

Euro+Med PlantBase

Design of the Internet Taxonomic Sector Editor

Version 1.5, June 2002

*Anton Güntsch, Jinling Li & Walter Berendsohn
Botanic Garden and Botanical Museum Berlin-Dahlem
Dept. of Biodiversity and Laboratories
<http://www.bgbm.org/BioDivInf/default.htm>
Königin-Luise-Str.6-8
14191 Berlin
Germany*

1 Introduction

The production of the Euro+Med plantbase is done in two partly parallel phases: Within a first phase, existing datasets are made available electronically and merged into a single database held by the project's secretariat in Reading. This resulting dataset is the basis for further evaluation (WP3), which finally leads to the initial Euro+Med checklist. Within a second phase (WP4), selected taxonomic groups will be revised and enriched with factual data by taxonomic experts and re-incorporated into the checklist. The format of revisions is defined in the "Guidelines for contributors of initial taxonomic accounts"¹ to make sure that the resulting data sets are compatible with the Euro+Med data standards.

Currently, taxonomic revisions are carried out offline in an electronic or traditional, paper based way. As a consequence, the incorporation of revision into the central checklist database is a time consuming task. This could be avoided to a large extent if revisers would enter their data directly into a central dataset, which provides catalogue data such as author names or bibliographic records. For this reason, the Euro+Med project dedicated two tasks (2.3, 2.4) to the development of a "prototype Internet taxonomic editor system" to enable revisers to edit "shadow" copies of taxonomic sectors over the World Wide Web.²

This document describes technical principles and user interface design of the Internet editor. It is based on the Euro+Med revision guidelines, earlier reports on the editor design distributed within the computer working group and Executive committee, discussions within the Euro+Med computing group, discussions with members of related projects (e.g. MoreTax³, Glopp⁴, and AlgaTerra⁵), as well as the experiences gained from a prototype implementation.

¹ http://www.euromed.org.uk/documents/4.6.01_revision_guidelines.pdf

² The software is not intended to serve as a maintenance tool for the central Euro+Med database nor as a publication tool. Nevertheless it can be designed in a way that makes its modules reusable for purposes outside from its original scope.

³ Modelling of rule-based functions for a computerized system to link complex taxonomic concepts with factual data of relevance to nature conservation (<http://www.bgbm.org/BioDivInf/Projects/MoreTax/default.htm>)

⁴ Global Information System for the Biodiversity of Plant Pathogenic Fungi (<http://160.45.63.11/>)

⁵ AlgaTerra: An information system for terrestrial algal biodiversity (www.algaterra.net)

2 Software platform

2.1 System architecture

The implementation of a remote editing tool for a centralized taxonomic database system can be based on either of the following two principles:

- The client program is a piece of software specifically designed and implemented for the editing task.
- The client is realized with a piece of software (typically a World Wide Web browser), which is installed on the majority of computers anyway. With this approach, all data and the entire set of taxonomic rules are located at the server's side and data entry forms are dynamically created.

The second option is considered preferable for the needs of Euro+Med because the client software will run on any operating system and authors of revisions will not have to install special software. Moreover, a user interface created dynamically on the basis of html could be modified and reused for the publication of data.

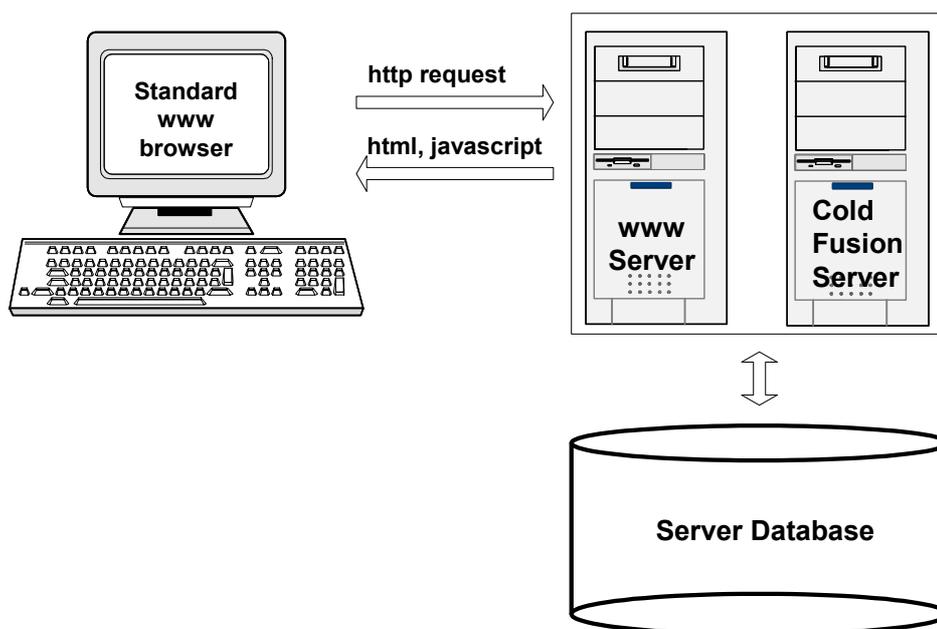


Fig.1: System Architecture

To support the development of a server side editing software, it was decided to use Macromedia “Cold Fusion”⁶, a server especially designed for the implementation of highly dynamical database-driven web applications (fig.1). Cold Fusion applications run with all major web servers (e.g. IIS and Apache) and on various operating systems such as Microsoft NT and Linux.

⁶ <http://www.macromedia.com/software/cfudstudio/>

The underlying database should be a relational database management system capable of processing stored procedures, functions, and triggers for the implementation of a taxonomic business layer at database level (see 2.3). Currently, the prototype implementation makes use of Microsoft SQL Server 2000⁷.

2.2 Client software requirements

Although the system architecture is designed in a way that makes it possible to carry out a revision with almost any World Wide Web browser, the browser configuration has to be specified as a stable basis for the server software:

- Rudimentary form field validation and the handling of events such as the alteration of a botanical name are best handled with JavaScript. Therefore, the client browser should be JavaScript compliant.⁸
- One major obstacle when implementing a server side web application is the fact that the http-protocol does not support states, which means that the server does not “know” what a user did before he or she accessed a form within the application. This obstacle can be overcome by either simulating the states by adding certain parameters to the URL (difficult to maintain and therefore not recommended) or by invoking server side session variables corresponding to cookies on the client computer. Consequently, the client browser should be configured to accept cookies.

2.3 Software layers

An important design issue for the implementation of the taxonomic remote editor is the assignment of system functionality to software modules or layers. This consideration directly influences maintainability, reusability, reliability, and performance of the application. The Euro+Med Computer Working Group considered a three-layer architecture appropriate for the purposes of a taxonomic editor application (Fig.2).

⁷ <http://www.microsoft.com/sql/default.asp>

⁸ Java applets will not be used. We think that applet based systems still tend to be unstable. Furthermore, insufficient response times which are prevalent especially when working with low bandwidth connections could reduce the user's acceptance of the system.

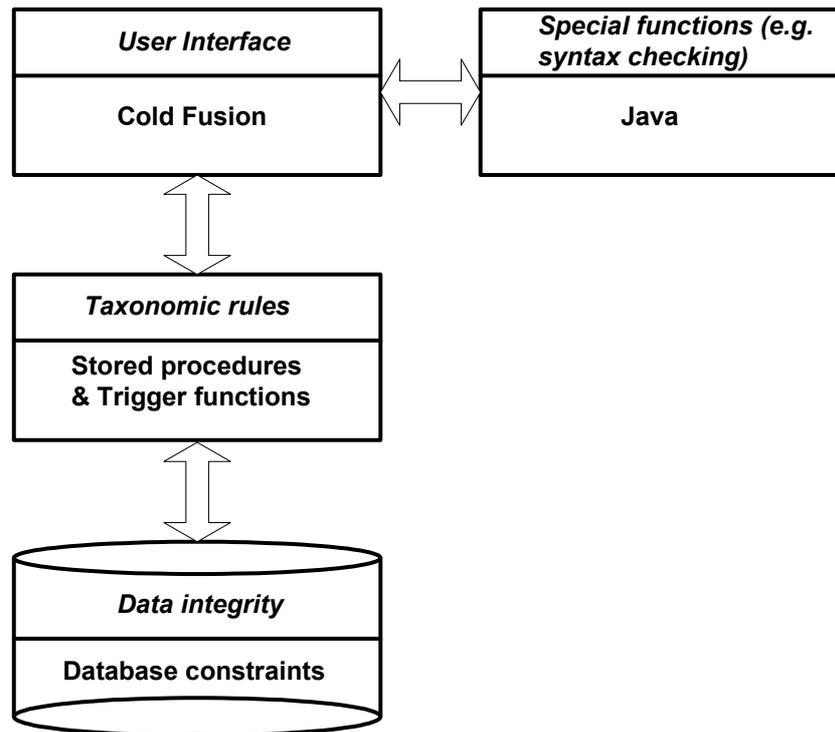


Fig.2: Software Layers

Basic data integrity rules are implemented at the level of tables, keys, and relations within the database kernel. For example, the fact that every botanical name should have a rank can be assured with a foreign key to the table defining the list of valid ranks (see ER diagram in the appendix).

More complex rules and functions, which for example are used to construct syntactically correct botanical names, are implemented with stored procedures and trigger functions. Stored procedures are used to implement complex functions at database level and to define abstract interface functions to avoid that external program modules have to “know” the internal database properties when retrieving data. Triggers are functions, which are performed automatically when certain events occur. E.g., the editor prototype implementation contains triggers that automatically rebuild author teams when one of the relevant author names was changed.

The User interface implementation (Cold Fusion) is exclusively dedicated to control the user activities and strictly separated from the taxonomic logic of the system. This has been achieved by implementing the majority of functions at the level of stored procedures. Functions that do not require data from the taxonomic database should not be implemented as stored procedures; they are programmed as server-side java classes.

3 Data model

The first step in the implementation of a database driven application is the definition of an appropriate data model, which is complex enough to meet the needs of the application and at the same time simple enough to be usable. The underlying model of the remote editor “backbone database” should support the Euro+Med data standards. It should also be capable of representing different taxonomic views for the same taxonomic group, in order to hold shadow copies of checklist sectors undergoing

revision, and in order to enable the system to express arbitrary relations between taxonomic views on taxa.

To meet these requirements, the Berlin partners developed in co-operation with several other projects a novel data structure, which has been distributed, discussed in the project and was implemented as an SQL Server database to serve as the backbone for the current editor prototype implementation.

The model closely follows the Euro+Med specifications as given in the “Guidelines for contributors of initial taxonomic accounts” (http://www.euromed.org.uk/documents/4.6.01_revision_guidelines.pdf).

Nevertheless, different taxonomic data requirements were considered in the construction of the model. For example, if deemed necessary hybrid formulas and type identifications can be accommodated either by appending a separate table or by the inclusion of additional fields in the *Name* and *NameHistory* tables.

The latest version of the model documentation is available at (<http://www.bgbm.org/biodivinf/docs/bgbm-model/>).

3.1 Central entities

Name, NameHistory, NomStatusRel, and RelName

To preserve the history of the name editing process, names are stored in two tables *Name* and *NameHistory*. The *Name* table contains the set of names currently in use by a system. The *NameCache* and *FullNameCache* fields contain the Latin names without and including author strings as calculated from their component fields by a trigger functions according to nomenclatural rules valid for a specific project. In addition, these cache fields can be used to capture preliminary (unstructured) names if needed (e.g. in imports). The *NameHistory* table is structured very similarly with a few exceptions: the name string fields preserve their value and *AuthorTeam* as well as *BasAuthorTeam* are carrying their respective content as strings and not as a pointer to the author tables. A self-referential pointer (*SuccNameHistId*) links a name to its successor within the editing history. An additional pointer (*CurrentNameId*) gives for every name the name currently in use. (N.B.: this does not represent a synonymic relationship, but just a history of edits to the name itself). The *NomStatusRel* table is used to assign one or several nomenclatural status values (e.g. 'nom. illeg.' or 'nom. nud.') to a name. The catalogue of name status values for a given project is defined in the *NomStatus* table. To express relations between taxon names considered to be nomenclatural rather than a taxonomic opinion the *RelName* table can link arbitrary entries of the name table. The type of this relation (e.g. 'is later homonym of' or 'is basionym for') is indicated with a pointer to the *RelNameQualifier* table.

References

References of any kind (e.g. nomenclatural references, references for taxonomic opinions, factual data references) are accommodated in a single recurrent structure reference. By using the self referential pointer *InRefFk* references can be defined as part of references in the same table (e.g. article as a part of a given book). The type of a reference is indicated with a pointer (*RefCategoryFk*) to the *RefCategory* table which defines the list of reference types. Since all attributes for the different reference types are accommodated in a single entity, a client program has to decide which attributes to use for a given reference category. Trigger functions may be used to support this process and to preserve data integrity.

Citations are created with the RefDetail table which contains a pointer to the Reference table and a field (Details) which accommodates the exact position in the reference as a string (e.g. page number, table number, record id in a database).

PotTaxon and RelPTaxon

The PotTaxon table's primary key combines name identifiers with reference identifiers. This combination enables the system to preserve statements on taxa as they were given in the original information sources (e.g. status of a name or distribution information). The RelPTaxon table is used to express directed binary relations between any two entries of the PTaxon table. This includes taxonomic inclusions (i.e. the classification system) and any kind of synonym and concept relations. The type of the relation is specified in the table RelPTQualifier (as indicated with RelQualifierFk). The source of the relationship is indicated by the attribute RelRefDetailFk. Holding all kinds of binary relations in a single table provides a convenient and transparent way to query the links between taxa, and it greatly facilitates the implementation of client-sided navigation functions.

Factual data

The model links factual data of any kind to the *PotTaxon* table thus providing a means to express different "opinions" for different versions of the checklist on the same name. Occurrence data are kept in an atomised table (*Occurrence*) as specified by the project. All other factual data sets are stored as summaries in the *Fact* table as free text and are distinguished with the *FactTypeFk* pointer to the *FactType* table. As soon as more detailed specifications for these data areas became available, appropriate data structures could be developed and directly linked to the *PotTaxon* table.

3.2 Creating "shadow copies"

Although all data will be held centrally at the Euro+Med secretariat site, the remote editor will work on "shadow copies" of specific taxonomic groups of the latest checklist version. Therefore, the system will have to create or duplicate certain entities of the underlying database whenever a new user (author) receives permission for revising a taxonomic group. This allows to follow the editing history within the project and to compare original and revised treatments on the World Wide Web.

Nevertheless, it has to be decided carefully which attributes should be duplicated or newly created before a revision is carried out and which attributes should stay globally unique because they belong to catalogue data and should therefore be agreed on.

The computer working group considers the following division of entities appropriate:

Globally unique data	Duplicated data	New created data
<ul style="list-style-type: none"> - Taxon names - Nomenclatural reference titles (books, journals) - Author names 	<ul style="list-style-type: none"> - Name status - Source citations (e.g. factual data sources) - Factual data 	<ul style="list-style-type: none"> - Name-reference pairs and relations between name-reference pairs with the new author as reference

A stored procedure has been programmed which automates the creation of shadow copies according to this division. It is currently used to create example data for the editor prototype implementation.

4 Forms

4.1 Basic principles

The design of the **Taxonomic Sector Editor** forms is based on the following principles:

Easy to use -- In order to make the **Taxonomic Sector Editor** software as easy as possible to understand and use, the form design is as much as possible based on the paper forms as specified in the “Revision Guidelines V1.2”. This will encourage revision authors used to working with paper forms to move to the web-based version.

Small forms – One critical bottleneck for an Internet application is the data transfer from the server to the client (browser) required for populating forms (e.g. a select box containing thousands of author names). Therefore all forms are designed in a way that the respective size does not exceed 50KB. This is achieved by avoiding graphics and by populating data intensive form fields with relevant data only instead of transmitting the entire data lists (e.g. author lists and bibliographic records).

Dynamic forms – The User should only see the relevant fields and options for a given context. Therefore, forms should appear differently for different “situations”. When entering a name for example, only the relevant fields for the selected ranks are displayed. If the user decides to alter the rank, the form changes its appearance immediately.

4.2 Logging in

The Login form (Fig. 3) is used to protect the system from unauthorized access. Once the author of a revision entered the correct username/password combination, the central form for navigation is displayed and access to his or her taxonomic group (“shadow copy”) to be revised is allowed.

Euro+Med Taxonomic Editor

EURO+MED
PlantBase

Username:

Password:

Version 0.97 (01/2002) - [A.Güntsch](#), [J.Li](#) & [W.G.Berendsohn](#)
[Botanic Garden and Botanical Museum Berlin-Dahlem, Dept. for Biodiversity Informatics](#)

Fig. 3: Login Form

Initially, the top node (highest taxon) of this group is focussed. All taxa not belonging to this “shadow copy” are not accessible.

4.3 Central form and navigation

The central form (Fig. 4) of the editor serves three purposes. It is used (i) to summarize as much data as possible on a single page, (ii) to navigate through the

Euro+Med Taxonomic Editor **EURO+MED**
PlantBase

[Search for taxon](#) [Logout](#) [Charsets](#) [Submit revision](#)

Name:	<i>Pitularia townsendii</i> Pomel	Edit
Nomencl. ref.	Sp. Umb.	Edit
Rank:	Species	
Taxonomy:	Genus: Pitularia L. Subgenus: Pitularia subgen. Rabata Bolle	
Status:	accepted	Alter
Basionym:		Add
Synonymy:	Pisum sativum Clairv. [synonym] Add synonym or misapplied name	Del
Homonyms:	Pitularia townsendii Biv. Add homonym	Del
Included Taxa:	subsp. Pitularia townsendii subsp. arvense Dandy Add included taxon	Del
Factual Data:	Description	Add
	Growthform	Add
	Hardiness	Add
	Ecology	Add
	Phenology	Add
	Karyology	Add
	Illustration	Add
	Identification key to included taxa	Add
	Observation	Add
	General distribution - Euro+Med	Add
	General distribution - world	Add
	Distribution	Edit

Version 0.97 (01/2002) - [A. Guntsch](#), [J. Li](#) & [W.G. Berendsohn](#)
Botanic Garden and Botanical Museum Berlin-Dahlem, Dept. for Biodiversity Informatics

Fig. 4: Main Form

taxonomic tree, and (iii) it gives access to all functions relevant for the focussed taxon. The central form consists of the following sections:

- Name, nomenclatural reference, and rank for the focussed taxon.
- Taxonomic classification, which can also be used to move the focus of the form to a higher taxon.
- Status of the name of the focussed taxon.
- Basionym of the focussed name.
- A list of synonyms if the focussed name is accepted
- An accepted name if the focussed name is a synonym or a misapplied name
- A list of accepted names if the focussed name is a sensu-lato synonym
- A link to a misapplication from a synonym or an accepted name, if the name exists both as a misapplication and as a correct name or synonym in the database.
- A list of later homonyms if the focussed name is accepted.

- For later homonyms, a link to the legitimate application of the name (Art. 53.1 St. Louis Code).
- A list of included taxa of lower rank, which can also be used to move the focus downwards within the taxonomic tree.
- A list of links to factual data display and editing templates. In order to speed up the navigation process, factual data will not be displayed in the central form.

At present, the header part includes the search function and logout function. The search function is used to find taxa directly without having to navigate through the taxonomic tree. The search form and the search result form are shown as Fig. 5 and Fig. 6.

Fig. 5: Taxon Search Form

Full name	Rank	Status
Pilularia L.	Genus	provisionally accepted
Pilularia Zotov	Genus	homonym
Pilularia subgen. Rabiata Copel.	Subgenus	homonym
Pilularia townsendill Bry.	Species	homonym
Pilularia globulifera L.	Species	accepted
Pilularia minuta Durieu ex A. Braun	Species	accepted
Pilularia subgen. Rabiata Bolle	Subgenus	accepted
Pilularia townsendill Pomet	Species	accepted
Pilularia townsendill subsp. arvensis Dandy	Subspecies	accepted

Fig. 6: Search Result Form

4.4 Adding and editing accepted taxa

There are two separate forms for adding and editing an accepted taxon. Forms for adding new data and editing existing data will change their layout depending on the context. For example, if the rank of a name is given as species, only the fields relevant for this rank will be displayed, irrelevant fields such as infraspecific epithet will

The screenshot shows the 'Euro+Med Taxonomic Editor' interface. At the top right is the logo for 'EURO+MED PlantBase'. Below the header are navigation links: 'Search for taxon', 'Logout', 'Charsets', and 'Submit revision'. The main form area is titled 'Edit name' and contains several input fields:

- 'Select rank:' with a dropdown menu set to 'Species'.
- 'Genus name:' with a text box containing 'Pilularia' and a checkbox for '(gen. hybr.)'.
- 'Specific epithet:' with a text box containing 'townsendii' and a checkbox for '(sp. hybr.)'.
- '(Basionym authors:)' with a dropdown menu showing 'no selection'.
- 'Authors:' with a dropdown menu showing 'Parnell'.
- 'New authorteam' with a link.
- 'Notes' with a large text area.

 At the bottom of the form are three buttons: 'Check', 'Apply', and 'Cancel'. Below the form, the version information is displayed: 'Version 0.97 (01/2002) - A.Güntsch, J.Li & W.G.Berendsohn, Botanic Garden and Botanical Museum Berlin-Dahlem, Dept. for Biodiversity Informatics'.

Fig. 7: Edit an accepted taxon

disappear. When adding or editing a taxon name (see Fig. 7), there is a check function that decides whether the name entered into the form fields follows the Euro+Med rules syntactically. For example, in the Genus name field an entry string like “pilularia” will result in a small window opening with a prompt (see Fig. 8) that the syntax is not correct after pressing the check button.

The screenshot shows a message box with a yellow background. At the top center is a large exclamation mark '!'. Below it, the text reads: 'Genus name does not start with an upper case character followed by lower case characters.' At the bottom center is a button labeled 'Back to Edit'.

Fig. 8: Message form for syntax checking

Nevertheless, the system does not “overrule” the decision of the author of a revision. Therefore, the user may enter incomplete or incorrect data and re-edit them later. New author teams can be created by following the “New authorteam” link. The form for the creation of author teams allows for both structured data entry by explicitly selecting from a list of authors or typing in the complete string into the “Rapid data

entry” field to be structured later (Fig. 9). It is checked (Fig. 10) whether a user accidentally entered structured and unstructured data.

Euro+Med Taxonomic Editor **EURO+MED PlantBase**

[Search for taxon](#) [Logout](#) [Charsets](#) [Submit revision](#)

Create new authorteam

Rapid data entry

Structured data entry

1st author: ▾
 2nd author: ▾
 3rd author: ▾
 4th author: ▾
 5th author: ▾

[New Author](#)

Notes:

Version 0.97 (01/2002) - [A.Güntsch](#), [J.Li](#) & [W.G.Berendsohn](#)
 Botanic Garden and Botanical Museum Berlin-Dahlem, Dept. for Biodiversity Informatics

Fig. 9: Create new authorteam form

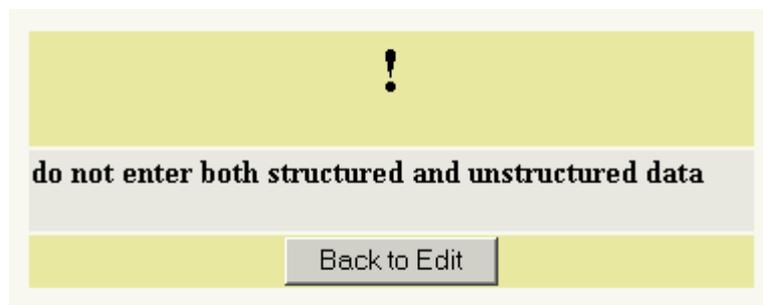


Fig.10: Syntax check message

If an author name is missing in the list of already existing author names, the user can add it by following the link “New Author” (Fig.11).

Euro+Med Taxonomic Editor



[Search for taxon](#) [Logout](#) Charsets [Submit revision](#)

Create new author

Abbreviation	<input type="text"/>
First name	<input type="text"/>
Last name	<input type="text"/>
Lifespan string	<input type="text"/>
Notes	<input style="height: 40px;" type="text"/>

Version 0.97 (01/2002) - [A.Güntsch](#), [J.Li](#) & [W.G.Berendsohn](#)
 Botanic Garden and Botanical Museum Berlin-Dahlem, Dept. for Biodiversity Informatics

Fig. 11: Create new author

The interface and functions of the form for adding a new taxon are very similar to those of the editing form, but only adding or selecting new strings are permitted instead of editing the existing strings. The interface of the form is shown as Fig. 12.

Euro+Med Taxonomic Editor



[Search for taxon](#) [Logout](#) Charsets [Submit revision](#)

Add included taxon for *Pilularia townsendill* Pomel

Select rank:	<input type="text" value="Subspecies"/>
Genus name:	<input type="text" value="Pilularia"/> <input type="checkbox"/> (gen. hybr.)
Specific epithet:	<input type="text" value="townsendill"/> <input type="checkbox"/> (sp. hybr.)
Infraspecific epithet:	<input type="text"/>
(Basionym authors:)	<input type="text" value="no selection"/>
Authors:	<input type="text" value="no selection"/>
New authorteam	
Notes	<input style="height: 40px;" type="text"/>

Version 0.97 (01/2002) - [A.Güntsch](#), [J.Li](#) & [W.G.Berendsohn](#)
 Botanic Garden and Botanical Museum Berlin-Dahlem, Dept. for Biodiversity Informatics

Fig. 12: Add included taxon form

4.5 Editing nomenclatural references

The form for editing nomenclatural references (Fig. 13) provides similar functions as the Edit Name form (check, apply and cancel). The checking function checks the strings in the text fields for Year, Volume and Pages.

The screenshot shows the 'Euro+Med Taxonomic Editor' interface. At the top right is the logo for 'EURO+MED PlantBase'. Below the header is a navigation bar with links: 'Search for taxon', 'Logout', 'Charsets', and 'Submit revision'. The main heading is 'Edit nomenclatural citation for *Pilularia townsendii* Pomel'. The form contains several input fields: 'Periodical or book' with a dropdown menu showing 'Sp. Umb.' and a 'New' link; 'Year' with a text box containing '1945-1948'; 'Volume' with a text box containing '5'; 'Pages' with a text box containing '234&656'; and 'Notes' with an empty text box. At the bottom of the form are three buttons: 'Check', 'Apply', and 'Cancel'. Below the form, the version information is displayed: 'Version 0.97 (01/2002) - A.Güntsch, J.Li & W.G.Berendsohn, Botanic Garden and Botanical Museum Berlin-Dahlem, Dept. for Biodiversity Informatics'.

Fig. 13: Edit Nomenclatural Citation Form

If a book or periodical title is missing in the list of already existing titles, it can be added by following the “New” link.

4.6 Altering the status of a taxon

At present, the editor prototype allows altering the status of a taxon between accepted or provisionally accepted through this form (Fig. 14). After pressing the apply button, the status will be changed and the user is directed back to the main form entry of the taxon.

The screenshot shows the 'Euro+Med Taxonomic Editor' interface. At the top right is the logo for 'EURO+MED PlantBase'. Below the header is a navigation bar with links: 'Search for taxon', 'Logout', 'Charsets', and 'Submit revision'. The main heading is 'Alter status for *Pilularia townsendii* Pomel'. The form contains a 'Select status:' dropdown menu with 'accepted' selected. Below the dropdown are two buttons: 'Apply' and 'Cancel'. A small window is open over the dropdown menu, showing the options 'provisionally accepted' and 'accepted'. Below the form, the version information is displayed: 'Version 0.97 (01/2002) - A.Güntsch, J.Li & W.G.Berendsohn, Botanic Garden and Botanical Museum Berlin-Dahlem, Dept. for Biodiversity Informatics'.

Fig. 14: Alter Status Form

The function for altering status values between accepted and synonym will be much more complex because the system should make sure that as much as possible information originally linked to the taxa involved can be preserved and have not to be retyped by the editor. Currently the following procedures are envisaged:

If the author of a revision wants to **change the status of a focussed name from accepted to synonym**, a dialog appears which asks for the (accepted) taxon of which the focussed name will be a synonym. Once this is selected, an additional form displays all taxonomic information attached to the previously accepted taxon (e.g. the list of synonyms). The user will now have to select which information will be transferred to the new accepted taxon. All links to included taxa will be severed and they will be marked for revision (they cannot be transferred automatically because they may have to change their name, if, for example, a new combination is accepted). The editor will be offered a choice to bulk-link factual data to the new accepted taxon to avoid re-entering information, but the system sets a flag indicating that all facts moved to the new accepted taxon have to be reedited.

Changing the status of the focussed name from synonym to accepted will open a dialog form to choose the new parent taxon. Neither factual data nor included taxa have to be considered, as they are not presently allowed for synonyms. The basionym (or replaced synonym) and the list of later homonyms will remain with the now accepted focussed name.

4.7 Adding and editing synonyms and misapplied names

Adding and editing synonyms and misapplied names (Fig. 15) for a given accepted

Euro+Med Taxonomic Editor



[Search for taxon](#)
[Logout](#)
[Charsets](#)
[Submit revision](#)

Edit synonym
Pisum sativum Clairv. for
Pilularia townsendii Pomel

Category:	Synonym: <input checked="" type="radio"/> Doubtful syn.: <input type="radio"/> Pro parte syn.: <input type="radio"/> Misapplied name: <input type="radio"/>		
Select rank:	Species <input type="text"/>		
Genus name:	<input type="text" value="Pisum"/>	<input type="checkbox"/> (gen. hybr.)	
Specific epithet:	<input type="text" value="sativum"/>	<input type="checkbox"/> (sp. hybr.)	
(Basionym authors:)	<input type="text" value="no selection"/>		
Authors:	<input type="text" value="Clairv."/>		
New authorteam	<input type="text"/>		
Notes	<input type="text"/>		

Check
Apply
Cancel

Version 0.97 (01/2002) - [A.Güntsch](#), [J.Li](#) & [W.G.Berendsohn](#)
 Botanic Garden and Botanical Museum Berlin-Dahlem, Dept. for Biodiversity Informatics

Fig. 15: Edit synonym and misapplied name form

name is similar to the adding and editing forms for accepted names. Form fields are displayed as required for the respective name structure. Additionally, the remote editor prototype form offers the option to choose between synonym, doubtful synonym, pro parte synonym, and misapplied name. A next version of the editor will also provide a partial synonym option and will allow marking any status as doubtful.

4.8 Adding and editing homonym and basionym

Adding and editing the basionym or replaced synonym and later homonyms for a given accepted name is similar to the adding and editing forms for accepted names. Form fields are displayed as required for the respective name structure. The editing homonym form is shown as Fig. 16.

The screenshot displays the 'Euro+Med Taxonomic Editor' interface. At the top right is the logo for 'EURO+MED PlantBase'. Below the header, there are navigation links: 'Search for taxon', 'Logout', 'Charsets', and 'Submit revision'. The main content area is titled 'Edit homonym' and shows the following details for *Pilularia townsendill*:
 - Original name: *Pilularia townsendill* Biv. for
 - New name: *Pilularia townsendill* Pomel
 - Select rank: Species (dropdown)
 - Genus name: Pilularia (text input) with a checkbox for '(gen. hybr.)'
 - Specific epithet: townsendill (text input) with a checkbox for '(sp. hybr.)'
 - (Basionym authors:): no selection (dropdown)
 - Authors: Biv. (dropdown)
 - A link for 'New authorteam' is present.
 - A 'Notes' section with a text area and scrollbars.
 - At the bottom, there are 'Check', 'Apply', and 'Cancel' buttons.

Version 0.97 (01/2002) - [A.Güntsch, J.Li & W.G.Berendsohn](#)
 Botanic Garden and Botanical Museum Berlin-Dahlem, Dept. for Biodiversity Informatics

Fig. 16 Edit homonym form

As with basionyms, the prototype system does not allow to link homonyms to synonyms. Since the forms for adding and editing homonyms already exist, this feature can easily be added if required by the project.

4.9 Deleting synonyms or accepted names

By choosing the delete option for a given synonym from the central form, a form is

The screenshot shows the 'Euro+Med Taxonomic Editor' interface. At the top right is the logo for 'EURO+MED PlantBase'. Below the header are navigation links: 'Search for taxon', 'Logout', 'Charsets', and 'Submit revision'. The main content area has a yellow background with the following text:

You are about to delete a synonym for
Pilularia townsendii Pomel

This action will **remove any link** to and from other taxa within the taxonomic group under revision. The name itself and its editing history will be **preserved but will not be visible** for you anymore.

Name: Pisum sativum Clairv.

Nomencl. ref.

Do you really want to delete this synonym from your revision?

Yes Cancel

At the bottom, there is a footer with version information: 'Version 0.97 (01/2002) - A.Güntsche, J.Li & W.G.Berendsohn Botanic Garden and Botanical Museum Berlin-Dahlem, Dept. for Biodiversity Informatics'.

Fig. 17: Feedback Form for Deleting a Synonym

opened summarizing the consequences of this action. A choice will be offered to simultaneously remove basionym/substituted synonym and later homonyms. Since names are “global objects” (see Fig. 17), the names themselves will not be deleted, but unlinked from the taxonomic tree under revision. Finally, by clicking the “Yes” button the action is performed.

The deletion of accepted names contained in a taxonomic sector under revision is much more critical because factual data are linked to accepted names, and other taxa might be included. This action will not delete any name strings from the system physically as names are global objects and must be available for different views on the data. An additional status (e.g. “deleted”) will be introduced to qualify potential taxa, which have been deleted by a reviser. A further form is used to display the deleted taxa and provide the options to either finally delete or relink the name to the tree under revision [not yet implemented in the prototype].

4.10 Factual data

According to the Euro+Med revision guidelines, the editor provides a form for free

The screenshot shows the 'Euro+Med Taxonomic Editor' interface. At the top right is the logo for 'EURO+MED PlantBase'. Below the header, there are navigation links: 'Search for taxon', 'Logout', 'Charsets', and 'Submit revision'. The main content area is titled 'Add description for *Pilularia townsendii* Pomel'. It features a large text input field for the description, followed by a 'Notes' section with another text input field. Below these are several form fields: 'Periodical or book' (a dropdown menu currently showing 'no selection' and a 'New' link), 'Citation details' (fields for 'Year', 'Vol.', and 'Pages'), 'Article title', 'Article authors', and 'Citation notes'. At the bottom of the form are 'Apply' and 'Cancel' buttons. A footer at the very bottom of the page reads: 'Version 0.97 (01/2002) - A.Giutsch, J.Li & W.G.Berendsohn, Botanic Garden and Botanical Museum Berlin-Dahlem, Dept. for Biodiversity Informatics'.

Fig. 18: Adding Factual Data

text entry and editing of factual data for the following factual data types: description, growth form, hardiness, ecology, phenology, karyology, illustration, identification key, observation, general distribution (Euro+Med), and general distribution (world).

Additionally, the form (Fig. 18) provides fields to add a citation for the factual data entered.

4.11 Editing distribution records

By following the link Distribution within the central navigation template, a summary table appears that is very similar to the paper form given in the revision guidelines, containing distribution information for all Euro+Med areas (Fig.19).



Euro+Med Taxonomic Editor

[Search for taxon](#) | [Logout](#) | Charsets | [Submit revision](#)

Distribution for
Pilularia townsendill Pomel

Area Code	Occur.	Nat.Stat.	Int.Stat.	Int.Agen.	Cult.Stat.	World Compl.	
EM							Edit
EUR							Edit
Ab							Edit
Ab(A)							Edit
Ab(N)							Edit
AE							Edit

Fig. 19: Distribution Information Form

Clicking the link *Edit* opens an editor for the respective area, which lets the author easily change occurrence values (Fig. 20).



Euro+Med Taxonomic Editor

[Search for taxon](#) | [Logout](#) | Charsets | [Submit revision](#)

Edit distribution status for
Pilularia townsendill Pomel and *EUR : Europe*

Occurrence	<input type="text"/>
Native Status	<input type="text"/>
Introduced Status	<input type="text"/>
Introduction Agency	<input type="text"/>
Cultivated Status	<input type="text"/>
World Dist. Compl.	C (Distribution complete)
Notes	<input type="text"/> <ul style="list-style-type: none"> C (Distribution complete) I (Distribution incomplete) U (Not known whether distribution complete)

|

Version 0.97 (01/2002) - [A.Gutsch](#), [J.Li](#) & [W.G.Berendsohn](#)
 Botanic Garden and Botanical Museum Berlin-Dahlem, Dept. for Biodiversity Informatics

Fig. 20: Edit Distribution Status

5 Complete list of tables and attributes

Tables are given alphabetically. SQL Server 2000 data types are used, which can easily be converted into any data type used by one of the other major database systems. NVARCHAR fields contain Unicode values up to the length given in the Precision column. NTEXT fields may contain Unicode strings of (almost) any length. Every primary key is represented with an integer number or a combination of integer numbers if it is constructed with foreign keys pointing to other tables. "Triggers" are small programs stored in the database and performing automatic functions as soon as a defined event (e.g. insert or update) occurs.

Please note that the latest version of this documentation is always available at <http://www.bgbm.org/scripts/ASP/BGBMModel/TablesAndAttr.asp?Cat=EM>

Area

Euro+Med project: The list of Euro+Med area codes including additional territories from Flora Europae and Med-Checklist considered to be outdated or unsuitable. The Area.Status field indicates this property.

Attribute	Type	Description
AreaId	int	Euro+Med project: primary key of Area table (geographical areas)
EMCode	char	Euro+Med project: geographic area code as given in "Revision Guidelines V1.2" - Geographical Standard
ISOCCode	char	Euro+Med project: ISO codes as given in "Revision Guidelines V1.2" - Geographical Standard
TDWGCode	char	Euro+Med project: TDWG geographic area codes as given in "Revision Guidelines V1.2" - Geographical Standard
Unit	nvarchar	Euro+Med project: area full name
Status	int	Euro+Med project: status for geographic area (1=Euro+Med area, 2=outdated)
OutputOrder	int	Euro+Med project: hierarchic level of geographic area for output

Author

Taxon author represented with first name(s), last names, and standard abbreviation. Additionally, Author.LifespanString can indicate when an author lived.

Attribute	Type	Description
AuthorId	int	Primary key for table Author
Abbrev	nvarchar	Abbreviation of author name as used in plant names and nomenclatural citations
FirstName	nvarchar	Author's full first name
LastName	nvarchar	Author's full last name
LifespanString	nvarchar	String indicating the author's lifespan (e.g. '1723-1780' or '1966-')
Created_When	datetime	Date and time when record was created

Updated_When	datetime	Date and time when record was last updated
Created_Who	nvarchar	Person who created the record
Updated_Who	nvarchar	Person who last updated the record
Notes	ntext	Remarks and notes

AuthorTeam

Representation of sets of authors with a unique identifier. The author team string or basionym author string itself will be calculated automatically from a trigger and be stored in the AuthorTeam.AuthorTeamCache field. However, this attribute can also be used to capture preliminary author strings where the individual authors have not yet been determined.

Attribute	Type	Description
AuthorTeamId	int	Primary key for table AuthorTeam
AuthorTeamCache	nvarchar	Complete author's team string to be constructed by a trigger (inserts/updates on tables Author and AuthorTeamSeq)
Created_When	datetime	Date and time when record was created
Updated_When	datetime	Date and time when record was last updated
Created_Who	nvarchar	Person who created the record
Updated_Who	nvarchar	Person who last updated the record
Notes	ntext	Remarks and notes

AuthorTeamSequence

A link between the Authors table and the AuthorTeam table representing a sorted list of authors within an author team. The AuthorTeam.Sequence field represents the position of an author within a team. A client program should make sure that this position is unique within a team.

Attribute	Type	Description
AuthorTeamFk	int	Pointer to AuthorTeam table and part of this table's primary key
AuthorFk	int	Pointer to table Author and part of this table's primary key
Sequence	int	Number indicating the position of this author within this author team. Make sure that sequence is different for different authors in an author team.

Fact

An entity holding any fact value represented as an unstructured text field. Fact.FactTypeFk indicates the type of fact (e.g. "description", "common name"), RefDetailFk points to a RefDetail table record to represent a literature reference for the individual fact.

Attribute	Type	Description
FactId	int	Primary key for table Fact
PTNameFk	int	Pointer to PTaxon table (name part)

PTRefDetailFk	int	Pointer to PTaxon table (reference detail part)
PTRefFk	int	Pointer to PTaxon table (reference part)
Fact	ntext	One fact's value as free text (e.g. "Daisy" for the fact category "Common name")
FactCategoryFk	int	Pointer to FactCategory table
FactRefDetailFk	int	Pointer to RefDetail table (giving the exact reference) for this fact
FactRefFk	int	Pointer to RefDetail table (giving the title or general description of the reference) for this fact
PTDesignationRefDetailFk	int	Pointer to RefDetail table (giving the exact reference) for the assignation of the fact to the potential taxon
PTDesignationRefFk	int	Pointer to RefDetail table (giving the title or general description of the reference) for the assignation of the fact to the potential taxon
DoubtfulFlag	bit	Indicating if the assignation of the fact to the potential taxon is doubtful or not
Created_When	datetime	Date and time when record was created
Updated_When	datetime	Date and time when record was last updated
Created_Who	nvarchar	Person who created the record
Updated_Who	nvarchar	Person who last updated the record
Notes	ntext	Remarks and notes

FactCategory

A list of factual data types.

Attribute	Type	Description
FactCategoryId	int	Primary key for table FactCategory
FactCategory	nvarchar	Fact category

Name

Representation of a botanical name (any rank). If a name is corrected, the “original version” is moved to the NameHistory table to preserve the editing process transparently. The Name.NameCache and Name.FullNameCache fields hold the latin name and the full name including author string as calculated from a trigger. Where no atomised values have been determined, it may also be used to capture preliminary name strings.

Attribute	Type	Description
NameId	int	Primary key for table Name
RankFk	int	Pointer to table Rank
NameCache	nvarchar	Complete latin name string to be constructed by a trigger (inserts/updates on name table)
UnnamedNamePhrase	nvarchar	Non-atomized addition to a name (e.g. for an

		'unnamed name': species A or for names including historical ranks).
FullNameCache	nvarchar	Complete latin name string including author strings to be constructed by a trigger (inserts/updates on name and author tables)
SupragenericName	nvarchar	Name of taxon with a rank above genus
Genus	nvarchar	Genus name - omit intergeneric hybrid marker for named hybrids, use 'x' for hybrid formula (e.g. 'Agrostis L. x Polypogon Desf.')
GenusSubdivisionEpi	nvarchar	Genus subdivision epithet
SpeciesEpi	nvarchar	Species epithet - omit interspecific hybrid marker for named hybrids, use 'x' for unnamed hybrids (e.g. 'carinthiacus x gouanii')
InfraSpeciesEpi	nvarchar	Infraspecific epithet
AuthorTeamFk	int	Pointer to authors in the AuthorTeam table
BasAuthorTeamFk	int	Pointer to basionym authors in the AuthorTeam table
MonomHybFlag	bit	Indication of monomial hybrid (e.g. generic hybrid)
BinomHybFlag	bit	Indication of binomial hybrid
TrinomHybFlag	bit	Indication of trinomial hybrid
CultivarGroupName	nvarchar	Cultivar group designation
CultivarName	nvarchar	Cultivar epithet
NomRefDetailFk	int	Pointer to RefDetail table (giving the exact reference) for this name's nomenclatural reference citation
NomRefFk	int	Pointer to RefDetail table (giving the title or general description of the reference) for this name's nomenclatural reference citation
Created_When	datetime	Date and time when record was created
Created_Who	nvarchar	Person who created the record
Notes	ntext	Remarks and notes

NameHistory

Older variants of a name which were replaced in the editing process. The Name.SuccNameHistId field contains a pointer to the next name within this revision history. Consequently, an empty Name.SuccNameHistId field indicates that the name is the last one within this sequence. The Name.CurrentNameFk field contains a pointer to the name table indicating the name string currently in use by the system.

Attribute	Type	Description
NameHistId	int	Primary key for table NameHistory
SuccNameHistId	int	Recursive pointer to this table indicating a newer (corrected) "version" of the name. SuccNameId = NULL --> this is the latest record of the name within

		the history table.
CurrentNameFk	int	Pointer to the name table indicating the "version" of the name currently used
RankFk	int	Pointer to table Rank
Name	nvarchar	Complete latin name string. Note that the contents of this field must not be updated by a trigger; the original content is to be preserved
FullName	nvarchar	Complete latin name string including author strings. Note that the contents of this field must not be updated by a trigger; the original content is to be preserved
UnnamedNamePhrase	nvarchar	Non-atomized addition to a name (e.g. for an 'unnamed name': species A or for names including historical ranks).
SupragenericName	nvarchar	Name of taxon with a rank above genus
Genus	nvarchar	Genus name - omit intergeneric hybrid marker for named hybrids, use 'x' for hybrid formula (e.g. 'Agrostis L. x Polypogon Desf.')
GenusSubdivisionEpi	nvarchar	Genus subdivision epithet
SpeciesEpi	nvarchar	Species epithet - omit interspecific hybrid marker for named hybrids, use 'x' for unnamed hybrids (e.g. 'carinthiacus x gouanii')
InfraSpeciesEpi	nvarchar	Infraspecific epithet
AuthorTeam	nvarchar	Author team as a string
BasAuthorTeam	nvarchar	Basionym author team as a string
MonomHybFlag	bit	Indication of monomial hybrid (e.g. generic hybrid)
BinomHybFlag	bit	Indication of binomial hybrid
TrinomHybFlag	bit	Indication of trinomial hybrid
CultivarGroupName	nvarchar	Cultivar group designation
CultivarName	nvarchar	Cultivar epithet
NomenclCitation	nvarchar	Complete nomenclatural reference citation as a string
Created_When	datetime	Date and time when record was created
Created_Who	nvarchar	Person who created the record
Notes	ntext	Remarks and notes

NomStatus

List of nomenclatural status value categories.

Attribute	Type	Description
NomStatusId	int	Primary key for table NomStatus
NomStatus	nvarchar	Nomenclatural status (e.g. "nom. cons.")

NomStatusRel

Representation of links between table Name and table NomStatus to assign one or several status values to a name

Attribute	Type	Description
NameFk	int	Pointer to Name table and part of this table's primary key
NomStatusFk	int	Pointer to NomStatus table and part of this table's primary key
NomStatusRefDetailFk	int	Pointer to RefDetail table (giving the exact reference) for the assignation of the nomenclatural status to the name
NomStatusRefFk	int	Pointer to RefDetail table (giving the title or general description of the reference) for the assignation of the nomenclatural status to the name
Created_When	datetime	Date and time when record was created
Updated_When	datetime	Date and time when record was last updated
Created_Who	nvarchar	Person who created the record
Updated_Who	nvarchar	Person who created the record
Notes	ntext	Remarks and notes

Occurrence

Euro+Med project: Indication of the occurrence of a taxon as specified in the Euro+Med Revision Guidelines, links an entry within the PTaxon table to the Area table. The OccurrenceRefDetailFk field can hold a pointer to an entry in the RefDetail table.

Attribute	Type	Description
PTNameFk	int	Euro+Med project: pointer to PTaxon table and part of this table's primary key
PTRefDetailFk	int	Euro+Med project: pointer to PTaxon table and part of this table's primary key
PTRefFk	int	Euro+Med project: pointer to PTaxon table and part of this table's primary key
AreaFk	int	Euro+Med project: pointer to Area table and part of this table's primary key
Occurrence	char	Euro+Med project: occurrence as specified in "Revision Guidelines V1.2"
NativeStatus	char	Euro+Med project: native status as specified in "Revision Guidelines V1.2"
IntroducedStatus	char	Euro+Med project: introduced status as specified in "Revision Guidelines V1.2"
IntroductionAgency	char	Euro+Med project: introduction agency as specified in "Revision Guidelines V1.2"
CultivatedStatus	char	Euro+Med project: cultivated status as specified

		in "Revision Guidelines V1.2"
WorldDistCompl	char	Euro+Med project: world distribution completeness as specified in "Revision Guidelines V1.2"
OccurrenceRefDetailFk	int	Euro+Med project: pointer to RefDetail table (giving the exact reference) for the occurrence record
OccurrenceRefFk	int	Euro+Med project: pointer to RefDetail table (giving the title or general description of the reference) for the occurrence record
PTDesignationRefDetailFk	int	Euro+Med project: pointer to RefDetail table (giving the exact reference) for the assignation of the occurrence to the potential taxon
PTDesignationRefFk	int	Euro+Med project: pointer to RefDetail table (giving the title or general description of the reference) for the assignation of the occurrence to the potential taxon
Created_When	datetime	Euro+Med project: date and time when record was created
Updated_When	datetime	Euro+Med project: date and time when record was last updated
Created_Who	nvarchar	Euro+Med project: person who created the record
Updated_Who	nvarchar	Euro+Med project: person who last updated the record
Notes	ntext	Euro+Med project: remarks and notes

PTaxon

The central entity in the model represents pairs of Names (PTNameFk) and References (PTRefDetailFk, PTRefFk). The status of a name and all kind of factual data are linked to this entity instead to the name itself to preserve data as they are given in literature, databases, or by persons.

Attribute	Type	Description
PTNameFk	int	Pointer to Name table and part of this table's primary key
PTRefDetailFk	int	Pointer to Reference table and part of this table's primary key
PTRefFk	int	Pointer to PTaxon table and part of this table's primary key
IdInSource	nvarchar	ID in the original source if this potential taxon (name) was imported from a dataset
DoubtfulFlag	bit	Flag indicating whether the status of this potential taxon is considered doubtful. For accepted names, this property is interpreted as 'provisionally'.
StatusFk	int	Pointer to Status table
NamePhrase	nvarchar	Additional name suffix to name this potential taxon (e.g.

		"sensu auct. amer.")
Created_When	datetime	Date and time when record was created
Updated_When	datetime	Date and time when record was last updated
Created_Who	nvarchar	Person who created the record
Updated_Who	nvarchar	Person who last updated the record
Notes	ntext	Remarks and notes

Rank

Rank linked to every name within the Name and NameHistory tables.

Attribute	Type	Description
RankId	int	Primary key for table Rank
RankAbbrev	nvarchar	Abbreviation of rank (e.g. 'subsp.')
Rank	nvarchar	Full name of rank (e.g. 'species')
HigherRankFk	int	Pointer to next higher rank within this table. May be used for sorting purposes
EMBasicFlag	bit	Euro+Med project: rank is family, genus, species, or subspecies (see "Revision Guidelines V1.2" - taxonomic concepts)
EMExtFlag	bit	Euro+Med project: rank is family, subfamily, tribe, subtribe, genus, subgenus, series, section, species, subspecies, or species aggregate (see "Revision Guidelines V1.2" - taxonomic concepts)
EMSynFlag	bit	Euro+Med project: flag indicating that this rank is used for synonyms

RefCategory

List of reference categories.

Attribute	Type	Description
RefCategoryId	Int	Primary key for table RefCategory
RefCategory	nvarchar	Category of a reference (e.g. 'database' or 'book')
RefCategoryAbbrev	nvarchar	Abbreviation of reference category

RefDetail

Representation of citations from a specific Reference (e.g. a book) represented by a link to the Reference table and qualified by means of values in the Detail field (a page number or table number for example).

Attribute	Type	Description
RefDetailId	int	Primary key (first part) for table RefDetail
RefFk	int	Pointer to Reference table indicating the title or general description of the reference this detail belongs to and part of this table's primary key
FullRefCache	nvarchar	Complete reference string to be constructed by a trigger

		(inserts/updates on Reference)
Details	nvarchar	String of reference details such as page, page range, no. of figure, etc.
SecondarySources	nvarchar	Secondary sources (sources named in the reference)
Created_When	datetime	Date and time when record was created
Updated_When	datetime	Date and time when record was created
Created_Who	nvarchar	Person who created the record
Updated_Who	nvarchar	Person who created the record
Notes	ntext	Remarks and notes

Reference

Hierarchical representation of references of different types such as books, journals, articles, CDs etc. The self referential foreign key RefCategoryFk is used to indicated that a reference is part of another reference. Reference types are given as a foreign key (RefCategoryFk) to the RefCategory table.

Attribute	Type	Description
RefId	int	Primary key for table Reference
RefCategoryFk	int	Pointer to RefCategory table indicating the nature of a reference (e.g. database)
InRefFk	int	Recursive pointer to this table indicating the reference this reference is part of (e.g. contribution in book)
Title	nvarchar	Full title of the reference
NomTitleAbbrev	nvarchar	Abbreviation of full title as used in nomenclatural citations (e.g. 'Sp. Pl. 2')
NomAuthorTeamFk	int	Pointer to the authorteam table
RefAuthorString	nvarchar	Author string
DateString	nvarchar	A date enabeling the system to distinguish different "versions" of the same source (important if the source is a person)
Edition	nvarchar	Edition string (e.g. 'ed.2')
Volume	nvarchar	Volume as a string (e.g. '33' or 'ser.3, 2')
Series	nvarchar	Publication series the reference belongs to
RefYear	nvarchar	Year of publication as a string
PageString	nvarchar	Pages of an article or other part in a book or journal
ISSN	nvarchar	ISSN code of the publication
ISBN	nvarchar	ISBN code of the publication
URL	nvarchar	Full URL ('http://...') if the data source is already accessible on the www
ExportDate	nvarchar	Date when the reference was exported (e.g. from a database)
PublicationTown	nvarchar	Place of publication

Publisher	nvarchar	Publisher string
ThesisFlag	bit	Indicating if the reference is a thesis
RefDepositedAt	nvarchar	Place where the reference is located (e.g. the library where a manuscript was deposited)
InformalRefCategory	nvarchar	Informal reference category
IsPaper	bit	Indication that the reference is printed publication or not
RefSourceFk	int	Pointer to RefSource table
IdInSource	nvarchar	ID in the original source if this reference was imported from a dataset
Created_When	datetime	Date and time when record was created
Updated_When	datetime	Date and time when record was last updated
Created_Who	nvarchar	Person who created the record
Updated_Who	nvarchar	Person who last updated the record
Notes	ntext	Remarks and notes

RefSource

Information source providing a reference list (e.g. a list of journal titles).

Attribute	Type	Description
RefSourceId	int	Primary key for table RefSource
RefSource	nvarchar	Source of a reference record
Created_Who	nvarchar	Person who created the record
Updated_Who	nvarchar	Person who last updated the record
Created_When	datetime	Date and time when record was created
Updated_When	datetime	Date and time when record was last updated
Notes	ntext	Remarks and notes

RelName

Representation of arbitrary binary relations between names. Can be used to e.g. to implement basionym and later homonym relations.

Attribute	Type	Description
RelNameId	int	Primary key for table RelName
NameFk1	int	Pointer to 1st Name
NameFk2	int	Pointer to 2nd name
RelNameQualifierFk	int	Pointer to RelNameQualifier table
RefDetailFk	int	Pointer to RefDetail table (giving the title or general description of the reference) for the source of this relationship
RefFk	int	Pointer to RefDetail table (giving the title or general description of the reference) for the source of this relationship

Created_When	datetime	Date and time when record was created
Updated_When	datetime	Date and time when record was last updated
Created_Who	nvarchar	Person who created the record
Updated_Who	nvarchar	Person who last updated the record
Notes	ntext	Remarks and notes

RelNameQualifier

List of relation categories qualifying relations between names within the RelName table.

Attribute	Type	Description
RelNameQualifierId	int	Primary key for table RelNameQualifier
RelNameQualifier	nvarchar	Qualification of relations between two names stored in Name table (e.g. "is basionym of" or "is later homonym of")

RelPTaxon

Representation of arbitrary binary relations between potential taxa as stored in the PTaxon table. This includes taxonomic hierarchies, synonymies of any kind, and concept relationships. The type of a given relation is indicated with the field RelQualifierFk pointing to the RelPTQualifier table.

Attribute	Type	Description
RelPTaxonId	int	Primary key for table RelPTaxon
PTNameFk1	int	Pointer to 1st potential taxon (name part)
PTRefDetailFk1	int	Pointer to 1st potential taxon (reference detail part)
PTRefFk1	int	Pointer to 1st potential taxon (reference part)
PTNameFk2	int	Pointer to 2nd potential taxon (name part)
PTRefDetailFk2	int	Pointer to 2nd potential taxon (reference detail part)
PTRefFk2	int	Pointer to 2nd potential taxon (reference part)
RelQualifierFk	int	Pointer to RelQualifier table indicating the type of relation between the potential taxa involved
RelRefDetailFk	int	Pointer to RefDetail table (giving the exact reference) for the assignation of this relationship between potential taxa
RelRefFk	int	Pointer to RefDetail table (giving the title or general description of the reference) for the assignation of this relationship between potential taxa
Created_When	datetime	Date and time when record was created
Updated_When	datetime	Date and time when record was last updated
Created_Who	nvarchar	Person who created the record
Updated_Who	nvarchar	Person who last updated the record
Notes	ntext	Remarks and notes

RelPTQualifier

List of relation categories as used in the RelPTaxon table.

Attribute	Type	Description
RelPTQualifierId	int	Primary key for table RelPTQualifier
RelPTQualifier	nvarchar	Qualification of relation between two potential taxa (e.g. "is synonym of")

6 Content of tables representing categories**FactCategory**

description
 growth form
 hardiness
 ecology
 phenology
 karyology
 illustration
 identification key to included taxa
 observation
 general distribution (Euro+Med)
 general distribution (world)

RefCategory

article in periodical
 part of other title
 book
 database
 informal reference
 not applicable
 website
 published compact disc
 journal

RelNameQualifier

is basionym for
 is homonym of
 is replaced synonym for
 validation of
 later validation of
 is type of
 is conserved type of
 is rejected type of

RelPTQualifier

is included in
is synonym of
is misapplied name for
is pro parte synonym of
is partial synonym of
is congruent to
is contained in
includes
overlaps
excludes

Status

accepted
synonym
partial synonym
pro parte synonym

7 Entity-Relationship diagrams

An Entity-Relationship diagram of the core information model is available at http://www.bgbm.org/biodivinf/docs/bgbm-model/Core_Diagram.pdf. Euro+Med specific entities (e.g. occurrence) are depicted with the diagram given at <http://www.bgbm.org/biodivinf/docs/bgbm-model/occurrence.pdf>

8 References

- Berendsohn, W.G. 1999: Names, Taxa, and Information. In Blum, S. (ed): Proceedings of the Taxonomic Authority Files Workshop, Washington, DC, June 22-23, 1998. [<http://research.calacademy.org/taf/proceedings/Berendsohn.html>]
- Geoffroy, M. & Güntsch, A. 2001: Handling factual information linked to parallel taxonomic concepts in biology. 17th annual meeting of the Taxonomic Databases Working Group (TDWG 2001), Sydney November 2001, Abstract Volume [http://plantnet.rgbsyd.gov.au/bioforum/TDWG_program/tdwg_abstracts.html]