

More on the Eclipse Rich Client Platform

The Eclipse Rich Client Platform is a Java library of interface components. It was first used to develop the Eclipse Platform for coding Java, basically a “word processor for programmers”. In essence, we will take elements from the Eclipse development environment and adapt them to the Editor.

Eclipse offers Windows-like application features such as showing elements in a tree view, managing multiple views in a single window (i.e. for viewing several taxa at once), and storing session data (i.e. the state of the application picks up where the user left off the day before, even when the data has not been explicitly saved to the CDM), all of which save significant programming time.

Programming Java and taxonomy both use strict rules and grammars, and both have a fairly well-defined workflow. As such, there are several Eclipse features that can be re-used in a taxonomist context, including grammar checking, error-reporting, and suggestions mechanisms.

There is also a component in the Eclipse framework with which a view of all errors contained in a Java project (equivalent to an entire revisionary treatment) can be created. This is not planned for the first version of the Editor, but should be implemented at a later date.

Simplified workflows

The standard approach of the Web Editor was to enforce data integrity at the interface level. This resulted in key workflows which required a tedious number of steps, and which could in the majority of cases be accomplished with one or two steps (the most notorious of these - changing an accepted taxon to a synonym – is discussed below). We have decided that as a matter of principle, design should focus on identifying and simplifying the default case, and not on designing complex workflows for cases that occur infrequently. In practice, this means automating as much as possible, and enforcing data integrity as little as possible – in short, trusting the taxonomist to do what is right.

Warnings not confirmations

Several operations in the Web Editor resulted in an unwieldy number of clicks and confirmation dialogs. As described by RASKIN (2000), “any confirmation step that elicits a fixed response soon becomes useless”. In other words, a user who is continually asked if he is sure he wants to do something will click “yes” whether he is sure or not; he will quickly cease to ask himself the question at all.

Instead, we will replace enforced integrity and confirmation messages with a less intrusive model of warning messages. We will show the user the implications of what he has done, and then allow him to manually undo the step he has just taken. Here an “undo” button would be desirable, but this is exceedingly difficult to implement, and we have decided not to include it with the first version of the Editor.

Warnings will often come with suggestions - we often have a fairly good idea of what the taxonomist is trying to do, but at the same time the program should give the taxonomist the benefit of the doubt, and not force him to do anything against his will.

A typical use of a warning: a taxon’s data – its author, for example – has been changed in a way that can be interpreted differently. For a given situation, the system will always act based on a default interpretation, but alert the user to other possibilities. (This is discussed further in “**Creating and editing**” below.) Another example: when editing shared datasets, warnings will be shown when another user has modified shared data.

Core gestures

Editing-in-place / freetext – Starting from the notion that taxonomists are most accustomed to editing their revisions in Microsoft Word, we will attempt to adapt to the Editor much of what is familiar to users of traditional word processing applications. We will move away from over-reliance on forms, a decidedly dreary way for users to input data. The core solution is to split all editing windows into two panels, one which contains an area for entering freetext and one for displaying parsed freetext data in structured form fields; data can of course also be entered manually in the form panel.

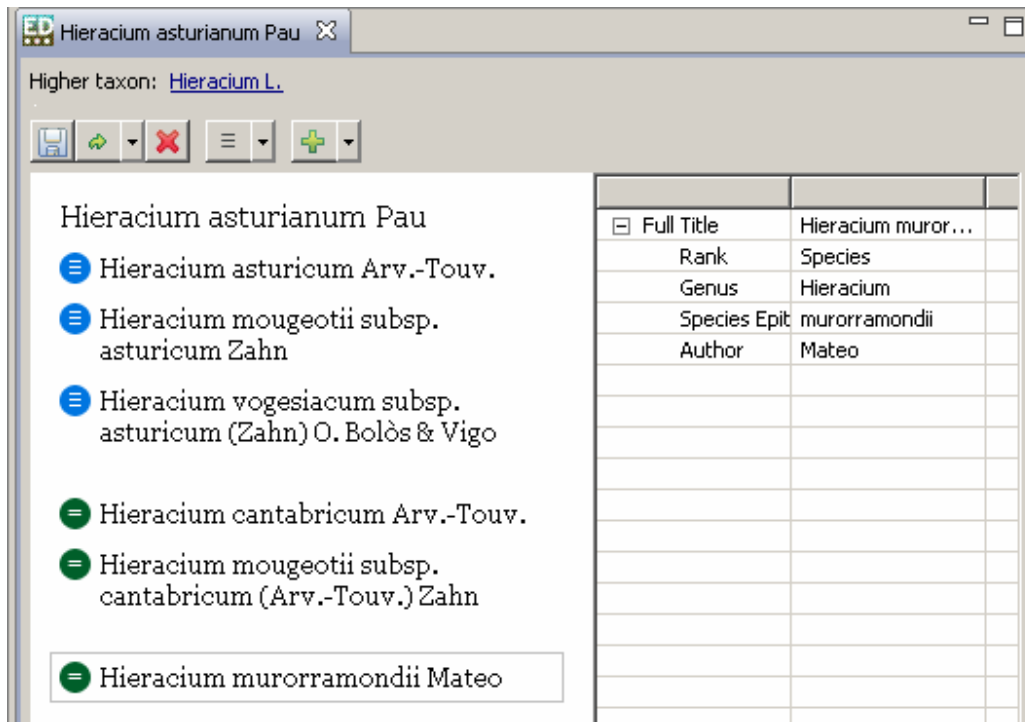


Figure 2: *Hieracium murorramondii* Mateo can be edited in either the freetext or form area.

The parsing algorithms developed over the years by programmers in the EDIT community will catch well over 95% of all name, author, and reference patterns; furthermore, prototypes have proven that this can take place fast enough to allow parsing to take place on-the-fly, i.e. input will be parsed with every keystroke. Data that cannot be parsed will be marked as such with the red squiggly underline familiar from spelling errors in Microsoft Word; the cause for the error will be shown by hovering over the error, which can then be corrected either in the freetext area or in the form view, or saved without the user being forced to make the data error-free; some errors – such as orthographic variants – are desired.

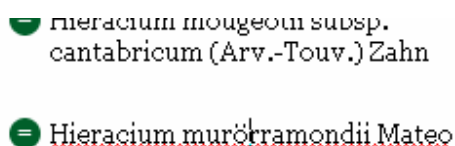


Figure 3: The name cannot be parsed correctly due to non-Latin characters in the specific epithet.

Drag-and-drop – while there will be menu and toolbar commands for all Editor functions, many of these can be most efficiently accomplished using drag-and-drop. These include: dragging a taxon from one parent to another in the taxonomic tree; dragging descriptive data from one taxon to another; dragging a synonym from one homotypic group to another; and dragging a name from the left panel – search results, recently viewed, and favorites - into a synonymy list.

Context menus – In the prototypes demonstrated to the EDIT community to date, users have been enthusiastic about the freetext and drag-and-drop possibilities of the new editor, and welcomed them as intuitive solutions to traditional workflow problems (see “**Changing a name’s status from taxon to synonym**” below). Context menus, however, may require a bit more familiarization. The Eclipse approach relies heavily on the idea that a right mouse click on most items in the interface will summon a drop-down menu of possible operations on the item. This has the significant advantage that the user’s focus remains in the same spot on the interface, without forcing his attention to wander up to the File menu or toolbar area to choose an operation (RASKIN, 2000). In addition, interface elements that are draggable should have a clear

separation between dragging with the left mouse button and viewing a context menu with the right button. However, this reliance on right mouse clicking – while fairly standard in Windows applications - may not be a familiar gesture to the average software user, and a challenge for training to use the Editor will be to habituate the user to expect context menus to be triggered via right-mouse click.

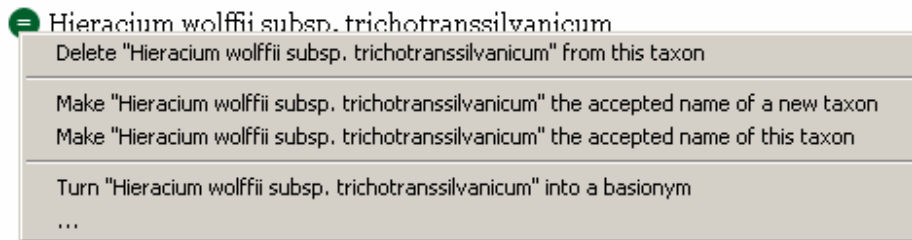


Figure 4: Context menu called with a right-mouse click on the relationship icon.

Autosuggest – While not strictly a gesture, insofar as it is not a user movement but a software response, autosuggest should be mentioned here, as it will be used extensively in the freetext area. Please see “**Creating and editing**” below for a detailed description of autosuggest workflow.

Description of functionality

Application layout

The Editor is taxon-centric; its most basic unit of work – analagous to a file in Microsoft Word – is the taxon. The user accesses open taxa via window tabs at the top of each taxon view, while individual taxon components – Name, Concept, Descriptive, Geography – are accessible via page tabs at the bottom of each taxon view.

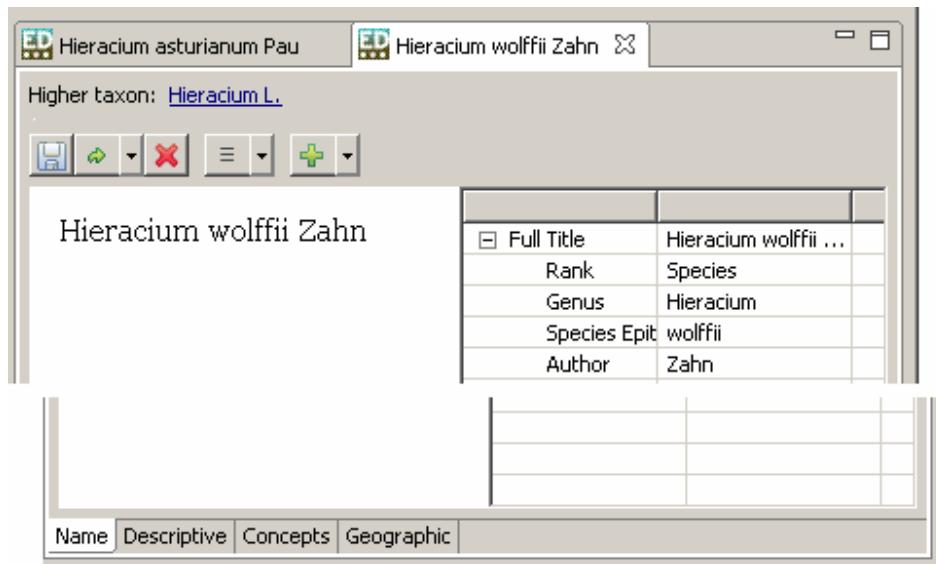



Figure 5: Top tabs are for taxa views, bottom tabs for taxon components.

The taxonomic tree in the left-hand panel (*see Figure 1*) is analagous to the file system. There are other ways to access data in the left panel: search, recent names, and favorite names (“mark as favorite” will be an option in a name’s context menu). In addition, there may be a container or trash can for “free-floating” information, such as names or descriptive data that have been marked for deletion. This concept must be further developed. Any data that appears in the left-hand panel can be dragged into the editing area where appropriate.

Another idea in need of further development is how non-name / -taxon CDM elements should make use of the left panel. For instance, there could be users who find it more comfortable to find a reference using a traditional search box, then dragging it over to the editing area to associate it with a name, as opposed to using the autocomplete functionality within the editing area.

Taxonomic tree

Taxa affected by a given operation will be revealed within the taxonomic tree, meaning that if these taxa are hidden at the start of the operation, the tree will open to show them. The taxon that currently has focus in the editing view will be highlighted in the tree, and its child taxa will also be revealed.

Dragging and dropping a taxon within the taxonomic tree will be allowed, either from one parent taxon to another or to the root level to disassociate it from all parents. If any taxonomic rules are broken in this process, the taxon will be marked either with a warning sign  or with the red underlining familiar from applications such as Microsoft Word spellchecking. We may decide to use only one of these – underlining or icons – for simplicity's sake, but the important thing is that hovering over either will provide information on the error detected.

A taxon can be deleted directly from the taxonomic tree, either by selecting it and hitting the “delete” key, or by choosing “Delete” from the tree node's context menu. (Note that all attached name, author, and reference data will remain in the CDM after the taxon is dissolved, resurfacing in searches and autocomplete).

For a quick overview of synonymy, the user need not open the taxon; its synonymy list will be displayed when the cursor hovers over its name in the taxonomic tree.

Creating a new taxon

Creating a new taxon will be possible from several locations in the interface: from the File menu; from the taxonomic tree, by right-clicking and choosing either “Add child taxon” or “Add child taxon with quick name”; or, from the taxon view, the two menu options “Add new taxon to this taxon” or “Add new taxon to parent taxon”.

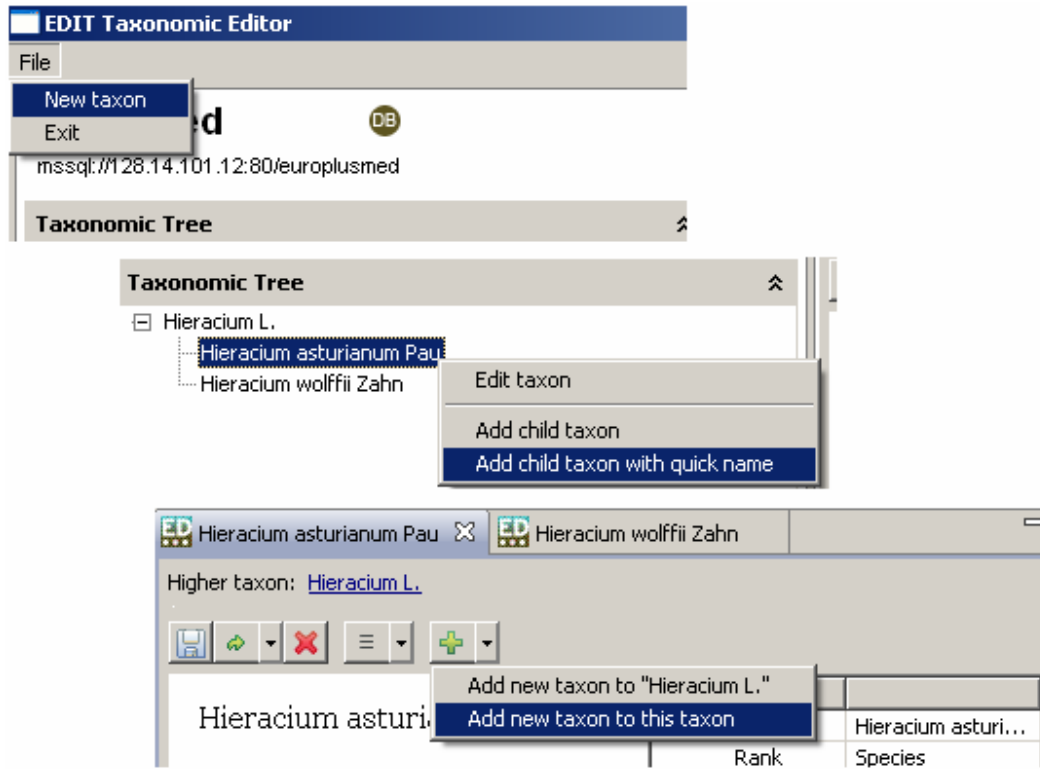


Figure 6: There are several ways to add a new taxon.

Except in the “quick name” case, a new empty taxon view will then be opened. This taxon cannot be saved until a name has been entered in some form; the name need not be parsable or syntactically correct to be saved. Note that a new taxon will only be created in the CDM when the user has saved it, either by pressing the save button or with control-s – until saved, new data exists only transiently in the Editor environment.

If a new taxon without a parent has been chosen, a button in the toolbar will open a link to the “choose parent taxon” wizard, which is the tree, recently viewed, and search components of the left Editor panel re-arranged. Upon saving the new taxon, it will appear for the first time in the taxonomic tree. Taxa that are not attached to a parent taxon will appear at the bottom of the taxonomic tree.

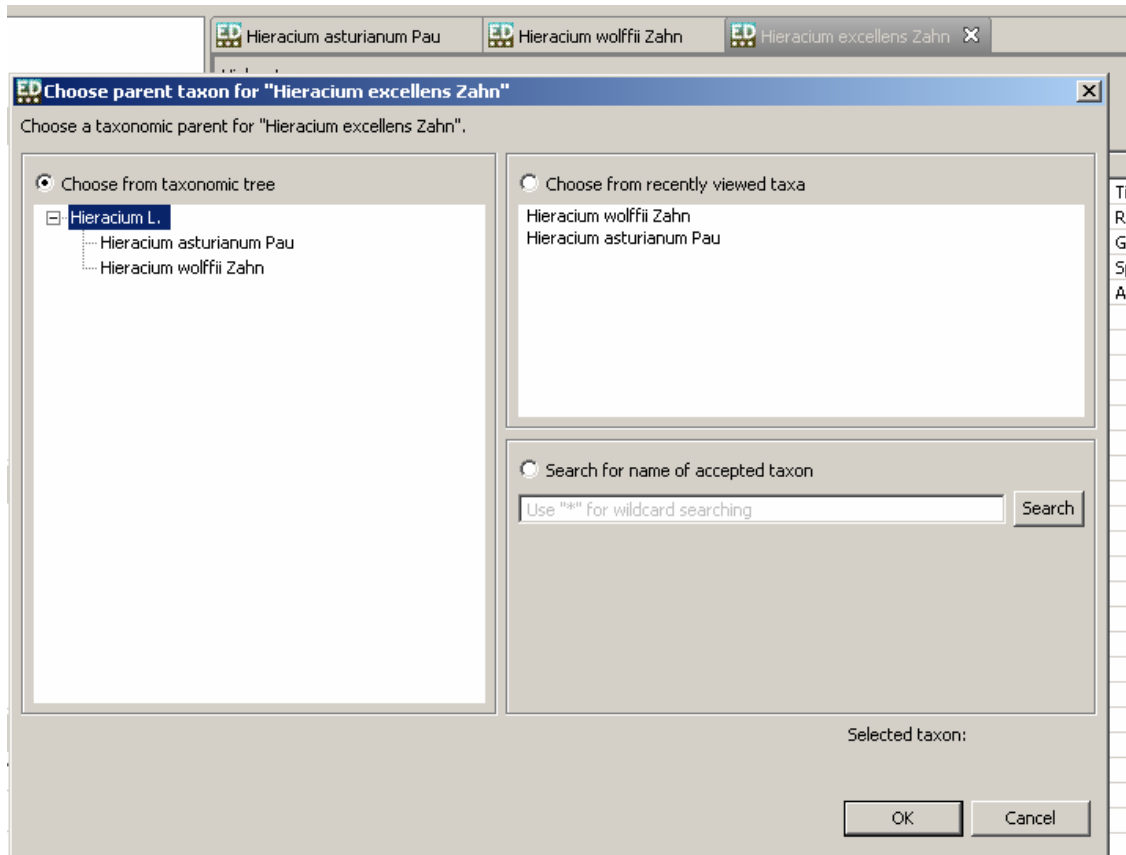


Figure 7: Choosing a taxonomic parent.

The “quick name” option results in the creation of an empty taxonomic child node in the taxonomic tree, in which the user then enters the taxon’s name. The empty node will disappear from the tree if the user changes focus before entering a name. The created taxon will consist of only a name and the relation to its parent taxon. Note that with “quick name”, the user does not have to explicitly save; simply by entering the name, the new child taxon is created in the CDM.

Creating and editing

In the central freetext area, the border between creating and editing a name is fluid. As the user enters data, autosuggest menus with appropriate data from the CDM will be displayed corresponding to cursor position, i.e. first for the genus; then for the full name with author team; if a name with author team was not chosen, then for the author; etc. For instance, if the user’s cursor is at a position where an author team is expected, and he has typed “Li”, an autocomplete menu will appear suggesting “Lid”, “Lidén”, “Lidén & A.Soler”, “Liebl.”, etc. If the user types his entry to the end without choosing anything from the list, a new author team record will be created in the CDM when the taxon is saved. Otherwise, the chosen item from the autocomplete becomes associated with the name.

Experience has shown that reference and author team lists often contain data that appear nearly identical at first glance, and that this can lead to the taxonomist choosing false data for association. Therefore, when viewing an entry in an autocomplete menu, the user can hover over an entry to see additional clarifying information, for example, the publication date, volume information, etc. of a reference.

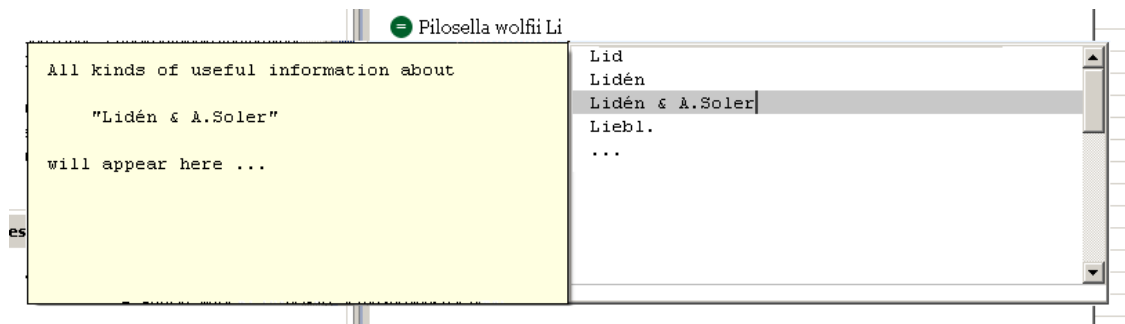


Figure 8: Autocomplete with additional information.

Entries will be parsed on-the-fly. This has two consequences: first, the form view of the data will be populated as soon as the parser can make sense of the entry; second, data that could not be parsed will be marked as such with red underlining. A mouse over the underline will show an explanation of why the data could not be parsed, and suggestions for making it valid (this functionality may not be in the first version of the Editor).

For existing data, the Editor proceeds with the assumption that any changes represent new data. Note again that these changes are only committed to the CDM when the user saves the taxon – not every keystroke creates a new record. If the user edits the author field – either in the freetext area or in the form view - the Editor will create a new author team the next time the taxon is saved. If the previous author team has no other associated records, it will be deleted from the CDM.

If, however, the author team is used by other names, the new author team will still be created, but here we make use of the suggestion functionality described earlier. A warning is shown; upon clicking on the warning, the user will be given the option of using the new author team for all records that are associated with the old one. This functionality will also be available by selecting the author team and choosing “change all associated records” from the context menu.

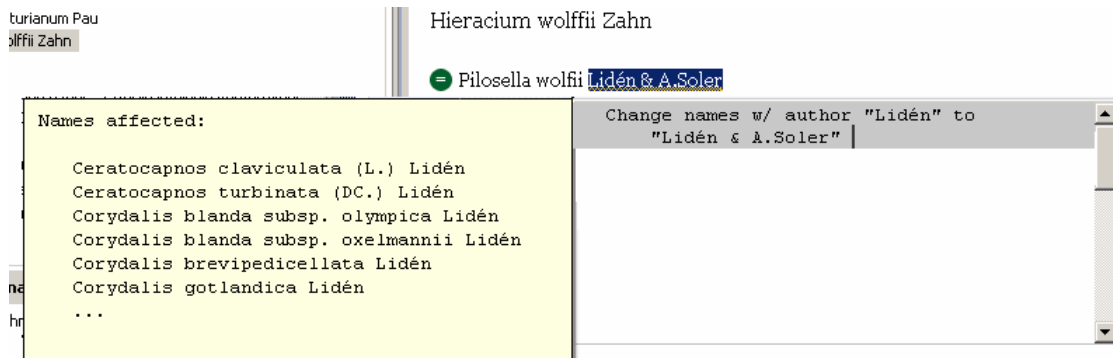


Figure 9: Change all records associated with an author team.

This paradigm – all changes create new records that don’t affect related data, unless explicitly told to do otherwise – will be used for name, author, and reference data.

The form view

While each editing window corresponds to a taxon, each form view (the right half of the editing window – see *Figure 1*) corresponds to a name. Users will enter all name, author, and nomenclatural data in the freetext area, where it will be parsed on the fly into the form view on the right. For example, nomenclatural references will be broken up into title, volume, pages, etc. Each object group can be clapped shut to reduce visual clutter. Some data not visible in the freetext area, such as type and specimen information, or the kind of orthographic variant (“as published”, “corrected in accordance with ICBN ...”), will also be edited in the form view.

[-] Full Title	Hieracium wolffii ...
Rank	Species
Genus	Hieracium
Species Epit	wolffii
Author	Zahn
[+] Reference	

[+] Full Title	Hieracium wolffii ...
[+] Reference	

Figure 10: Clapping name information shut.

Creating synonymy

The Editor must show not only a taxon's synonyms, but also the type relations among the names of the accepted taxon and its synonyms. When two names share the same type information, they are said to be members of a homotypic group. Synonyms in the same homotypic group as the accepted taxon are called homotypic synonyms; synonyms outside the accepted name's homotypic group are referred to as heterotypic. The taxon editing window separates these homotypic groups with extra white space (*see Figure 2*).

The simplest way to create a synonym relationship is to hit carriage return at the end of a line in the freetext area. This creates: a new name input field; a new heterotypic synonym relationship; and a new homotypic group. The name can then be entered and edited as described above. To put the synonym into an existing homotypic group, the name is dragged over the target group. Resulting empty homotypic groups will be deleted. Note that a heterotypic synonym dragged into the same homotypic group as the taxon automatically becomes a homotypic synonym.

If the synonym also contains a name relation to the accepted taxon – for instance, if a homotypic synonym is also a basionym - this can be chosen from a context menu accessed by pressing the synonymy type icon (*see Figure 4*).

The option of deleting a synonym will also be included in the context menu. Alternately, clicking on the name or its icon will cause a box to be drawn around the entry, at which point the user can click the “delete” key. The user will also be able to choose several synonyms at once by holding the “control” button while clicking multiple entries; this could also be used to move several entries into a single homotypic group. Note that this behavior will also be used in other taxon editing windows, i.e. descriptive data and concept relations.

Here a quick word about the difference between synonymy relations and name relations within the context of the CDM is necessary. We are attempting to assign names to real biological entities in nature; this name information, combined with additional author, nomenclatural, and descriptive feature information, is called a taxonomic concept. Within a taxonomic treatment, one name for a concept is designated accepted and constitutes the taxon; different names used for the concept in other literary sources or historically are, when specifically cited within the treatment, called synonyms. However, there are also relations between names that have not yet been explicitly cited. For instance, if NAME1 is a basionym of NAME2, but NAME1 has not yet been explicitly designated a status in the current CDM treatment, it does not have a synonym relation to NAME2. This is frequently the case when an existing treatment is imported into a CDM dataset to use its name data as a basis for a new treatment. (For a discussion of the use of concepts in taxonomy, please see BERENDSOHN (1997) http://www.bgbm.fu-berlin.de/BioDivInf/Docs/IOPI_Model/IOPI_Model.pdf)

If the taxon has name relations that don't have corresponding synonymy relations, a warning with non-synonymized names will be shown next to the accepted name, along with suggestions for appropriate steps to take, i.e. “create homotypic synonym for basionym Xxxx yyyy”.

Changing from synonym to taxon

There are two actions available from the synonym's context menu here: changing a synonym into a new taxon; and swapping a synonym with the existing taxon. Both options can be chosen from the synonym's

context menu called by right-mouse clicking on the “synonym type” icon next to the name field. The first action is analogous to creating a new taxon above, and the user will be asked to choose a new parent taxon. The second is also relatively straightforward, and the new taxon will assume all information – synonymy, descriptions, etc. – previously belonging to the old taxon, which becomes its synonym; synonym type will be determined by whether the two are in the same homotypic group.

Changing a name’s status from taxon to synonym

This action will be called from the taxon’s toolbar, since the taxon’s name field has no icon to which to attach a context menu. Per default, all data attached to a taxon will automatically be transferred to the new taxon. Appropriate warnings will be shown for recently transferred data, unless the user has in application preferences explicitly asked that these warnings not be shown. The user can then look at all transferred descriptive data, for instance, and click on a “Remove all transferred data warnings” link at the top of the target taxon view.

Alternately, before moving the taxon, the user has the possibility of placing two taxon views side by side and dragging information from one view to the other. The user could then compare the descriptive data, drag to the target taxon that which he wants to keep, and manually delete that which he does not want to transfer. This is discussed in more detail below.

A question that has proven vexing in the past is what to do with included taxa when changing their parent to a synonym, especially when the move will require renaming the included taxa according to taxonomic rules. Here, we decided that no automatic renaming will take place. Rather, included taxa will be included in the new taxon, and revealed and marked with a “transferred data” warning icon in the taxonomic tree. If the move has caused an included taxon to violate taxonomic naming rules, the tree node will be underlined red.

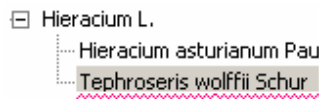


Figure 11: The genus of the recently-moved species causes an error to be shown.

Side-by-side editing of taxa

As described above, the default action when turning a previously accepted name into a synonym is moving all of a taxon’s information to a new taxon. For cases where this is not desired, the user must be given an effective way to first edit both the source and destination taxon. We will need to offer intuitive ways to enable side-by-side editing. Some possibilities: a menu item “enable side-by-side editing” followed by a window to choose the two taxa to be displayed; Windows-like buttons for “cascade” / “tile horizontal” / “tile vertical”; the user selects multiple taxa from the tree, then opens them all at once with the context menu.

This feature should prove useful in other editing contexts, for instance, when copying feature text from one taxon to another.

Note that application preferences will contain an option to mark all data which have been moved from another taxon, with a reference to the data’s original location. **to be completed**

A comparison mechanism for comparing two taxa and highlighting differences is under consideration.

Editing references, authors, types

Nomenclatural references and authors will be editable in the same way as the name, on the same line as the name in the freetext area; dealing with the ramifications of changing existing entries has already been described above. The form view will allow the user to edit the object’s individual fields.

The exact workflow for editing type and specimen information is to be determined, but will be based on the software used by the AlgaTerra project (<http://www.algaterra.net/>).

Editing concepts

As described above in “**Creating synonymy**”, concepts are made up of a taxonomic treatment’s name information combined with its author, nomenclatural, and descriptive feature information. The concept editing window shows the relations between a taxon’s concept and similar concepts in other treatments.

The concept editing window will function similarly to the name editing window, although the taxon name will not be editable in this view. Concepts can be added either from the toolbar or with carriage return in the freetext area. After thus initiating a new concept name, the user will be able to choose any combination of concept relationships – “congruent”, “included in”, “includes”, “overlaps”, etc. Parsing will function similarly to name editing, except that instead of nomenclatural reference, the “sec.” reference will be expected.

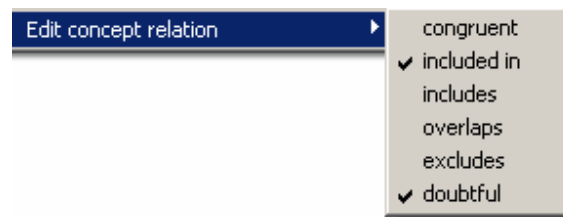


Figure 12: Choosing more than one relation from the concept's context menu.

Similar to the homotypic groups in the name editing window, there will be one group for identical (“congruent”) concepts, into which concepts can be drag-and-dropped. Experience has shown that most concepts created by taxonomists using the Berlin Model have been identical, so newly created concepts will be automatically put into the identical group.

Editing features (the entities formerly known as facts)

To be determined, but will implement the freetext / form view model. ****to be completed****

Duplicate detection windows

A departure from the Editor’s taxon-centric approach will be a module allowing the detection and merging of duplicate author, reference, and name entries, again drawing on experience from the Web Editor. Historically, imports into the Berlin Model resulted in a number of entries which differed only in how they were written or abbreviated in the source material. Missing from the Web Editor was an easy way for a taxonomist to compare entries and identify variations on the same entity, and to specify which variation should take precedence.

While this module has yet to be prototyped, it will take the form of a spreadsheet, with each entity’s unique fields displayed for identification. A grouping mechanism – most likely the “control” and mouse click method described above for performing an operation on multiple entries – will be implemented, along with a means of choosing a preferred entity; data associated with any of the grouped entries will henceforth be associated only with the preferred entity.

Integrity checker

The Editor will include a menu command for generating a report of all instances in the current dataset which violate integrity rules.