, das dem Austausch von Informationen, Energie oder Materie mit anderen Systemen dient. Eine Schnittstelle wird durch eine Menge von Regeln beschrieben, der Schnittstellenbeschreibung.“*[1]

Beispiele für Schnittstellen sind:

- (grafische) Benutzeroberfläche: eine „Mensch-Maschine-Schnittstelle“ zur Bedienung eines Programms oder Computers
- SCSI: „small computer system interface“: eine standardisierte Schnittstelle zur Datenübertragung zwischen verschiedenen Geräten in einem Computer, z. B. Festplatte, Scanner oder Drucker.
- in der Programmierung: Module und Bibliotheken definieren und realisieren Schnittstellen, zur Verwendung von bereitgestellten Prozeduren oder Klassen in anderen Programmen.

Im Bereich der Datenbankanwendungen sind neben der (graphischen) Benutzeroberfläche vor allem die Schnittstellen zum Datenaustausch mit anderen Systemen von Interesse. Dabei definiert die Schnittstelle das Format, in dem die Daten ausgegeben werden bzw. bei einem Import vorliegen müssen. Während eine Ausgaberoutine relativ einfach realisierbar ist – hier müssen die Datensätze lediglich in geeigneter Weise abgefragt und im Schnittstellenformat ausgegeben werden –, kann eine Importprozedur wesentlich komplexer werden.

Es muss eine Plausibilitäts- und Integritätsprüfung der zu importierenden Daten stattfinden. Weiterhin muss festgestellt werden, ob ein Datensatz schon existiert und somit ein Einfügen des neuen Datensatzes zu Dubletten führen würde. Weiterhin es das Abfangen von und Reagieren auf Fehler notwendig. Ebenso muss auf die unterschiedliche

[1] http://de.wikipedia.org/wiki/Hauptseite

Behandlung von Einfüge- und Ändern-Operationen geachtet werden. Bei der letztgenannten Prozedur besteht unter anderem das Problem des Auffindens des zu ändernden Datensatzes.

Dieses Kapitel beschäftigt sich mit den grundlegenden Problemen bei der Definition einer Schnittstelle für den Import biologischer Daten in SysTax.

## 6.2 Allgemeines zur Importschnittstelle

Innerhalb des EDIS- und später des GBIF-Projekts gibt es einige Institutionen, welche die von ihnen erhobenen Daten in SysTax speichern und über dessen Abfragemöglichkeiten im Internet zur Verfügung stellen. Die meisten dieser Projektpartner hatten schon vor Projektbeginn relativ viele Daten elektronisch erfasst, meist in Form von (Excel-) Tabellen oder kleinen (Access-) Datenbanken. Auf Wunsch der Projektpartner sollen diese Datenbestände weiter gepflegt und regelmäßig ein entsprechendes Update an SysTax übermittelt werden. Die Verwendung der SysTax-Benutzeroberfläche war folglich keine Option, um Daten nicht doppelt eingegeben zu müssen.

Damit bestand die Notwendigkeit, ein Format zu definieren, in dem die Daten der einzelnen Projektpartner (im Weiteren allgemein mit Datenlieferanten bezeichnet) an SysTax geliefert und dort importiert werden sollten. Dieses Format musste den folgenden Anforderungen genügen:

- Darstellung der komplexen Struktur biologischer Daten: Hierarchien (z. B. die Zuordnung eines Taxons zu seinem Vorgänger) und 1–n Abhängigkeiten (z. B. mehrere Taxa als Bestimmung für denselben Sammlungsbeleg)
- Datensätzen sollten änderbar sein
- möglichst einfache Konvertierung der Daten des Datenlieferanten in das Schnittstellenformat
- Unabhängigkeit vom Betriebsystem

Grundlegendes Problem bei der Definition eines Schnittstellenformats für biologische Daten ist die Vielfalt der auszutauschenden Daten: Datensätze aus den Bereichen der Sammlungsverwaltung, der Geographie, der Taxonomie und Synonymie, der Literatur und Multimediaobjekte, sowie Verknüpfungen zwischen den einzelnen Gebieten sollen an SysTax geliefert und dort importiert werden. Damit ist es wenig sinnvoll, eine einzige, große Schnittstelle zu definieren, die alles beinhaltet, sondern es ist besser, für jeden Themenbereich ein eigenes Schnittstellenformat zu verwenden. Verknüpfungen zwischen den einzelnen Themenbereichen müssen dann analog einer relationalen Datenbank mittels Vergabe von eindeutigen Schlüsselwerten realisiert werden, welche jedem zu importierenden Datensatz zugeordnet sind.[2]

[2] zur Problematik bei der Erstellung solcher Schlüssel siehe Abschnitt 6.5

Für jeden Bereich müssen also eigene Schnittstellen definiert werden. Um diese verschiedenen Formate möglichst einheitlich zu gestalten, sollten sie ähnlich aufgebaut sein und sich nur durch die Art der darüber zu transportierenden Daten unterscheiden. Wenn im Folgenden von *der* Schnittstelle die Rede ist, so gelten diese Aussagen für jede einzelne Schnittstelle aus den vorhandenen Bereichen.

Am Anfang galt es, sich für ein physikalisches Format der Schnittstellen und in einem zweiten Schritt für eine Struktur der Daten zu entscheiden. Über den dritten Punkt, den Inhalt der einzelnen Schnittstellen, soll an dieser Stelle nicht diskutiert werden. Dieser ist einerseits abhängig davon, was die Projektpartner an Daten erfassen und liefern können, zum anderen kann die inhaltliche Entwicklung der Schnittstellen nicht als abgeschlossen gelten. Sobald neue Datenlieferanten hinzukommen, kann es sein, dass Daten geliefert werden, die in der Schnittstelle noch nicht berücksichtigt sind, sodass diese erweitert werden muss.

Als physikalische Form wurden reine Textdateien gewählt, da sämtliche zu importierende Datensätze aus Text oder Zahlen bestehen[3]. Diese ASCII-Dateien haben gegenüber allen anderen (proprietären) Formaten entscheidende Vorteile: Sie sind sowohl von Computersystemen als auch von Nutzern lesbar, in der Regel vom Betriebssystem unabhängig und benötigen zum Betrachten oder Editieren nur minimale Voraussetzungen. Ein einfacher Texteditor ist meistens schon ausreichend. Hinzu kommt, dass bei den einzelnen Datenlieferanten unterschiedliche Programme zur Datenerfassung verwendet werden. Die meisten dieser Tabellenkalkulations- oder Datenbankprogramme sind nicht nur in der Lage, die Daten neben ihrem eigenen Dateiformat und einigen anderen, fremden Formaten zu speichern, sondern eben auch als ASCII-Textdateien zu exportieren.

Bei ASCII-Dateien können zwei Möglichkeiten zur Darstellung der Datensätze unterschieden werden: zeilenbasierte, tabellarische Daten und strukturierte Daten. Von tabellarischen Daten spricht man, wenn jede Zeile genau einen Datensatz beinhaltet. Die einzelnen Felder besitzen entweder eine feste Breite oder werden durch ein festgelegtes, nicht in den einzelnen Feldern vorkommendes Zeichen voneinander getrennt, häufig das Tabulatorzeichen. Strukturierte Daten hingegen sind nicht an diese Einschränkungen gebunden und können je nach Problemstellung verschiedenste Gestalt annehmen. Eine der vielen Möglichkeiten, Daten in Textdateien zu strukturieren, ist die Verwendung der *Extensible Markup Language* (XML). Diese hat den Vorteil, dass sie mittlerweile sehr verbreitet ist und es eine Vielzahl an Programmen und Werkzeugen gibt, um XML-Daten zu erzeugen, auszulesen und auf Validität zu überprüfen. Dies ist bei anderen Strukturen meistens nicht der Fall, sodass hierfür geeignete Programme erst entwickeln werden müssen.

Die beiden folgenden Abschnitte gehen näher auf die Vor- und Nachteile von tabellarischen und XML-Dateien ein und überprüfen beide auf ihre Tauglichkeit für ein Importschnittstellenformat von SysTax.

[3] Außer Multimediaobjekte wie Bilder oder Tondateien; diese werden separat geliefert, wobei die Schnittstelle einen „Zeiger“ (Verzeichnis und Dateinamen) auf das Objekt beinhaltet.

## 6.3 Importschnittstellenformat

### 6.3.1 XML-Dateien

Die *Extensible Markup Language* ist eine Metasprache, die es ermöglicht, eine den entsprechenden Bedürfnissen angepasste Auszeichnungssprache zu beschreiben. Die grundlegende Idee hinter diesen Auszeichnungssprachen ist die Trennung von Inhalt (also Daten oder Text), Semantik und meist auch Darstellung, was durch die Kennzeichnung von Textstellen mit besonderer Bedeutung erreicht wird. XML kann als Untermenge der *Standard Generalized Markup Language* (SGML)[4] betrachtet werden, deren prominentestes Beispiel wohl die *Hypertext Markup Language* (HTML) ist. HTML, hauptsächlich zur Darstellung von Dokumenten im *World Wide Web* verwendet, zeigte sich bald als zu unflexibel, um den immer weiter gestiegenen Anforderungen zu genügen, sodass immer mehr Erweiterungen dieser Sprache notwendig waren. Mit XML wurde eine relativ einfache Möglichkeit gefunden, neue Elemente und Attribute zu definieren um eigene Daten- und Dokumentstrukturen zu beschreiben. Obwohl SGML umfangreicher ist als XML, war die Einführung einer neuen Metasprache notwendig, da die Komplexität von SGML die Softwareentwicklung, aber auch die Lesbarkeit erschwert (Behme und Mintert, 2000).

Welche Elemente und Attribute in einer durch XML beschriebenen Sprache vorkommen, also ihr Vokabular, und in welcher Beziehung diese zueinander stehen, vergleichbar mit der Grammatik bzw. Syntax einer natürlichen Sprache, wird in der so genannten *Document Type Definition* (DTD) beschrieben. Eine Weiterentwicklung davon ist das Anfang 2001 zum W3C-Standard[5] erhobene *XML-Schema*, welches die Gestaltungsmöglichkeiten einer XML-Sprache gegenüber der DTD erweitert. Neben der DTD und dem XML-Schema gibt es noch weitere Standards zur Definition einer XML-Sprache, auf die an dieser Stelle nicht weiter eingegangen werden soll.[6]

Eine XML-Datei, die gemäß den XML-Spezifikationen aufgebaut ist, nennt man *wohlgeformt*. Entspricht eine wohlgeformte XML-Datei zusätzlich noch in allen Punkten einer DTD bzw. einem XML-Schema, so wird sie *valid* bzw. *gültig* genannt. Zur Überprüfung beider Bedingungen existiert bereits eine Vielzahl von Programmen und Routinen. Die Verwendung von XML zum Datenaustausch hat damit den großen Vorteil, dass kein Programmieraufwand notwendig ist, um bei einer Datenlieferung zu überprüfen, ob es sich tatsächlich um gültiges XML handelt und wenn ja, ob sie der Schnittstelle, also dem XML-Schema, das die Schnittstelle beschreibt, entspricht. Logische Fehler wie die Fehlverwendung eines Elements (z. B. der Ortsname im Feld für die Ländernamen) können auf diese Weise natürlich nicht entdeckt werden, wohl aber können für Attribute Wertebereiche angegeben werden, falls diese nur bestimmte Werte annehmen dürfen. Mit der Validierung der XML-Datei würden falsche Werte erkannt werden.

[4] ISO-Standard 8879:1986

[5] *World Wide Web Consortium* – ein internationales Konsortium zur Entwicklung und Unterstützung standardisierter Technologien für das Web

[6] Für weiterführende Informationen und einer ausführlichen Link-Sammlung sei auf die Homepage von Prof. Mario Jeckle, http://www.jeckle.de, verwiesen.

Die Gestaltungsmöglichkeiten einer XML-Sprache sind vielfältig. Prinzipiell ist ein XML-Dokument baumartig aufgebaut: Ausgehend von einem Hauptelement können beliebig viele Unterelemente definiert werden, die wiederum selbst Unterelemente besitzen können. Für jedes Element kann dabei die minimale und maximale Anzahl seines Auftretens angegeben werden. Häufig wird für den Minimalwert „Null“, was einem optionalen Element entspricht, oder „Eins“, wenn es sich um ein Pflichtfeld handelt, eingesetzt. Der Maximalwert wird meistens auf „Eins“ oder „unbeschränkt“ gesetzt. Andere Werte sind durchaus zulässig aber unüblich. Des Weiteren ist es sogar möglich, rekursive Strukturen zu definieren, sodass also ein Element sich selbst als Unterelement besitzt.

Ein kleines Beispiel für eine solche Struktur:

```
<?xml version="1.0" encoding="UTF-8"?>
<taxa xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="bsp1.xsd">
  <taxon>
    <taxonid>1</taxonid>
    <name>Nonsensea</name>
    <rank>family</rank>
    <lowertaxon>
      <taxonid>2</taxonid>
      <name>Nonsensia</name>
      <rank>genus</rank>
      <lowertaxon>
        <taxonid>4</taxonid>
        <name>Nonsensia neglecta</name>
        <rank>species</rank>
      </lowertaxon>
      <lowertaxon>
        <taxonid>5</taxonid>
        <name>Nonsensia nihil</name>
        <rank>species</rank>
      </lowertaxon>
    </lowertaxon>
    <lowertaxon>
      <taxonid>3</taxonid>
      <name>Pipapoa</name>
      <rank>genus</rank>
    </lowertaxon>
  </taxon>
</taxa>
```

Das dazugehörige XML-Schema könnte folgende Gestalt haben:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema elementFormDefault="qualified"
           attributeFormDefault="unqualified"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType name="taxontype">
    <xs:sequence>
```

```
      <xs:element name="name"/>
      <xs:element name="rank"/>
      <xs:element name="taxonid"/>
      <xs:element name="lowertaxon" type="taxontype" minOccurs="0"
                  maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="taxa">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="taxon" type="taxontype"
                    maxOccurs="unbounded"/>
      </xs:sequence>
     </xs:complexType>
  </xs:element>
</xs:schema>
```

Das Hauptelement *taxa* besitzt hier nur ein Unterelement namens *taxon* vom Typ *taxontype*, welches beliebig oft vorkommen kann, mindestens jedoch einmal vorkommen muss. Die durch den Namen *taxontype* definierte Struktur besteht aus der Folge dreier Elemente: *name* und *rank* müssen genau einmal vorkommen, das Element *lowertaxon* braucht nicht oder kann beliebig oft angegeben werden. Dieses Element ist wieder vom Typ *taxontype*, sodass innerhalb dieses Elements wieder die Elemente *name* und *rank* vorkommen müssen und weitere Elemente *lowertaxon* angegeben werden können.

Dieses Beispiel belegt, wie einfach es ist, hierarchische Strukturen in XML abzubilden, vor allem dann, wenn die Anzahl der Unterstufen beliebig sein kann oder nicht bekannt ist. Des Weiteren können auf jeder Stufe beliebig viele Attribute oder Elemente, hier durch Erweiterung des Typs *taxontype*, angegeben werden. Durch die Möglichkeit zur beliebigen Wiederholung eines Elements, wie in diesem Beispiel das Element *lowertaxon*, ist XML auch zur Darstellung von c-mc-Relationen geeignet. Zusätzlich kann durch entsprechende Strukturierung jedem Oberelement, das ja ebenfalls einem vollständigen Datensatz entspricht, ein eigener Datensatzschlüssel vergeben werden.

Aufgrund der gestalterischen Vielfalt entspricht die Erstellung eines XML-Schemas dem Vorgang der Datenmodellierung, denn „*letztlich handelt es sich bei XML-Sprachen auch um Modelle einer Wirklichkeit, die zu modellieren sind*“ (Jeckle 2000). Somit ist nicht alles, was durchführbar ist, auch sinnvoll in Hinblick auf die Effizienz einer Sprache. Analog zu dem relationalen Datenmodell, für das es Regelungen gibt, deren Einhaltung zwar nicht zwingend notwendig, aber trotzdem zweckdienlich sind (z. B. Normalformen, Redundanzfreiheit), existieren auch für XML-Sprachen solche optionalen Forderungen.

Ein Schnittstellenformat, gleichgültig ob in XML oder in einem anderen Format, sollte immer an die Bedürfnisse und technischen Möglichkeiten aller über diese Schnittstelle kommunizierenden Partner angepasst sein. Doch gerade bei XML tritt hier ein Problem auf: Während auf der Seite des Datenimports eine Vielzahl von Routinen zum Transfer von Daten aus einem XML-Dokument in eine Datenbank existieren, die teilweise auch schon in Datenbankmanagementsysteme wie Oracle integriert sind, muss auf der

Seite des Datenlieferanten teils erheblicher Aufwand betrieben werden, um die Daten in das gewünschten Format zu konvertieren. Ein Beispiel hierfür ist die taxonomische Hierarchie: Häufig werden Taxa als Attribut der Bestimmung eines Sammlungsbeleges „flach“ erfasst, d. h. ein Datensatz beinhaltet nicht nur die unterste Stufe, sondern häufig sämtliche Stufen, sodass zur Bestimmung nicht nur der Artname, sondern zusätzlich die Gattung der Art, die Familie der Gattung usw. gehören. Damit stellt sich die Frage, ob auf die Vorteile von XML, hier die Verschachtelung von Elementen, verzichtet werden und die Hierarchie „flach“ modelliert werden soll. Andernfalls wäre es einigen Datenlieferanten aufgrund der Struktur ihrer Datenbank nicht möglich, ohne großen Programmieraufwand die Schnittstelle zu bedienen. Im Hinblick auf die Kundenfreundlichkeit wäre eine komplexe XML-Schnittstelle nicht sinnvoll.

### 6.3.2 Tabellarische Textdateien

Tabellarischen Textdateien liegt ebenfalls eine Struktur zugrunde, wenn auch eine einfache. Jede Zeile besteht hier aus genau einem Datensatz, der Zeilentrenner ist somit gleichzeitig auch Datensatztrenner. Die einzelnen Daten innerhalb einer Zeile besitzen entweder eine feste Zeichenbreite, auf die ein Datum gegebenenfalls mit Leerzeichen aufgefüllt werden muss, oder sie werden durch ein festgelegtes Trennzeichen (z. B. Tabulator, Semikolon oder Doppelpunkt) voneinander getrennt. Im letzten Fall spricht man auch von so genannten CSV-Dateien, *character separated value*. Bei der Wahl des Trennzeichens ist darauf zu achten, dass es entweder nicht innerhalb der einzelnen Felder verwendet wird, oder dass jedes Vorkommen des Trenners innerhalb der Daten vor der falschen Interpretation als Feldtrenner geschützt wird, z. B. durch Voranstellen eines „Schutzzeichens“. Als Quasi-Standard für CSV-Dateien hat sich die von Microsoft-Excel verwendeten Form etabliert: Hier wird das Semikolon als Trennzeichen benutzt und jedes Feld, welches mindestens ein Semikolon beinhaltet, wird mit doppelten Anführungszeichen umschlossen. Kommt auch dieses Zeichen in einem Feldinhalt vor, so wird es durch Voranstellen eines weiteren doppelten Anführungszeichens geschützt.

Der Vorteil von tabellarischen Textdateien, insbesondere von CSV-formatierten Dateien, besteht darin, dass sie aus den gängigen Tabellenkalkulationsprogrammen und Datenbankmanagementsystemen generiert werden können. Tatsächlich besteht sogar explizit die Möglichkeit, beim Speichern nicht das Standardformat der Anwendung sondern das CSV-Format auswählen zu können. Zur Lieferung von Daten in einem flachen Schnittstellenformat ist damit wenig bis gar kein programmiertechnischer Aufwand nötig. Gegebenfalls müssen Felder umgruppiert oder, wenn sie der Datenlieferant nicht erfasst, leer gelassen werden, um dem inhaltlichen Format der Schnittstelle zu entsprechen. Datenlieferanten, die ihre Datenbestände mittels Programmen wie Excel oder Access verwalten, können somit ihre Daten verhältnismäßig einfach in einer tabellarischen Textdatei nach SysTax exportieren.

Der Umstand, dass jede Zeile einer solchen Textdatei genau einem Datensatz entspricht, bedingt einen großen Nachteil dieser Struktur: Es können nur solche Daten dargestellt werden, die in einer direkten 1–1-Beziehung zu dem Schlüssel des Datensatzes stehen;

1–n- oder gar n–m-Verknüpfungen sind nicht realisierbar und müssen entsprechend aufgelöst werden. Bei solchen Daten, z. B. bei der Bestimmungen eines Sammlungsbeleges, ist es notwendig, analog der Realisierung einer 1–mc- oder mc–mc-Relation zweier Entitäten in einer relationalen Datenbank, jeden Datensatz mit einem eindeutigen Primärschlüssel zu versehen und eine weitere flache Schnittstelle zu definieren, welche die Verknüpfungsdaten aufnimmt, die aus den entsprechenden Schlüssel und evtl. zusätzlichen Attributen bestehen.

Des Weiteren bietet eine Textdatei mit einer Zeile pro Datensatz keine Möglichkeit, hierarchische Strukturen direkt darzustellen, wie es z. B. in der Taxonomie oder bei politischen Geographieangaben erforderlich ist. Indirekt können Hierarchien auf zwei verschiedene Arten abgebildet werden: Entweder die Hierarchie wird „ausgeflacht“ , d. h. für jede Stufe werden ein oder mehrere Felder angelegt, oder jede Stufe wird als eigener Datensatz mit eindeutigem Schlüssel behandelt und erhält als Attribut den Schlüssel der nächsthöheren Stufe.

Die erste Möglichkeit kommt vor allem dann in Betracht, wenn die Anzahl der Stufen fest und nicht allzu hoch ist und wenn pro Stufe nur wenige Felder benötigt werden. Tabelle 6.1 zeigt beispielhaft einen Ausschnitt aus einer Schnittstelle für taxonomische Daten. Es werden zwei Stufen (Familie und Gattung) der Hierarchie dargestellt. Dabei besitzt jede einzelne Stufe nur die beiden Attribute Taxonname und Taxonautor incl. Jahreszahl. Gefüllt ist diese Schnittstelle in diesem Beispiel mit einer Familie und zwei Gattungen, wobei diese beiden Gattungen dieselbe Familie als direkten Vorgänger besitzen.

| TaxonId | ... | Family | FamilyAuthorYear | Genus | GenusAuthorYear | ... |
|---|---|---|---|---|---|---|
| T10 | ... | Pipapoidae | TL, 2004 | | | ... |
| T20 | ... | Pipapoidae | TL, 2004 | Pipapoa | TL, 2004 | ... |
| T30 | ... | Pipapoidae | TL, 2004 | Miserabilis | TL, 2004 | ... |

Tabelle 6.1: 1. Möglichkeit: „Ausflachung“ hierarchischer Beziehungen

Damit wird auch schon das Problem dieser Struktur sichtbar: die Redundanz bei höheren Stufen. Sollte hier aus irgendeinem Grund ein Fehler in den Daten vorliegen, würde z. B. dasselbe Taxon mit und einmal ohne Autorenangabe vorliegen, so könnte die Importroutine nicht erkennen, dass es sich hierbei tatsächlich um dasselbe Taxon handelt. Die Daten würden wie zwei verschiedene Taxa behandelt und entsprechend importiert werden.

Zusätzlich ist es notwendig festzulegen, auf was genau sich der Datensatzschlüssel (hier die *TaxonId*) bezieht. Bezieht er sich auf die gesamte Hierarchie, so kann man nicht auf eine einzelne Stufe explizit referenzieren. Bezieht er sich hingehen auf eine einzelne Stufe innerhalb der Hierarchie, ist es fraglich, auf welche. Die von SysTax gewählte Konvention ist, dass sich der Datensatzschlüssel immer auf die unterste der angegebenen Stufen bezieht, also in der taxonomischen Schnittstelle auf die Gattung, wenn keine Art angegeben ist, auf die Familie, wenn keine Gattung angegeben ist usw. Somit bezieht

sich *TaxonId* T20 in diesem Beispiel auf die Gattung *Pipapoa TL, 2004* und die *TaxonId* T30 auf die Gattung *Miserabilis TL, 2004*. Wird es gewünscht, eine höhere Stufe explizit mittels einem Schlüssel referenzieren zu können, so muss für diese Stufe ein eigener Datensatz in der Tabelle erzeugt werden, mit leeren Einträgen für alle darunter liegenden Stufen (wie in diesem Beispiel der Eintrag mit *TaxonId* T10).

Die andere Möglichkeit, also jede Stufe der Hierarchie als eigenständigen Datensatz mit Referenz auf die nächste Stufe zu betrachten, ist vor allem dann interessant, wenn es beliebig viele hierarchische Stufen geben kann oder jede Stufe aus vielen einzelnen Feldern zusammengesetzt wird. Die Zuordnung eines Datensatzes zu seinem Vorgänger erfolgt in diesem Fall über den Schlüssel des Vorgängers, der dem Datensatz als weiteres Attribut angehängt wird. Dieses Prinzip wird beispielhaft in Tabelle 6.2 veranschaulicht. Hier bezeichnet die Spalte *TaxonId* den Schlüssel des Datensatzes und das Feld *VTaxonId* beinhaltet die *TaxonId* des direkten Vorgängers.

| TaxonId | VTaxonId | Stufe | Name | AuthorYear | ... |
|---|---|---|---|---|---|
| T10 | ... | family | Pipapoidae | TL, 2004 | ... |
| T20 | T10 | genus | Pipapoa | TL, 2004 | ... |
| T30 | T10 | genus | Miserabilis | TL, 2004 | ... |

Tabelle 6.2: 2. Möglichkeit: Beibehaltung der hierarchischen Struktur

Bei dieser Struktur ist die Reihenfolge der Datensätze innerhalb einer Datei von Bedeutung, da tabellarische Textdateien meist zeilenweise, also datensatzweise, eingelesen und importiert werden. Damit ein Datensatz auch tatsächlich einem Vorgänger zugeordnet wird, muss sichergestellt sein, dass dieser Vorgänger bereits importiert wurde. Dies kann entweder durch einen früheren Import geschehen sein oder der Vorgänger-Datensatz muss oberhalb des gerade zu importierenden Datensatzes in der Datei stehen. Umgangen werden kann dieses Problem, indem man eine zu importierende Datei mehrmals einliest: Im ersten Durchgang werden nur die Datensätze auf der höchsten Stufe, im zweiten diejenigen auf der nächstniedrigeren ausgelesen usw.

Im Gegensatz zur „Ausflachung“ kommt bei dieser Struktur jeder Datensatz genau einmal vor, unabhängig davon, auf welcher Stufe er sich befindet. Dadurch wird unter anderem auch die Fehleranfälligkeit reduziert. Eine neue Fehlerquelle stellt allerdings die Verwendung von Fremdschlüsseln zur Referenzierung des Vorgängers dar. Da ein Wert im Fremdschlüssel auch als Primärschlüssel eines anderen Datensatzes vorkommen muss, muss hier genauso wie bei relationalen Datenbanken auf die Fremdschlüsselintegrität geachtet werden. Im Gegensatz zu relationalen Datenbanken, kann diese aber nicht ohne Programmieraufwand überprüft werden.

Trotz der Vorteile der hierarchischen Struktur gegenüber der flachen Darstellung wurde für die SysTax-Schnittstelle die ausgeflachte Struktur gewählt, denn fast jeder Datenlieferant hatte zum Zeitpunkt der Konzipierung der Schnittstellen seine Daten in einer solchen flachen Struktur vorliegen. Eine Konvertierung in die hierarchische Darstellungs-

weise ließe sich ohne hohen Programmieraufwand auf Seiten der Datenlieferanten, meisten Mitarbeiter eines Instituts ohne eigene Informatik-Abteilung, nicht bewerkstelligen und wurde von ihnen auch nicht gewünscht.

Neben der erwähnten fehlenden Möglichkeit zur Darstellung von mc–mc-Beziehungen zwischen Daten und der eben besprochenen Problematik mit hierarchischen Daten ist ein weiterer Nachteil tabellarischer Textdateien, dass, anders als bei XML-Dateien, keine fertigen Programme zur Prüfung auf Konsistenz und Plausibilität existieren. Damit müssen für jede einzelne Schnittstellen eigene Programme entwickelt werden, die testen, ob die gelieferten Daten auch der Schnittstellendefiniton entsprechen, also ob die Anzahl und Namen der Felder stimmen, ob alle Pflichtfelder gefüllt sind, ob in Feldern, die nur bestimmte Werte annehmen dürfen, auch genau diese Werte enthalten sind etc.

### 6.3.3 Schlussfolgerung

Beide der hier vorgestellten Schnittstellenformate erfüllen drei der eingangs formulierten Bedingungen: Über Umwege sind komplexe Strukturen darstellbar, mittels Schlüsseln lässt sich jeder Datensatz eindeutig identifizieren und somit später ändern und da beide Formate auf reinen Textdateien basieren, sind sie unabhängig vom verwendeten Betriebssystem. Lediglich bei der Forderung nach der möglichst einfachen Konvertierbarkeit der Daten in das Schnittstellenformat müssen beide Möglichkeiten differenziert betrachtet werden.

Während der Vorteil bei Verwendung der XML-Struktur eindeutig beim Importeur, also bei SysTax liegt, dem fertige Programme zur Validierung und zum Auslesen der Daten stehen bereits zur Verfügung stehen, besteht hierbei jedoch Schwierigkeit, die Daten des Exporteurs in das notwendige Format zu konvertieren. Die meisten Projektpartner benutzen nur wenige, dafür aber relativ flache und somit redundante Tabellen. Diese in das XML-Format zu bringen ist häufig relativ schwierig und aufwändig. Ein entsprechender Versuch zu Projektbeginn ist aus diesem Grund gescheitert.

Somit wurde für die SysTax-Importschnittstellen das CSV-Format gewählt. Die Datenlieferanten können ihre Daten ohne großen Aufwand in diesem Format liefern. Die Validierung der Daten auf der Importseite ist zwar schwieriger, da entsprechende Prüfroutinen entwickelt werden müssen. Diese stehen dann aber für zukünftige Importe zur Verfügung und müssen nur bei Änderungen des Schnittstellenformates, etwa beim Hinzufügen von neuen Spalten, entsprechend angepasst werden.

## 6.4 Updates

Eine der Forderungen an eine SysTax-Importschnittstelle ist, dass schon importierte Datensätze später wieder durch einen Reimport änderbar sind. Dabei treten mehrere Schwierigkeiten auf: Als Erstes muss der zu ändernde Datensatz identifiziert werden,

anschließend muss geklärt werden, ob dieser Datensatz geändert werden darf und wenn ja, auf welche Weise.

Das erste Problem betrifft das Auffinden des zu ändernden Datensatzes. Eine Möglichkeit besteht darin, sowohl die alten, zu ändernden Daten zusammen mit der jeweiligen Änderung zu liefern. Dieses Verfahren ist jedoch wenig praktikabel. Zum einen muss der Datenlieferant den Zustand des Datensatzes zum Zeitpunkt seiner ersten Lieferung archivieren, was kaum jemand durchführt, zum anderen darf der Datensatz nach seinem Import in SysTax nicht geändert worden sein. Zudem wäre es notwendig, zwei verschiedene Versionen der Schnittstelle zu erstellen, eine nur für neue und eine nur für zu ändernde Datensätze.

Eine wesentlich zweckmäßigere Lösung ist die Verwendung der Datensatzschlüssel, die zur Realisierung der Verknüpfungen zwischen den Datensätzen in den einzelnen Schnittstellen verwendet werden. Damit ein solcher Schlüssel bei einem Update zur Identifikation des zugehörigen Datensatzes herangezogen werden kann, muss er zusammen mit der Kennung des Datenlieferanten einen global eindeutigen Wert ergeben (siehe hierzu Abschnitt 6.5) und jedem importierten Datensatz als Metadatum zugewiesen worden sein. Zusätzlich muss der Schlüssel, ist er einmal einem Datensatz zugeordnet, konstant bleiben und darf nicht mehr geändert werden. Der Vorteil dieser Methode ist, dass für beide Vorgänge, also sowohl das Einfügen eines neuen als auch das Ändern eines schon importierten Datensatzes, dieselbe Schnittstellendefinition und damit auch dieselben Prüfroutinen verwendet werden können.

Sobald ein zu modifizierender Datensatz in SysTax gefunden wurde, kann er mit den neuen Daten überschrieben werden. Im Vorfeld eines solchen Updates ist zu klären, wie leere Felder in den neuen Datensätzen zu behandeln sind, ob sie zum Leeren bzw. Löschen der entsprechenden Felder des alten Datensatzes verwendet oder ob sie ignoriert werden sollen. Im ersten Fall müssen zwingend alle Felder gesetzt werden, unabhängig davon ob der Feldinhalt geändert wurde oder nicht; der Importvorgang überschreibt sämtliche alten Felder mit neuen Inhalten. Im zweiten Fall brauchen nur die Felder angegeben werden, die sich geändert haben, doch es besteht keine Möglichkeit, alte Felder zu leeren. Aus diesem Grund wurde für SysTax die erste Variante gewählt.

Problematisch beim Ändern von Datensätzen durch „Reimporte“ ist die flache Datenstruktur der Schnittstelle, wodurch 1-mc-Relationen redundant aufgelöst werden. Damit ist es oft schwer zu entscheiden, was tatsächlich geändert werden soll, die verknüpften Daten oder die Verknüpfung selbst. Ein kleines Beispiel soll dies verdeutlichen: Dem Fundort eines Sammlungsbelegs ist neben Angaben wie Koordinaten und Meereshöhe meistens ein Referenzort zugeordnet. Da mehrere Belege demselben Ort zugewiesen werden können, ist die Verknüpfung zwischen Fundort und Referenzort in SysTax eine c-mc-Relation, in der Schnittstelle aber lediglich ein weiteres Feld. In diesem Beispiel sei einem Beleg ursprünglich der Ort „Ullm“ zugewiesen worden und erst nach einem Import wurde der Schreibfehler im Ortsnamen erkannt. Verschickt der Datenlieferant nun den korrigierten Datensatz mit dem richtigen Referenzort „Ulm“, so kann man sofort erkennen, dass hier die verknüpften Daten geändert werden müssen, damit der Schreibfehler für alle Belege behoben wird, die diesem Ort zugeordnet sind. Ist in einem anderen Fall

einem Beleg der Referenzort „Ulm“ zugeordnet und der Datenlieferant stellt später fest, dass es tatsächlich „Neu-Ulm“ heißen müsste, so darf hier nur die *Verknüpfung* geändert werden – würde man die verknüpften Daten ändern, hätten alle Belege, die ursprünglich mit Recht „Ulm“ zugeordnet wurden, nun ebenfalls „Neu-Ulm“ als Referenzort.

Ein menschlicher Bearbeiter könnte bei einem Update meistens sofort entscheiden, um welchen der beiden Fälle es sich handelt, ein Computerprogramm hingegen nicht. Bei der Erstellung einer Importroutine muss man sich für eine der beiden Varianten entscheiden: Entweder es werden die verknüpften Daten geändert und man nimmt in Kauf, dass andere Datensätze nach einem Update mit für sie falschen Daten verknüpft sind, oder man ändert nur die Verknüpfung und kann damit keine generellen Schreibfehler für alle Datensätze korrigieren bzw. die alten Daten nicht löschen, da bei jedem Update ein neuer Wert für die verknüpften Daten angelegt werden muss und der alte Wert parallel bestehen bleibt, sofern diesem noch weitere Datensätze zugeordnet sind. An dieser Stelle kann keine generelle Empfehlung für die eine oder die andere Vorgehensweise ausgesprochen werden. Je nach Struktur der Daten kann die eine oder die andere Variante sinnvoll sein. Bei den SysTax-Schnittstellen hat man sich darauf verständigt, dass lediglich die Verknüpfung geändert werden soll. Damit müssen zur Korrektur von Schreibfehlern in den verknüpften Daten sämtlich Datensätze reimportiert werden.

Nachdem nun die Frage geklärt wurde, wie verknüpfte Daten zu ändern sind, stellt sich das Problem, zu entscheiden, ob ein Datensatz überhaupt modifiziert werden darf. Kann ein schon in SysTax gespeicherter Datensatz eindeutig einem Datenlieferanten zugeordnet werden, so ist diese Frage mit einem einfachen „ja“ zu beantworten. Ein Beispiel dafür sind Sammlungsbelege, denn kaum ein Beleg aus einer Sammlung wird von verschiedenen Personen unabhängig voneinander erfasst. Schwieriger wird es bei Daten, die von mehreren Projekten gleichzeitig geliefert werden könnten. Darunter fallen insbesondere Taxonnamen und Literaturzitate. Um hier doppelte Einträge zu vermeiden, wird bei einem Erstimport zunächst geprüft, ob die zu importierenden Daten schon in SysTax zu finden sind. Ist dies der Fall, so wird der Import abgebrochen und die schon vorhandenen Daten verwendet, beispielsweise für Verknüpfungen mit Datensätzen aus anderen Bereichen. Trotzdem werden die Kennung des Datenlieferanten und der Datensatzschlüssel dem gefundenen Datensatz zugewiesen.

Die Folge davon ist, dass ein Datensatz zwei oder mehrere Datenlieferanten gleichzeitig zugeordnet sein könnte. Damit stellt sich die Frage, wer diesen Datensatz ändern darf. Jeder, der schon existierende Daten seinen eigenen Daten zuordnet, sei es unfreiwillig durch Importe oder gewollt durch Benutzung der Benutzeroberfläche, verlässt sich darauf, dass die Daten korrekt sind und konstant bleiben bzw. höchstens zur Fehlerkorrektur geändert werden. Es ist daher fraglich, ob Datensätze, die schon anderen Daten zugeordnet sind, überhaupt durch Reimporte geändert werden dürfen. Bei Verwendung der Benutzeroberfläche lässt sich dieses Problem durch Vergabe von Rechten umgehen. Nicht jeder Benutzer soll alles können dürfen, im Idealfall existieren Experten, die für die Qualität der ihnen zugewiesenen Daten verantwortlich sind. Diese dürfen, einmal gesperrt, von niemanden mehr geändert werden, außer vom Experten selbst.

Diese Problematik ist bis zum momentanen Zeitpunkt noch nicht ausreichend geklärt.

Als Provisorium werden nur demjenigen Änderungsrechte zugestanden, der den Datensatz als Erster importiert hat, sofern dieser nicht schon vorher durch Verwendung der Benutzeroberfläche eingefügt wurde und er nicht weiteren Daten zugeordnet ist, die von jemand anderem als diesem ersten Datenlieferanten stammen. Andernfalls kann ein Verändern der Daten nur manuell mittels der Benutzeroberfläche erfolgen.

## 6.5 IDs

In den vorangegangenen Abschnitten wurde gefordert, dass jeder zu importierende Datensatz mit einem eindeutigen Schlüssel zu versehen ist. Im Folgenden soll diskutiert werden, wie diese Eindeutigkeit erreicht werden kann.

Zunächst einmal muss der Begriff „eindeutig“ an dieser Stelle relativiert werden: Wirklich eindeutige Schlüsselwerte erhält man nur, wenn diese durch entsprechende Programme automatisch generiert werden. Sobald eine menschliche Komponente hinzukommt, kann die Eindeutigkeit nicht garantiert werden[7]. Auch könnten verschiedene Verfahren zu demselben Schlüsselwert kommen. Damit sollte die Eindeutigkeit eines solchen Wertes immer im konkreten Kontext des Schlüssels betrachtet werden. Demnach muss also z. B. eine *TaxonId* in der SysTax-Taxonomieschnittstelle nur für alle *TaxonIds* eindeutig sein; ob jemand denselben Wert z. B. als Dateinamen für seine selbst aufgenommenen Bilder verwendet, ist für die Eindeutigkeit der *TaxonId* ohne Belang.

Ein eindeutiger Schlüsselwert, auch *globally unique identifier* genannt, dient prinzipiell dazu, ein Objekt eindeutig zu identifizieren. Dabei kann es sich um eine Datei, einen Datensatz oder sogar um einen real existierenden Gegenstand, wie etwa ein Sammlungsobjekt, handeln. Lässt sich die Menge aller Objekte in disjunkte Bereiche oder Klassen untergliedern, kann ein Schlüssel in zwei Teile getrennt werden: Ein Teil besteht aus einer meist von einer zentralen Institution vergebenen, eindeutigen Kennzeichnung für jeden dieser Bereiche. Dieser Teil kann somit als „Klassenkomponente“ oder „Klassenkennung“ des Schlüssels betrachtet werden. Der zweite Teil eines solchen Schlüsselwertes muss dann nur innerhalb des jeweiligen Bereiches eindeutig sein, dem er zugeordnet ist. Dies lässt sich wesentlich einfacher realisieren als eine Eindeutigkeit über alle Bereiche, weshalb dieser zweite Bestandteil auch als „lokale Komponente“ eines eindeutigen Schlüssels betrachtet werden kann.

Jeder dieser beiden Teile für sich ist als Schlüsselwert nicht eindeutig: Wenn nur der „lokale Teil“ gegeben wäre, könnte derselbe Schlüssel in verschiedenen Bereichen generiert werden. Falls nur die „Klassenkomponente“ verwendet würde, wären alle Schlüssel, die im selben Bereich erstellt werden, identisch. Beide Angaben zusammen, in geeigneter Form kodiert und zusammengeführt, ergeben schließlich einen global eindeutigen Wert. Diese Aufteilung ist vor allem bei „verteilten Systemen“ wichtig, bei denen verschiedene Institutionen unabhängig voneinander Schlüsselwerte generieren. Jede Institution

[7] Verschiedene Personen könnten sich dieselben Werten einfallen lassen – ein von Hand eingegebener Schlüssel könnte theoretisch denselben Wert besitzen wie ein automatisch generierter.

braucht sich dann nur um die Eindeutigkeit des Schlüssels innerhalb ihres eigenen Bereiches zu kümmern, global eindeutig wird der Wert durch Hinzufügen der Kennung für die jeweilige Institution.

Die „Klassenkennung“ eines eindeutigen Schlüssel sollte den Bereich (also z. B. den Standort, den Computer, das Projekt) während der Schlüsselgenerierung eindeutig identifizieren. Unter der Voraussetzung, dass der Computer, auf dem der Schlüssel erzeugt werden soll, an ein Netzwerk angeschlossen ist, könnte für die Klassenkomponente z. B. die so genannte *MAC*-Adresse (*Media Access Control*) verwendet werden. Diese Adresse dient zur eindeutigen Identifizierung eines jeden Gerätes im Netzwerk. Sie ist meist durch die Hardware gesetzt und unveränderbar. Im Fall von Netzwerkkarten setzt sich die MAC-Adresse zusammen aus einer Herstellerkennung, die von einer zentralen Kontrollinstanz, dem *Institute of Electrical and Electronics Engineers*[8] (*IEEE*), vergebenen wird und einem eindeutigen Bezeichner, den jeder Hersteller für jede seiner Karten vergibt. Damit ist die MAC-Adresse für sich alleine auch schon ein global eindeutiger Schlüsselwert, unterteilt in eine Klassenkomponente (der Herstellerkennung) und einer lokalen Komponente (dem vom Hersteller vergebener Bezeichner).

Andere räumlich eindeutige Kennzeichnungen für einen Computer sind denkbar. So könnte etwa die IP-Adresse verwendet werden, unter der Bedingung, dass der Rechner direkt mit dem Internet verbunden und seine IP somit fest und eindeutig ist.

Die „lokale“ Komponente ist meistens vom Zeitpunkt der Schlüsselerstellung abhängig, muss aber nicht unbedingt eine Uhrzeit beinhalten. Beispielsweise wäre eine fortlaufende Nummerierung legitim, sofern die Vergabe mehrfacher Nummern ausgeschlossen ist. Wird tatsächlich für diese Komponente eine Zeitangabe verwendet, so sollte sie möglichst präzise sein, etwa das Datum mit der sekundengenauen Uhrzeit als Zeichenkette oder die so genannte „Unix-Zeit“, welche als Ganzzahl die Sekunden angibt, die seit dem 01.01.1970, 00:00:00 Uhr vergangen sind.

Ein Beispiel für einen solchen global eindeutigen Schlüssel ist das so genannte *UUID*-Format (*Universally Unique Identifier*), ein von der *Open Software Foundation*[9] entwickelter Standard zur Erstellung global eindeutiger Schlüssel. Dieser Standard verwendet die weiter oben vorgestellte *IEEE*-Adresse in ihrer „Bereichskomponente“ und für den lokalen Bestandteil den Zeitpunkt der Schlüsselerstellung[10], gemessen an der Anzahl der 100-Nanosekunden-Intervalle, die seit dem 15.10.1582, 00:00:00.00 Uhr[11] verstrichen sind (Leach und Salz, 1998).

Auch in Bereichen, die primär nichts mit Informatik zu tun haben, können global eindeutige Bezeichner eine wichtige Rolle spielen, um Objekte weltweit eindeutig zu identifizieren. So gibt es etwa bei den biologischen Sammlungen Bestrebungen, jedes Sammlungsobjekt mit einer global eindeutigen Kennung zu versehen und diese als Barcode

[8] MAC-Adressen werden aus diesem Grund auch IEEE-Adressen genannt

[9] ein Gremium aus Vertretern der Softwareindustrie zur Erstellung von (Software-)Standards

[10] in manchen *UUID*-Versionen verknüpft mit einer Zufallskomponente

[11] Beginn des gregorianischen Kalenders

verschlüsselt dem Objekt anzufügen. Im Idealfall besitzt jedes Sammlungsobjekt eine innerhalb der jeweiligen Sammlung eindeutige Inventarnummer, die als lokale Komponente im Schlüsselwert verwendet werden kann. Eine weltweit eindeutige Kennzeichnung für jede Sammlung vervollständigt diesen Schlüssel zu einem *globally unique specimen identifier*. Während bei den zoologischen Sammlungen noch Standards für diese eindeutige Kennung der einzelnen Institute erarbeitet und eine zentrale Kontrollinstanz gebildet werden müssen, gibt es für botanische Sammlungen (Herbarien) den *Index Herbariorum*[12] als Kontrollorgan für die Eindeutigkeit der Akronyme aller öffentlichen Herbarien. Damit ist jeder Herbarbogen durch seine Inventarnummer zusammen mit dem Akronym seines Herbariums weltweit eindeutig gekennzeichnet.

Bei den SysTax-Importschnittstellen kann an jeden Datenlieferanten eine Kennung vergeben werden (z. B. die Abkürzung der Projektnamen), für deren Eindeutigkeit SysTax verantwortlich ist. Diese Kennung wird dann als Klassenkomponente für die Schlüssel der zu importierenden Datensätze verwendet. Der einzelne Datenlieferant ist dann lediglich für den von ihm zu vergebenen „lokalen“ Teil des Schlüssels verantwortlich.

Im einfachsten Fall, falls beim Datenlieferanten eine zentrale Datenbank vorliegt oder nur eine Instanz einer Datenbank auf einem einzelnen Rechner, könnte dieser lokale Teil aus einer fortlaufenden Nummer bestehen. Sobald sich aber ein Datenlieferant in einzelne, voneinander unabhängige Unterprojekte gliedert oder Datensätze aus verschiedenen Datenbanken stammen, gelten für die lokale Komponente dieselben Überlegungen wie für einen global eindeutigen Schlüssel. Hierfür könnte die weiter oben vorgestellte *UUID* verwendet werden, denkbar wäre auch eine eindeutige Kennung für jede Tabellen- oder Datenbankinstanz. Im letzteren Fall könnte für den zweiten Bestandteil des Schlüssels tatsächlich eine fortlaufende Nummer verwendet werden. Eine weitere Möglichkeit wäre die Verwendung einer Kennzeichnung für jeden Benutzer, unter der Voraussetzung, dass niemand mit der Kennung eines anderen arbeiten kann. Ist sichergestellt, dass jeder Benutzer in jeder Datenbankinstanz eine andere Kennung erhält und ist die Eindeutigkeit aller Kennungen dabei gewährleistet, so könnte ebenfalls für jede Instanz eine fortlaufende Nummer verwendet werden. Ist dies nicht der Fall, könnte der lokale Teil aus dem Datum und der Uhrzeit der Schlüsselgenerierung bestehen. Dabei sollte sichergestellt sein, dass tatsächlich das aktuelle Datum und die aktuelle Uhrzeit, möglicherweise über das Internet synchronisiert, verwendet wird, damit der lokale Teil bei derselben Nutzerkennung auf verschiedenen Computern nicht zufälligerweise durch eine falsch eingestellte Systemzeit doppelt vergeben wird.

Da für die Vergabe des Datensatzschlüssels und deren Eindeutigkeit ausschließlich der Datenlieferant zuständig und verantwortlich ist, liegt es in seinem Ermessen, ob eines der in diesem Abschnitt vorgestellten Verfahren zur Erzeugung eindeutiger Schlüssel tatsächlich zur Anwendung kommt und welches er bevorzugt.

[12] ein Projekt der *International Association for Plant Taxonomy* und des *The New York Botanical Garden*

# 7 Zusammenfassung

Die Speicherung der wissenschaftlichen biologischen Namen und ihrer hierarchischen Klassifikation in einer Datenbank stellt ein Problem dar, da sich ein Großteil der Namen aus den Namen verschiedener Stufen zusammensetzt. Zudem sind die Namen bezüglich ihrer Gültigkeit nicht konstant. Stellt sich beispielsweise heraus, dass zwei unterschiedliche Namen dasselbe Taxon beschreiben, so wird einer, in der Regel der jüngere von beiden, als Synonym des anderen behandelt. Solche Synonyme sind aber nicht immer eindeutig. Wissenschaftler können unterschiedlicher Meinung darüber sein, ob zwei Namen tatsächlich dasselbe Taxon umschreiben. Aber nicht nur die Gültigkeit von Namen kann sich ändern, auch die Umschreibung eines Namens ist nicht absolut festgelegt. Verschiedene Autoren können unterschiedliche Merkmale mit ein und demselben Namen bezeichnen.

In der vorliegenden Arbeit wird das von Berendsohn beschriebene Prinzip der Potential Taxa als mögliche Lösung dieser Problematik vorgestellt. Indem zu einem Namen immer eine Referenz auf eine Quelle angegeben wird, ist seine Umschreibung eindeutig bestimmt. Dadurch können verschiedene Umschreibungen eines Namens unterschieden werden und man kann direkt darauf Bezug nehmen. Ebenso lassen sich auf diese Weise unterschiedliche, möglicherweise sogar widersprüchliche Meinungen über die Gültigkeit eines Namens, seiner Synonyme und sogar seiner Klassifikation speichern.

Um das beschriebene Prinzip umzusetzen, wird ein Modell entwickelt, das die Speicherung von Potential Taxa in einer relationalen Datenbank ermöglicht. Angelehnt an das von Berendsohn Mitte der neunziger Jahre vorgestellte IOPI-Modell, behandelt das vorliegende Modell die Potential Taxa als Verknüpfung der Entität der wissenschaftlichen Namen mit der Entität der Quellen. Als Folge davon, dass ausschließlich Potential Taxa miteinander verknüpft werden dürfen und nicht die Namen selbst, ist eine beliebige Anzahl verschiedener taxonomischer Vorgänger sowie Synonymlisten zu ein und demselben Namen speicherbar.

Die Einführung eines neutralen Potential Taxons ermöglicht einen problemlosen Umgang mit der Komplexität, die die parallele Speicherung verschiedener taxonomischer Konzepte mit sich bringen kann. Hierdurch kann nicht taxonomisch orientierten Nutzern, denen entsprechende taxonomische Daten häufig nicht vorliegen, die Notwendigkeit zur Wahl eines bestimmten Potential Taxons eines Namens abgenommen werden.

Verschiedene Möglichkeiten zur Realisierung dieses Modells werden betrachtet, insbesondere bezüglich der Namensverwaltung, sowie ihre Vor- und Nachteile diskutiert. In Anlehnung an die für den botanischen Programmteil verwendete Tabellenstruktur wird hier eine Form der Umsetzung gewählt, bei der Namen auf der Ebene einer Unterstufe

zusammen mit den Namen ihrer höheren Stufen bis hin zur Hauptstufe als elementarer Eintrag betrachtet werden.

Die Digitalisierung biologischer Daten beinhaltet aber nicht nur die Verwaltung der wissenschaftlichen Namen sowie der botanischen oder zoologischen Sammlungsbelege, sondern auch die Anfertigung und Bereitstellung multimedialer Inhalte. Dazu wurde in SysTax die Möglichkeit geschaffen, Bilder, Töne und sonstige Dokumente speichern und den einzelnen Taxa sowie Sammlungsbelegen zuordnen zu können. Zu diesem Zweck stellt Oracle den Datentyp *binary large object* zur Verfügung, dessen Verwendungsmöglichkeiten und dessen physikalische Speicherung vorgestellt werden. Um auf eine möglichst flexible Weise multimediale Daten einem oder mehreren Taxa oder Belegen zuordnen zu können, wird in dieser Arbeit ein Datenbankmodell entwickelt, das diese Verknüpfungen zu „Informationscontainern“ gruppiert.

Das Datenbanksystem SysTax war ursprünglich auf die Verwendung durch botanisch arbeitende Nutzer ausgerichtet. Die Notwendigkeit zur Implementierung eines zoologischen Programmteils entstand mit der Entscheidung innerhalb des EDIS-Projekts, die erhobenen Daten via SysTax im Internet zu veröffentlichen. Zu diesem Zweck wurden in Zusammenarbeit mit den einzelnen Projektpartnern Schnittstellen definiert, über die die erfassten Daten an das SysTax-Projekt geliefert und in die Datenbank importiert werden können. Die verschiedenen Möglichkeiten der äußeren Form dieser Schnittstellen, ob im XML- oder CSV-Format, werden in dieser Arbeit vorgestellt und hinsichtlich ihrer Vor- und Nachteile diskutiert. Letztendlich hat man sich im EDIS- und später im GBIF-Projekt auf flache CSV-Schnittstellen geeinigt, um den Datentransfer so einfach wie möglich zu halten.

Da Daten aus unterschiedlichen Bereichen, wie etwa Taxonomie, Literatur, Sammlungsbelege oder Bilder, ausgetauscht werden, ist trotz der Verwendung flacher ASCII-Dateien eine inhaltliche Trennung dieser Bereiche unumgänglich. Dadurch ergibt sich die Notwendigkeit, Verknüpfungen von Daten aus verschiedenen Bereichen analog einem relationalen Datenbankschema mittels Datensatzschlüssel zu realisieren. Eine wichtige Voraussetzung für dieses Verfahren ist die Vergabe global eindeutiger Schlüssel. Die Problematik der Erstellung solcher eindeutiger Schlüsselwerte wird aufgezeigt sowie Mechanismen zu ihrer Generierung vorgestellt.

Gefördert wurde SysTax in den letzten Jahren im Rahmen eines Projekts, das es sich zur Aufgabe gemacht hat, international Daten zur Artenvielfalt zu erheben und durch Veröffentlichung im Internet weltweit zur Verfügung zu stellen. Lag bisher der Schwerpunkt auf der Erhebung und Sammlung von Daten, so wird in Zukunft das Hauptaugenmerk sicherlich auf der Auswertung derselben liegen, um Trends in der Entwicklung der Biodiversität zu erkennen. Ein Hilfsmittel hierzu stellen sicherlich so genannte Geoinformationssysteme dar, mit denen es möglich ist, durch die Visualisierung von Fundorten Verbreitungskarten von Arten, Gattungen oder ganzen Familien zu konstruieren. Auch der Datenbestand in SysTax kann, vor allem durch die Vielzahl historischer Sammlungsbelege, hierzu sicherlich seinen Beitrag leisten.

# 8 Summary

SysTax, a database application, had originally been designed to store botanical taxonomic data, specimen data of herbaria and accession data of botanical gardens. In 2000, EDIS project members decided to use SysTax in order to publish their collected zoological specimen data via the Internet. Due to these extended requirements, the database had to be upgraded to enable the handling of zoological taxonomic data.

This exigence had been taken as a chance to implement the principle of potential taxa as published by Berendsohn. A potential taxon consists of a taxon name and a source assigning a state to that name, which indicates whether the name is accepted in this specific source or not. By these means it is possible to store alternative opinions on a name, its synonyms, and its classification.

In this thesis a data model is developed for the use of potential taxa in databases. Furthermore, several ways of realisation of this model are discussed, especially concerning the administration of scientific names.

Nowadays, storing taxonomical or collection data is not the only requirement in a biological database system. Gathering and publishing multimedia contents about taxa or specimens is of growing importance. Therefore, it has been enabled to manage images, sounds, and other documents with SysTax and to assign them to taxa and collection items. For this purpose, Oracle provides the data type *binary large object*. The possibilities of its use and physical storage are presented in this thesis. Moreover, a database model is developed for the assignment of multimedia data to one or more taxa or specimens in a way that is required to be as flexible as possible. In the present model, these relationships are grouped to 'information containers'.

Due to the fact that many project partners go on running databases of their own, a mechanism to transfer data to the SysTax database was necessary in order to keep those databases up-to-date. Interfaces had been defined in collaboration with the project partners to exchange those data. The different possibilities for the technical design of such interfaces, whether as XML or as CSV files, are presented and their advantages and disadvantages are discussed. The project partners agreed on the use of flat text files in CSV style for the data exchange format.

As data of different fields are exchanged via these interfaces and despite of the use of a flat text format, these miscellaneous domains had to be separated from each other. Thus data of different fields have to be linked to each other with unique identifiers, comparable to a relational database schema. Possible ways of generating such unique identifiers are also presented and discussed in this thesis.

# Literaturverzeichnis


Behme, Henning; Mintert, Stefan: *XML in der Praxis.* Zweite Auflage. 2000

Berendsohn, Walter G.: The concept of „potential taxa“ in databases. In: *Taxon* 44 (1995), S. 207–212. – ISSN 0040-0262

Berendsohn, Walter G.: A taxonomic information model for botanical databases: the IOPI Model. In: *Taxon* 46 (1997), S. 283–309. – ISSN 0040-0262

Boos, Evelin: *Botanische Klassifikation und Taxonomie. Konzeption und Realisierung eines Informationssystems*, Universität Ulm, Fakultät für Mathematik und Wirtschaftswissenschaften, Dissertation, 1992

Fowler, Martin: *UML distilled: a brief guide to the standard object modeling language.* Third edition. Addison-Wesley Boston, San Francisco, [u.a.], 2004. – ISBN 0-321-19368-7

GPC-Homepage: *Homepage der Global Plant Checklist.* – URL `http://bgbm3.bgbm.fu-berlin.de/iopi/gpc/`. – Zugriffsdatum: 20.01.2003

Greuter, W.; McNeill, J.; Barrie, H.-M.; Demoulin, V.; Filgueiras, T. S.; Nicolson, D. H.; Silva, P. C.; Skog, J. E.; Trehane, P.; Turland, N. J.; Hawksworth, D. L.: *Indernational Code of Botanical Nomenclature (St. Luis Code).* Regnum Vegetabile 131. Koeltz Scientific Books, Königsstein, 2000. – ISBN 3-904144-22-7

International Commission on Zoological Nomenclature: *International Code of Zoological Nomenclature.* Fourth Edition. January 2000. – ISBN 0 85301 006 4

IOPI-Homepage: *Homepage der International Organisation for Plant Information (IOPI).* – URL `http://plantnet.rbgsyd.gov.au/iopi/iopihome.html`. – Zugriffsdatum: 20.01.2003

Jeckle, Mario: *Entwurf von XML-Sprachen.* 2000. – URL `http://www.jeckle.de/entwurfxml/entwurfxml.html`. – Zugriffsdatum: 02.08.2004

Koletzke, Peter; Dorsey, Paul D.: *Oracle Developer - Advanced Forms & Reports.* Osborne/McGraw-Hill, Berkeley, New York [u.a.], 2000. – ISBN 0-07-212048-7

KOPERSKI, Monika; SAUER, Michael; BRAUN, Walther; GRADSTEIN, S. R.: *Referenzliste der Moose Deutschlands.* Bundesamt für Naturschutz, 2000 (Schriftenreihe für Vegetationskunde 34)

KOPKA, Helmut: *LaTeX Einführung Band 1.* 3. überarb. Aufl. Addison-Wesley, München, Boston [u.a.], 2000. – ISBN 3-8273-1557-3

LEACH, Paul J.; SALZ, Rich: *UUIDs and GUIDs.* 1998. – URL `http://hegel.ittc.ukans.edu/topics/internet/internet-drafts/draft-l/draft-leach-uuids-guids-01.txt`. – Zugriffsdatum: 02.08.2004

LONEY, Kevin; KOCH, Geroge: *Oracle8i - Die umfassende Referenz.* Carl Hanser Verlag, München, Wien, 2001. – ISBN 3-446-21570-0

MABBERLEY, D.J.: *The Plant-Book, a portable dictionary of the vascular plants.* Cambridge, 1997. – ISBN 0 521 41421 0

MUENCH, Steve: *Building Oracle XML Applications.* O'Reilly, Beijing, Campridge, [u.a.], 2000. – ISBN 1-56592-691-9

NEUMANN, Horst A.: *Objektorientierte Softwareentwicklung mit der Unified Modeling Language.* Hanser Verlag, München, Wien, 1998. – ISBN 3-446-18879-7

PULLAN, Martin R.; WATSON, Mark F.; KENNEDY, Jessie B.; RAGUENAUD, Cédric; HYAM, Roger: The Prometheus Taxonomic Model: a pracitcal approach to representing multiple classifications. In: *Taxon* 49 (2000), S. 55–75

RUMPE, Bernhard: *Modellierung mit UML.* Springer Verlag, Berlin, Heidelberg, 2004. – ISBN 3-540-20904-2

SCHWEIGGERT, Franz: *UNIX-basierte Implementierung kleiner Datenbanken I.* 1997. – Vorlesungsskript, Universität Ulm, Fakultät Mathematik u. Wirtschaftswissenschaften, Abteilung Angewandte Informationsverarbeitung

SCHWEIGGERT, Franz: *Software Engineering Praxis.* 2003. – Vorlesungsskript, Universität Ulm, Fakultät Mathematik u. Wirtschaftswissenschaften, Abteilung Angewandte Informationsverarbeitung

SIVARAJAN, V.V.; ROBSON, N.K.B. (Hrsg.): *Introducation to the principles of plant taxonomy.* Second edition. Cambridge University Press, Cambridge, New York, [u.a.], 1991. – ISBN 0-521-35679-2

STACE, Clive A.: *Plant taxonomy and biosystematics.* Second edition. Cambridge University Press, Cambridge, New York, [u.a.], 1989. – ISBN 0-7131-2955-7

VOSSEN, Gottfried: *Datenmodelle, Datenbanksprachen und Datenbank-Management-Systeme.* 2. aktualisierte und erweiterte Auflage. Addison-Wesley Publishing Company, Bonn, Paris [u.a.], 1994. – ISBN 3-89319-566-1

Wahl, Günter: UML kompakt. In: *OBJEKTspektrum* (1998). – URL `http://www.sigs-datacom.de/sd/publications//os/1998/02/OBJEKTspektrum_UM_kompakt.htm`

Wikipedia-Homepage: *Wikipedia, die freie Enzyklopädie.* – URL `http://de.wikipedia.org/wiki/Hauptseite`. – Zugriffsdatum: 07.07.2004

Winston, Judith E.: *Describing Species, Practical Taxonomic Procedure for Biologists.* New York, 1999

# Abbildungsverzeichnis

# Tabellenverzeichnis