CRAFT

Chemical Reactivity and Fate Tool

Version 1.0

CRAFT Knowledge Base Editor User Manual

Molecular Networks GmbH Computerchemie
February 2009
http://www.molecular-networks.com
Contents

CRAFT Knowledge Base Editor

1. Introduction 1
   1.1. Short description and definitions 1
   1.2. Structure of the installation directory 1

2. Starting the program 3

3. User interface 7
   3.1. Main window 7
   3.2. Main menu 8
   3.3. Reaction types area 9
   3.4. Reaction rules area 9
   3.5. Example reactions area 9
   3.6. Status bar area and control buttons 10

4. Detailed GUI description 11
   4.1. Reaction type handling 11
      4.1.1. Find reaction type 11
      4.1.2. Create a new reaction type 12
      4.1.3. Edit a reaction type 12
      4.1.4. Test a reaction type 17
      4.1.5. Delete a reaction type 18
      4.1.6. Committing and reloading the CRAFT knowledge base 19
   4.2. Reaction rule handling 19
      4.2.1. Opening the CRAFT Editor Reaction Rules dialog 19
      4.2.2. Selection of a reaction rule 20
      4.2.3. Editing a reaction rule 20
      4.2.4. Closing the Reaction Rule Editor 21
      4.2.5. Annotation of reaction conditions 21
      4.2.6. Addition of new categories to the reaction conditions 21
      4.2.7. Deleting a category used to describe the reaction conditions 22
      4.2.8. Renaming a reaction condition category 22
      4.2.9. Annotation of the likelihood model and likelihood category 22
      4.2.10. Opening the Reaction Rule Editor from the main window 23
      4.2.11. Deleting a reaction rule using the CRAFT Editor main window 24
   4.3. Example reaction handling 24
      4.3.1. Overview on the Manage example reactions dialog 24
      4.3.2. Search an example reaction by name 25
      4.3.3. Import example reactions from files 26
      4.3.4. Delete selected example reactions 27
      4.3.5. Delete all example reactions 27
      4.3.6. Export selected example reactions to a file 28
      4.3.7. Change settings for the display of chemical structures 28
      4.3.8. Input of references for an example reaction 28
      4.3.9. Exit the Manage example reactions dialog 29
      4.3.10. Committing and reloading the CRAFT knowledge base 29
      4.3.11. Find example reactions 30
      4.3.12. Edit example reactions 30
      4.3.13. Delete example reactions 31
   4.4. Manage likelihood models 32
4.4.1. Reaction conditions of the UM-BBD likelihood model
4.4.2. Using (Q)SAR models as likelihood models
4.5. Manage ignorable products
4.6. Manage example reactions
4.7. Manage reaction conditions
4.8. Scan and repair database
4.9. Export database
4.10. Import database

5. CRAFT Annotations
5.1. Introduction
5.2. Annotating model
5.3. Implementation of CRAFT-RGE annotations
   5.3.1. Syntax and grammar definitions
   5.3.2. Default dictionary of annotations
   5.3.3. Hints and examples

6. Troubleshooting
6.1. Memory problems
6.3. Log file location
6.4. Reporting an error
1. Introduction

1.1. Short description and definitions

The CRAFT Knowledge Base Editor (later referred to as CRAFT Editor) allows the user to input, inspect, and modify the knowledge base which is used by the CRAFT Explorer to evaluate the chemical reactivity and the fate of chemicals. The knowledge base distinguishes reaction types, reaction rules, and example reactions.

**Reaction types** define the changes of a chemical compound in the course of a reaction. A set of *annotations* – which either generalize or constrain the sketched substructure fragment – refine the definition of a reaction type. These annotations are explained in more detail in the chapter ‘CRAFT annotations’ in the part ‘Default dictionary of annotations’.

A *reaction rule* adds an evaluation to a reaction type providing the user with a measure of its likelihood.

**Example reactions** can be linked to a reaction type to illustrate where a particular reaction was observed and reported in literature.

By employing the CRAFT Editor the user has full access to the knowledge base used for the prediction of the reactivity of chemicals. Therefore, the reasoning of the predictions is not a black box model but quite transparent.

Biotransformation rules originating from the University of Minnesota Biocatalysis/Biodegradation Database (UM-BBD) [http://umbbd.msi.umn.edu/; Ellis LBM, Roe D, Wackett LP (2006) "The University of Minnesota Biocatalysis/Biodegradation Database: The First Decade," *Nucleic Acids Research* 34: D517-D521] were input with the CRAFT Editor. They are available from the UM-BBD likelihood model packaged together with the CRAFT Explorer.

The CRAFT Editor provides a graphical user interface (GUI) assisting the user to work on the knowledge base for reactivity prediction. All users can easily implement additional reaction types and reaction rules or modify them without being obliged to have any special IT and programming skills.

1.2. Structure of the installation directory

During installation, the setup program creates the following folders in the installation directory of CRAFT applications (Figure 1). The *docs* folder contains the API documentation of all the CRAFT packages and programs. The *ext* folder is designed to hold the CRAFT plug-ins and their required libraries. Initially, this folder contains the default set of plug-ins distributed together with the CRAFT applications. The *lib* folder contains JAR files and binaries that the CRAFT Editor depends on (libraries). Finally, the *toxtree-stub* folder contains the binary *mn-toxtree-stub.jar* file that is used by the ToxTree plug-in for the CRAFT Explorer.
Figure 1  CRAFT installation directory structure

The binary *CRAFT.Editor-v.1.0.exe* is the executable file of CRAFT Editor – the Java launcher that starts Java™ runtime environment and loads the CRAFT Editor application from the JAR file *mn-kbe.jar*. 
2. Starting the program

During the installation process of CRAFT applications in Microsoft® Windows™ XP environment, the icons for CRAFT Editor are created on the user desktop and in the Start menu. Use either of these icons to start the application.

If the application is used under the Linux operating systems, in order to start CRAFT Explorer, change into the directory where you have installed it and type at the command prompt:

$> java –jar mn-kbe.jar

This will start the CRAFT Editor application; however, the start-up window does not appear in this case.

On starting the CRAFT Editor application, the start-up window as shown in Figure 2 appears.

![Figure 2: CRAFT Editor start-up window](image)

By default, the CRAFT Editor uses the following user credentials to connect to the CRAFT knowledge base instance:

- **MySQL server host name** - `localhost`
- **MySQL service port** – `3360`
- **User name** - `craft`
- **Password** - `craft`
- **Database name** – `craft`

While the application loads, it tries to connect to the default CRAFT knowledge base using the default user credentials (or the previously saved ones, see below), and, if a login attempt succeeds, the main application window will appear. Otherwise, a **Database login** error message (see Figure 3) pops up.

![Figure 3: CRAFT Editor – database login error message](image)

After leaving this dialog by pressing the OK button the CRAFT Editor reports the error message again (see Figure 4).
In this case, make sure that either the local MySQL server running the CRAFT knowledge base is online, or you have an account on a remote MySQL database that hosts the CRAFT knowledge base. Either way, prepare and verify the user account credentials and press the OK button to close this error message. The Database login dialog window (see Figure 5) will appear.

Please enter the required information into the input fields of this dialog in order to repeat the login attempt:

- **Host** field should contain the name or IP address of the MySQL database server hosting the CRAFT knowledge base. By default, the reserved host name localhost (or its equivalent IP address 127.0.0.1) is assumed for a MySQL server instance running on the same PC where the CRAFT applications are installed.
- **Port** field indicates the TCP port used to connect to the MySQL service – the value of 3306 is the default value used by the MySQL service.
- **User name** field should contain the login name of a valid MySQL account. The default user name for CRAFT applications is assumed to be craft.
- **Password** field should contain the password of the MySQL account. The default password for CRAFT applications is assumed to be craft.
- **Database schema** field should contain the name of the MySQL database schema that contains the CRAFT knowledge base. By default, craft database schema is assumed.

As soon as you have entered all the login credentials, you may want to check the **Save login data (including password)** checkbox in order to store the login credentials for the next time. If done so, CRAFT Editor stores the login information in the CRAFT configuration file craft.ini located at the user home directory (e.g. C:\Documents and Settings\<User name>\craft.ini if running on Microsoft® Windows XP operating system or /home/<user name>/craft.ini on Linux systems). This configuration file is used by all CRAFT applications to connect to the database.
Please note that this configuration file contains the database login password in an encrypted form. Even though the well-known secure encryption algorithms (SHA1 and MD5) are employed to encrypt the password, it is still theoretically possible to decrypt it gaining an unauthorized access to the CRAFT knowledge base.

Click **Connect** button to connect to the CRAFT knowledge base. If all the credentials are correctly entered and the MySQL server hosting the CRAFT knowledge base is running and accessible, the application establishes the database connection and the main window will appear.

If the application still could not connect to the database, the **Database login** dialog will appear again. Please correct any possible errors and try again.

The application repeats the login attempts 3 times. If after all these attempts the user fails to provide correct login credentials, the error dialog appears (Figure 6) and the application quits.

![Figure 6] CRAFT Editor database login failure window
3. User interface

If the connection to the CRAFT knowledge base is successfully established, the main application window will appear.

3.1. Main window

Figure 7 displays the main application window of the CRAFT Editor application. The main window of the CRAFT Editor application is separated into certain control elements and areas:

1. Main application menu
2. Reaction types area
3. Reaction rules area
4. Example reactions area
5. Status bar area and control buttons

Each of these areas serves either to control the application behavior or to indicate the status of the application.

Just after application start, the main window displays the information about the CRAFT knowledge base it connected to both in the status bar and in the window caption. The database information is shown permanently in the window caption while the status bar displays also informative messages accordingly.

![Image of CRAFT Editor main application window with labeled areas](image)

Figure 7  CRAFT Explorer main application window. The red lines show the boundaries between the different areas.
3.2. Main menu

The main application menu (Figure 7, pos. 1) controls the most common functionality of the CRAFT Editor application. It consists of Database, Manage, Tools, and Help submenus.

Database menu

The Database menu contains commands to connect to the CRAFT knowledge bases and to close the application:

- \(\text{Connect…}\) (keyboard shortcut Alt-C) – connect to different CRAFT knowledge base instances and also to save the connection data for the next time.
- \(\text{Exit}\) (Alt-X) – close the application.

Choosing the Database \(\Rightarrow\) Connect… command will open the Database login dialog window (Figure 5) so that the user can specify new connection credentials and also save them for the next time.

Manage menu

The Manage menu contains commands allowing the user to access the different kinds of data which is comprised in the knowledge base: Reaction rules, likelihood models, ignorable products, example reactions, and reaction conditions.

- \(\text{Reaction rules…}\) (Alt-R) – opens the reaction rule browser. A dialog window with a list of all available reaction rules is shown. The user gets more detailed information on a reaction rule after selection of a particular one from the reaction rule list. In addition, the rule can be edited.
- \(\text{Likelihood models…}\) (Alt-L) – manages the likelihood models. The implemented likelihood models for the evaluation of the reactivity are listed. Details are displayed for a likelihood model selected by the user. In addition, the user can create new likelihood models and modify or delete existing ones.
- \(\text{Ignorable products…}\) (Alt-G) – opens a list of products which are not added to the degradation tree. For a selected chemical the structure diagram is visualized. Furthermore, the user can add additional compounds either by drawing them or by import from a structure file. Compounds which should not be ignored can be deleted from the list. The entire list of ignorable chemicals can be exported to a structure file.
- \(\text{Example reactions…}\) (Alt-E) – manages example reactions. All available reactions are listed and can be annotated. In addition, example reactions can be imported, exported, and deleted.
- \(\text{Reactions conditions…}\) (Alt-D) – defines the annotations which are available to specify reaction rules, e.g. the species, compartments, oxygen demand and biotic conditions which are valid for this reaction rule.

Tools menu

The Tools menu contains commands to easily import, export, and maintain the CRAFT knowledge base stored in the relational database:

- \(\text{Scan and repair…}\) (Alt-U) – opens the Scan database dialog window which enables the user to scan the database for missing cross-references.
- \(\text{Export database…}\) – shows a dialog allowing the user to export the contents from the database to a single file.
• Import database… – opens a dialog to import a knowledge base from a single file to the relational database. The content will be added to the already existing contents of the current knowledge base which is in use.

Help menu

The Help menu contains the entry About. Selecting this entry from the menu displays the About: CRAFT Knowledge Base Editor 1.0 window (Figure 8). The first tab About CRAFT Knowledge Base Editor (Figure 8 a)) contains an application description, its version number, a link to the home page of Molecular Networks and credits information. Additionally, credits are listed in the API documentation and the source code of application. The second tab CRAFT knowledge base statistics (Figure 8 b)) presents selected statistical data on the currently used CRAFT knowledge base.

![Figure 8 CRAFT Editor About window](image)

3.3. Reaction types area

The reaction types area (Figure 7, pos. 2) allows the user to browse, enter, modify and delete new reaction types. A more detailed description of the handling of reaction types follows in the section “4.1 Reaction type handling” (page 11).

3.4. Reaction rules area

This area (Figure 7, pos. 3) shows all available reaction rules for the reaction type selected from the list of reaction types on the left hand side of the reaction types area. The user can add, delete, and modify the reaction rules by clicking the buttons “New…”, “Edit…”, or “Delete” resp. on the right hand side of this area. A more detailed explanation on manipulating reaction rules is given in the section “4.2 Reaction rule handling” (page 19).

3.5. Example reactions area

If the user selects a reaction type in the reaction type area all example reactions linked to this reaction type are shown in a list in the example reactions area (Figure 7, pos. 4). For
a more comprehensive description of the functionality of the example reactions area the reader is referred to the section “4.3 Example reaction handling” (page 24).

3.6. Status bar area and control buttons

The status bar area (Figure 7, pos. 5) contains information about the applications activity and current state, e.g. the information about the CRAFT knowledge base to which the application is connected.

The status bar also displays short information about application errors, if any.

In the bottom right corner the control buttons “Commit”, “Reload”, and “Exit” are placed:

- In case the current instance of the knowledge base was modified with the CRAFT Editor the “Commit” button is activated and can be used to save these changes in the relational database.
- The button “Reload” updates the current instance of the knowledge base of the CRAFT Editor by loading the knowledge base again from the relational database. This option might be used if several users extend the knowledge base simultaneously.
- The “Exit” button terminates the CRAFT Editor.
4. Detailed GUI description

In this chapter both the various menu entries and the different areas of the CRAFT Editor main window are explained in more detail. The description of the usage of the different functions starts with the handling of reaction types. A part on reaction rules including the reaction rule manager (Manage → Reaction rules…) follows. Then, the addition of example reactions to the knowledge base is described. Furthermore the following entries from the drop-down menus are covered:

- Manage → Likelihood models…
- Manage → Ignorable products…
- Manage → Example reactions…
- Manage → Reaction conditions…
- Tools → Scan and repair…
- Tools → Export database…
- Tools → Import database…

4.1. Reaction type handling

The reaction types area in the CRAFT Editor main window shows a list of the reaction types on the left hand side which are present in the current instance of the knowledge base. This instance of the knowledge base was defined by the connection to the relational database either after the first start of the CRAFT Editor or by selecting the function Connect… from the Database menu.

Five different buttons are placed at the bottom of the reaction type area:

- Find… – searches a reaction type by name.
- New… – creates a new reaction type.
- Edit …– edit the selected reaction type.
- Test… – test the selected reaction type.
- Delete – delete the selected reaction type(s).

4.1.1. Find reaction type

Clicking the “Find…” button opens the input dialog (Figure 9) for the search of a reaction type by name.

![Input search pattern dialog](image)

Figure 9 CRAFT Editor Input search pattern dialog to find a reaction type by name

After confirmation of the search pattern input by clicking the “OK” button in Figure 9 the next matching reaction type starting from the current position is highlighted in the reaction type list and the structural alert, related reaction rules and example reactions are shown in the main window (Figure 10).
In order to search again for the next hit the user can subsequently press the “Find…” and the “OK” button multiple times.

4.1.2. Create a new reaction type

A new reaction type named “<New reaction type>” is created in the reaction types list by the “New…” button in the reaction types area (Figure 12).

The structural alert and all annotations for a new reaction type can be defined by clicking on the “Edit…” button.

4.1.3. Edit a reaction type

Once a reaction type is selected from the reaction type list on the left hand side of the reaction types area in the CRAFT Editor main window the “Edit…” button is activated. Clicking the “Edit…” button will then act on the selected reaction type. Either the user can define a new reaction type or an already existing reaction type can be modified.
Here, we illustrate the creation of a new reaction type. After clicking the “Edit …” button the new window CRAFT Reaction Type Editor appears on the screen.

The CRAFT Reaction Type Editor embeds the open source molecule editor JChemPaint [http://jchempaint.sourceforge.net] provided by CDK. JChemPaint was originally implemented by Christoph Steinbeck. In the meantime, JChemPaint became a part of the Chemistry Development Kit (CDK; http://cdk.sourceforge.net).

Figure 12  CRAFT Reaction Type Editor window

The detailed description of the functionality of JChemPaint is beyond the scope of this user manual. The interested reader may try to get additional information on the use of JChemPaint from the original documentation and/or the corresponding project pages the web site. Nevertheless, sketching of a structural alert should be intuitive and not a big obstacle.

The name of the reaction type can be modified by editing the line above the toolbar on the top of the Reaction Type Editor window (Figure 13). The name “<New reaction type>” is replaced by “1,4-dihydropyridine to pyridine derivative”.

Figure 13  CRAFT Reaction Type Editor window showing a 1,4-dihydropyridine as structural alert
In the following, the generation of a reaction type is shown for the dehydrogenation of a 1,4-dihydropyridine to a pyridine derivative (Figure 14).

![Figure 14 Dehydrogenation reaction of 1,4-dihydropyridines to a pyridine derivative](image)

The generation of this reaction type can be accomplished by the following steps:

- First select the six-membered ring illustrated in the toolbar of JChemPaint.
- Click once with the left mouse button on the empty, white space in the center of the JChemPaint window. A cyclohexane ring appears. Let us assume we number the carbon atoms clockwise from one to six starting from the carbon atom at the bottom.
- Select the button showing a single bond in the toolbar.
- Click on the single bonds between C2 and C3 as well as between C5 and C6 where the single bonds should become double bonds.
- Add a carbon atom connected by a single bond to carbon atom C1 by clicking on this atom. This carbon atom will be referred to as C11.
- Add a carbon atom connected by a single bond to carbon atom C4 by clicking on this atom. This carbon atom will be referred to as C41.
- Select the nitrogen atom from the toolbar.
- Then click on C1 to replace the carbon by a nitrogen atom. C1 becomes N1.
- Select the hydrogen atom from the toolbar.
- Click on atom C11 to change it into a hydrogen atom. C11 becomes H11.
- Click on atom C41 to change it into a hydrogen atom. C41 becomes H41.

These steps are sufficient to define the structural alert for this example reaction type. The result is shown in Figure 13.

The information about the bond order changes and other annotations which are necessary to get a working reaction type are explained below:

- Click on H11 with the right mouse button. A context menu is displayed.
- Select the item “Annotate...” at the bottom of this context menu. The window “Annotations for atom a7 [H]” pops up (Figure 15).
Check the box “Delete atom” in the part “Transformation annotations” (Figure 16).

Close the window pressing the “OK” button. Moving the mouse above atom H11 will display the annotations made for atom H11.

Repeat the preceding four steps for H41.

Click with the right mouse button on the bond between N1 and C2 and select “Annotate…” from the context menu.

Select the entry “Double” from the list box next to entry “Replace with bond of type” in the part “Transformation Annotations” in the window Annotations for bond b1(Figure 17).

Close the window by pressing the “OK” button. Moving the mouse above the bond between N1 and C2 will display its annotations.
Click with the right mouse button on the bond between C2 and C3 and select “Annotate…” from the context menu.

Select the entry “Single” from the list box next to entry “Replace with bond of type”.

Close the window by pressing the “OK” button.

Click with the right mouse button on the bond between C3 and C4 and select “Annotate…” from the context menu.

Select the entry “Double” from the list box next to entry “Replace with bond of type”.

Check the box “Structural alert” in the part “Info annotations”.

It is advisable to make the annotation indicating the substructure serving as structural alert after having finished all other annotations.

Close the window by pressing the “OK” button.

Enter an optional comment in the box “Comments” at the bottom of the CRAFT Reaction Type Editor.

In the bottom right corner of the CRAFT Reaction Type Editor there are four buttons:

- Redraw – redraws the structural alert in the CRAFT Reaction Type Editor.
- Test… – opens a structure editor window to sketch a chemical which is used to test the reaction type. This button has identical functionality as the “Test…” button placed at the bottom of the reaction types area of the CRAFT Editor main window and is described later (please refer to section 4.1.4, page 17).
- OK – leaves the CRAFT Reaction Type Editor and accepts all changes which were made.
- Cancel – leaves the CRAFT Reaction Type Editor without making any changes of the knowledge base.

A more comprehensive overview on all available annotations as well as a technical description of the implementation is given in the section “5
4.1.4. Test a reaction type

Clicking on the “Test…” button either in the reaction type area of the CRAFT Editor main window or in the Reaction Type Editor window opens the structure editor where the user specifies a chemical structure to be used to test the currently selected reaction type. An example compound for the reaction type created above is shown in Figure 18.

In addition to the structure editor functionality the “Testing” window has two checkable options and three buttons:

- Preprocess structure – if checked, normalizes the structure of the test compound.
- Consider ignorable – if not checked, outputs also products of the reaction which are defined as ignorable products.
- New – clears the sketched chemical if this is applicable.
- Run reaction – starts the reaction generation.
- Done – returns to the previous window.

Starting the reaction generation for the example compound in Figure 18 by pressing the button “Run reaction” delivers the result shown in Figure 19 as expected.
Both the reactant and the product(s) are depicted as output of the reaction generation. The reaction type is applied only once to the test compound. The slider on the right hand side of the window in Figure 19 controls the zoom level. In addition, the display of all hydrogen atoms can be toggled with the "ån" button in the upper right corner of the window.

If a reaction type is not applicable the following message will be displayed (Figure 20).

### 4.1.5. Delete a reaction type

If a reaction type is selected and highlighted in the reaction type list then this reaction type can be deleted by pressing the “Delete” button in the reaction type area of the CRAFT Editor main window. In order to avoid that a reaction type is deleted by mistake a dialog window asks for the confirmation of this action.
in the text

4.1.6. Committing and reloading the CRAFT knowledge base

When the user closes the Reaction Type Editor dialog the changes made are not committed to the CRAFT knowledge base. In this case the user either submits all changes to the CRAFT knowledge base by clicking the "Commit" button or reverts all changes by clicking the "Reload" button in the CRAFT Editor main window (Figure 22).

![Figure 22](image.png)

Figure 22 Save ("Commit" button) or discard ("Reload" button) changes made on reaction rules in the CRAFT Editor main window.

The selection of the command "Reload" prompts the user the dialog shown in Figure 23.

![Figure 23](image.png)

Figure 23 Dialog box asking the user to confirm whether all unsaved changes should be discarded and whether the CRAFT knowledge base should be reloaded from the database.

4.2. Reaction rule handling

The user can handle reaction rules either by selecting the entry "Manage → Reaction Rules..." from the menu bar - which opens the CRAFT Editor Reaction Rules dialog (Figure 24) and allows one to select and manipulate a reaction rule - or by clicking the following buttons from the reaction rule area:

- New… – creates a new reaction type.
- Edit… – edits the selected reaction type.
- Del… – deletes the selected reaction type(s).

4.2.1. Opening the CRAFT Editor Reaction Rules dialog

After opening the CRAFT Editor Reaction Rules dialog by selecting the entry "Reaction Rules..." from the "Manage" menu the user can browse the list of reaction rules in the upper left quadrant of this window (Figure 24).
If the user wants to close the reaction rule manager this can be done by clicking the “Done” button in the bottom right corner of the window.

### 4.2.2. Selection of a reaction rule

The selection of a reaction rule from this list by clicking it with the left mouse button causes that the comments related to a rule are displayed in this dialog (Figure 25, upper right corner). In addition, the reaction type name, the reference of the reaction type, the reaction conditions (for more details please refer to section “4.4.1 Reaction conditions of the UM-BBD likelihood model”, page 34), the likelihood model, and the likelihood category are shown (Figure 25, bottom half).

### 4.2.3. Editing a reaction rule

In case a reaction rule is selected as in Figure 25 the “Edit highlighted rule...” button is activated. Clicking on this button results in opening the CRAFT Editor Reaction Rule Editor dialog (Figure 26). This dialog supports annotating the reaction rule name, the reference, the reaction conditions (biotic, oxygen, compartments, and species), the
likelihood model, the likelihood category, and comments. The fields “Reaction rule name”, “Specification reference”, and “Comments” contain text which can be edited after clicking the corresponding white field with the left mouse button.

4.2.4. Closing the Reaction Rule Editor

If the user wants to close the Reaction Rule Editor dialog (Figure 26) there are two alternatives. Clicking the “OK” button will confirm any changes made and will commit them to the CRAFT knowledge base, whereas pressing the “Cancel” button will discard them.

![CRAFT Editor Reaction Rule Editor dialog](image)

Figure 26 CRAFT Editor Reaction Rule Editor dialog

4.2.5. Annotation of reaction conditions

The biotic conditions, the oxygen conditions, compartments, and species can be selected by clicking on an entry in the corresponding lists. Selected items are highlighted with white letters and a blue background. Multiple selections can be made by simultaneously pressing <Ctrl> on the keyboard and clicking with the left mouse button. In the CRAFT Editor all reaction conditions which are needed to cover the reaction rules from the UM-BBD are available in the distributed version of CRAFT. As the UM-BBD does not provide information about the species only the category “Not specified” is defined in the CRAFT Editor by default.

4.2.6. Addition of new categories to the reaction conditions

New categories can be added by clicking the “Add” button below the list of categories for the reaction condition of interest. The dialog to input a new species is shown in Figure 27. A new species is added by typing in the name in the white box and terminating the dialog by clicking the “OK” button. The dialogs for the biotic conditions, the oxygen conditions, and compartments are similar to the one shown in Figure 27.
4.2.7. Deleting a category used to describe the reaction conditions

All categories for the definition of a particular reaction condition can be eliminated by selecting a particular category, e.g. the category “Biotic” for the biotic conditions, and by clicking the “Delete” button. If the MySQL database server uses the InnoDB engine and the reaction condition to be deleted is used by other rules then an error message will be displayed (Figure 28).

![Figure 28 Error message caused by trying to delete reaction conditions in a reaction rule which is used by other reaction rules](error-message.png)

If MySQL database server does not use the InnoDB engine then deleting a category with the “Delete” button may have unexpected side effects, i.e. annotations for other reaction rules might be deleted, too. Therefore, the use of InnoDB is recommended for the CRAFT Editor and CRAFT Explorer.

Deleting a category for the reaction conditions affects the entire CRAFT knowledge base and does not act on an individual reaction rule only. For an individual reaction rule the reaction conditions can be easily deselected by clicking on the corresponding item. If multiple reaction conditions from one group are selected simultaneously, e.g. “Water” and “Soil” as compartment, then the user can selectively deselect an annotated reaction condition from by pressing the <Ctrl> key on the keyboard and clicking on the corresponding item.

4.2.8. Renaming a reaction condition category

A double-click in the table showing the available reaction conditions starts the mode in which the user can edit the name of a category.

4.2.9. Annotation of the likelihood model and likelihood category

The likelihood model and the likelihood category can be selected from the list boxes as shown in Figure 29 a) and b).

The likelihood models and likelihood categories are defined using the entry “Likelihood models…” from the “Manage” menu (please refer to section 4.4 Manage likelihood models on page 32 for more details).
4.2.10. Opening the Reaction Rule Editor from the main window

Alternatively, the CRAFT Editor Reaction Rule Editor dialog can be opened by pressing the “New...” or “Edit...” button in the reaction rule area of the CRAFT Editor main Window. The “New...” button is activated if a reaction type was selected. The “Edit...” button is activated if a reaction type and a reaction rule were selected as shown in Figure 30. The usage of the CRAFT Editor Reaction Rule Editor is exactly the same as described above.

![Reaction Rule Editor](image1.png)

Figure 29 Selection of the likelihood model and the likelihood category in the CRAFT Editor Reaction Rule Editor dialog

![Main Window](image2.png)

Figure 30 Selection of a reaction rule for a selected reaction type in the CRAFT Editor Main Window
4.2.11 Deleting a reaction rule using the CRAFT Editor main window

In order to delete a reaction rule with the “Delete” button in the reaction rules area of the CRAFT Main Window a reaction rule has to be selected first. The user has to confirm whether a reaction rule should be deleted from the CRAFT knowledge base (Figure 31).

![Figure 31 Dialog box asking the user to confirm whether a reaction rule should be deleted from the CRAFT knowledge base](image)

4.3 Example reaction handling

All reaction types which are available in the CRAFT knowledge base can be linked with one or several example reactions. This functionality is supported by the example reactions area in the CRAFT Editor main window.

If a reaction type is selected in the list of reaction types in the reaction type area of the CRAFT Editor main window the following buttons in the example reactions area are activated:

- ![Add…](image) – adds new example reaction(s) to the selected reaction type
- ![Find…](image) – finds example reactions which are already present in the CRAFT knowledge base
- ![Edit …](image) – allows the user to edit the current set of example reactions.
- ![Delete](image) – deletes the selected example reaction.

4.3.1 Overview on the Manage example reactions dialog

Clicking the “Add…” button in the example reaction area opens the CRAFT Editor Manage example reactions window shown in Figure 32. On the left hand side of this window there is a scrollable list of all currently available example reactions. On the top of the window there is a search bar. At the bottom of the window there are a couple of control buttons:

- ![Import file(s)](image) – imports example reactions from a single or several RXN files
- ![Delete selected](image) – deletes the currently selected example reactions
- ![Clear all](image) – clears all example reactions present in the current knowledge base
- ![Export file(s)](image) – exports the currently selected example reactions either as a RDF file or a CML file
- ![Commit](image) – saves the changes made with the CRAFT Editor in the currently active instance of the database
- ![Reload](image) – reloads the CRAFT knowledge base from the relational database of the currently active instance and discards all changes made
- ![OK](image) – closes the CRAFT Editor Manage example reactions dialog
- ![Cancel](image) – leaves the CRAFT Editor Manage example reactions dialog and discards all changes made
In addition to these buttons there are some pictograms on the right hand side of the window which are also active if an example reaction is selected from the list of example reactions.

- ☓ Reset search filter next to the search bar – clears the query in the search bar
- ☐ Toggle display of hydrogen atoms – switches the display of hydrogen atoms on and off
- ✎ Zoom out, ✊ Original zoom, and ✱ Zoom in – change the zoom level used for the depiction of chemical structures
- ✎ Add reference – allows the user to add a reference to an example reaction
- ✱ Delete reference – allows the user to delete a reference of an example reaction

Figure 32  CRAFT Editor Manage example reactions dialog

4.3.2. Search an example reaction by name

The user can search the list of example reactions by name. In the top of the CRAFT Editor Manage example reactions dialog (Figure 32) there is a field in which a query for the name or a part of the name of the example reaction can be specified.

While the user is typing the query text the scrollable list of example reactions on the left hand side of the dialog window is continuously updated. The query text can match the reactions name at any position and not necessarily from the beginning. An example for the query text “xant” is shown in Figure 33.

Clicking the “☓ Reset search filter” button next to the search bar the user can clear the query in the search bar. The entire list of example reactions is shown again in the CRAFT Editor Manage example reactions dialog.
4.3.3. Import example reactions from files

The set of example reactions can be extended by clicking the “Import file(s)” button. The RXN file format is supported. After a suitable input file is selected (Figure 34) and the file dialog is closed by pressing the “Open” button the new reaction is added at the end of the example reaction list.

The reaction name, comments to the reaction, and reference(s) can be added in the corresponding fields of the CRAFT Editor Manage example reactions dialog which are marked with a red rectangle in Figure 35.
4.3.4. Delete selected example reactions

If a single or several example reactions are selected in the example reaction list the user can delete them by pressing the button "Delete selected" at the bottom of the CRAFT Editor Manage example reactions dialog. A window is displayed which asks the user to confirm that all selected reactions should be deleted from the CRAFT knowledge base (Figure 36). Clicking the "Yes" button will delete the selected reactions.

4.3.5. Delete all example reactions

The "Clear all" button allows the user to delete all example reactions in a single turn. Again a dialog box opens which asks to confirm whether all example reactions should be deleted (Figure 37). Clicking the “Yes” button will delete all example reactions. Otherwise close the dialog box by clicking the “No” button.
4.3.6. Export selected example reactions to a file

A set of selected example reactions can be exported to a file by clicking the “Export file(s)” button. The reactions are written either in RDF file format or as CML. After browsing to the directory where the file should be stored the user enters the name of the file in the field “File name” at the bottom of the export file dialog (Figure 38) and confirms the choice by clicking the “Save” button.

Figure 38 Dialog box asking the user to confirm whether a reaction rule should be deleted from the CRAFT Knowledge base

4.3.7. Change settings for the display of chemical structures

On the right hand side of the CRAFT Editor manage example reactions dialog there are four buttons which can be used to modify the way the chemical structures are displayed:

- Toggle display of hydrogen atoms – switches the display of hydrogen atoms on and off. In some cases it might be easier to inspect a structure if its depiction is not overcrowded with implicit hydrogen atoms. On the other hand the display of hydrogen atoms is desirable if they are involved in the reaction.
- Zoom out – decreases the zoom level used for the depiction of chemical structures
- Original zoom – resets the zoom level used for the depiction of chemical structures
- Zoom in – increases the zoom level used for the depiction of chemical structures

4.3.8. Input of references for an example reaction

If an example reaction is selected the following buttons are also active:

- Add reference – allows the user to add a reference to an example reaction
- Delete reference – allows the user to delete a reference of an example reaction

All fields of a reference which is displayed can be edited after a double-click in the corresponding field.
Figure 39  Manage example reactions dialog displaying a graphical representation of an example reaction, the literature reference, the source of this entry in the database with URL and year: a) original state; b) state after clicking the “Add reference” button.

A reference for an example reaction which is selected from the list on the left hand side of the CRAFT Editor Manage example reactions dialog can be deleted by clicking the “Delete reference” button.

4.3.9. Exit the Manage example reactions dialog

The user closes the CRAFT Editor Manage example reactions dialog by clicking the “OK” button. In this case the user accepts all changes which were made.

By clicking the “Cancel” button all changes made in the CRAFT Editor Manage example reactions dialog are discarded and the window is closed.

Closing the CRAFT Editor Manage example reactions dialog does not submit any changes to CRAFT knowledge base stored in the relational database.

4.3.10. Committing and reloading the CRAFT knowledge base

Changes made on the CRAFT knowledge base with the CRAFT Editor Manage example reactions dialog can be saved or discarded in two ways:

a) The user submits or discards changes made on the current instance of the CRAFT knowledge base to the relational database by clicking the “Commit” or the “Reload” button resp.

b) The user exits the CRAFT Editor Manage example reactions window by clicking the “OK” button and uses the “Commit” or the “Reload” button in the CRAFT Editor main window.

Committing and reloading the CRAFT knowledge base corresponds to the description given in section 4.1.6 (page 19).
4.3.11. Find example reactions

The CRAFT Editor supports to automatically look up example reactions matching to a selected reaction type among those reactions which are present in the CRAFT knowledge base. The user clicks the “Find…” button in the example reactions area of the CRAFT Editor main window. The progress window (Figure 40) displays information on the status of the retrieval of matching example reactions.

After termination of the search for example reactions the progress window is closed automatically and a window with a message how many examples were found is displayed (Figure 41).

The user closes this window with the “OK” button. The example reactions which were retrieved from the CRAFT knowledge base are listed in the example reactions area of the CRAFT Editor main window (Figure 42).

4.3.12. Edit example reactions

The “Edit …” in the CRAFT Editor example reactions area allows the user to edit the current set of example reactions which are shown for the currently selected reaction type. The window below (Figure 43) shows a list of example reactions on the left hand side. If an item from this list is selected this example reaction is depicted (Figure 44). The currently selected example reaction can be removed from the list by clicking the red delete button (🗑️).
Figure 43  Window for browsing and editing example reactions

Reaction comments, the reference, URL, and year can be edited for the example reaction which is currently displayed. After clicking in the corresponding areas of the window the user can modify the existing text or can add new text.

Figure 44  Depiction of an example reaction

Clicking the “Ok” button closes the window. All changes made can be submitted to the CRAFT knowledge base by clicking the “Commit” button on the CRAFT Editor main window. Leaving the dialog with the “Cancel” button has the effect that all changes made are discarded.

4.3.13. Delete example reactions

If an example reaction linked to a particular reaction type is displayed in the list in the CRAFT Editor example reactions area this link can be removed by clicking the “Delete” button next to this list (Figure 45).
Neither an additional confirmation is requested nor a warning message is displayed if an example reaction is removed. All changes can be reverted by reloading the CRAFT knowledge base by pressing the “Reload” button in the CRAFT Editor main window, whereas clicking the “Commit” button will save all changes made to the knowledge base.

**4.4. Manage likelihood models**

In CRAFT two likelihood models are available by default.

- The first likelihood model implements the rule base from the UM-BBD.
- The second one is a reference implementation of a (Q)SAR reactivity model. The implemented (Q)SAR model predicts the hydrolytic stability of esters.

The user can inspect and modify the existing models by opening the CRAFT Editor Likelihood models dialog from the entry “Likelihood models…” in the manage menu.

**Figure 46** CRAFT Editor **Likelihoods models** window

Furthermore, additional models can be added. Clicking the “Add…” below the list of available likelihood models adds a new likelihood model. The name of the new likelihood model can be specified by the user in the dialog shown in Figure 47.

**Figure 47** a) Addition of a new likelihood model. b) The name of the new likelihood model is displayed in the likelihood models list.
A currently selected likelihood model can be deleted with the “Delete” button below the list of likelihood models. This action is forbidden if the likelihood model is referenced.

If a likelihood model is selected a brief description of this model is given in the “Comments” frame and a reference is shown (Figure 48). The individual categories of a likelihood model are shown in the bottom half of the CRAFT Editor Likelihood models dialog. On the left hand side the different categories are shown for the selected likelihood model. On the right hand side a few explanations for the selected category are displayed.

![Figure 48 Likelihood models available in the CRAFT Editor](image)

**Figure 48** Likelihood models available in the CRAFT Editor

a) Detailed view for the UM-BBD likelihood model  
b) Detailed view for the reference (Q)SAR model

New likelihood model categories for the selected likelihood model can be added by clicking the “Add…” button in the bottom left corner of the CRAFT Editor Likelihood models dialog. The name of the category and the value can be edited after a double-click with the left mouse button on the corresponding fields.

![Figure 49 Likelihood models available in the CRAFT Editor](image)

**Figure 49** Likelihood models available in the CRAFT Editor

A selected likelihood model category can be deleted by clicking the “Delete” below the list of likelihood model categories.
Except for the list of available likelihood models all fields, e.g. the “Comments”, “Reference”, and “Category comments” fields can be edited after a double-click with the left mouse button.

4.4.1. Reaction conditions of the UM-BBD likelihood model

The UM-BBD likelihood model distinguishes the following categories:

- Very likely reaction
- Likely reaction
- Neutral
- Unlikely reaction
- Very unlikely reaction
- No decision

4.4.2. Using (Q)SAR models as likelihood models

The reactivity of a compound can also be estimated by (Q)SAR model(s). The implementation of a (Q)SAR model in CRAFT may serve as a reference implementation for other (Q)SAR models. The (Q)SAR model implemented in the class `com.molecularnetworks.reactivity.model.EsterHydrolysisModel` contains a simple evaluation of the reactivity related to the hydrolysis of esters. This evaluation is based on experimental data.

More details are given in the Comment fields for this likelihood model in the CRAFT Editor Likelihood model window.

4.5. Manage ignorable products

All chemical compounds defined as ignorable products will not be displayed as product generated by the CRAFT Explore reaction generation engine if the reaction rules are applied to a query compound. These ignorable products are specified via the entry “Ignorable products...” in the manage menu. The execution of this function opens the CRAFT Editor Manage ignorable products dialog (Figure 50).

![Figure 50 CRAFT Editor Manage ignorable products window](image)

On the left hand side the names of all ignorable products defined so far are displayed. If a particular chemical is selected from this list its 2D structure diagram is shown on the right...
hand side. In addition to these two elements this dialog has six buttons at the bottom of the window:

- Draw and add – allows the user to sketch a new ignorable product and to add it to the list
- Add from file – adds further ignorable products read from a file
- Delete – deletes the selected ignorable products without any further warning
- Export to file – saves the selected ignorable products to a structure file
- OK – commits all changes made to the knowledge base and terminates the window
- Cancel – closes the window discarding all changes made

The command “Draw and add” opens the CRAFT Editor Edit ignorable products window (Figure 51). In this window the user can sketch the structure. If the checkbox “Auto saturate” is checked all implicit hydrogen atoms are added automatically to the sketched chemical compound.

The button “New” clears everything the user sketched so far with the structure editor.

The sketched compound is added to the list of ignorable products if the user exits the structure editor with the “OK” button. The changes are discarded if the “Cancel” button is used instead of the “OK” button.

![Figure 51 CRAFT Editor Edit structures window](image)

The name of an ignorable product can be edited after a double-click on the corresponding entry in the list on the left hand side.

4.6. Manage example reactions

Selecting the entry “Example reactions…” from the Manage menu opens the CRAFT Editor Manage example reactions dialog. Alternatively, the user can click the “Add…” button in the example reaction area.

The CRAFT Editor Manage example reactions dialog is described in more detail in section 4.3.1 (page 24).
4.7. Manage reaction conditions

The last entry in the Manage menu is the command “Reaction conditions…” which is used to define all needed reaction conditions. The biotic conditions, the oxygen conditions, the compartments, and the species can be defined. Additional categorical attributes can be added to each of these four groups of reaction conditions by clicking the “Add…” button in the corresponding part of the window shown in Figure 52.

Figure 52   CRAFT Editor **Named conditions** window

An attribute which is selected in a list can be deleted by clicking the corresponding “Delete” button.

The usage of this window is exactly the same as for the corresponding part in the CRAFT Editor Reaction Rule Editor dialog which is described in sections 4.2.6 (page 21) and 4.2.7 (page 22). Therefore, the reader please may refer to these sections for more details.

All table entries can be edited after a double-click in the corresponding field.

4.8. Scan and repair database

The Tools menu has the function “Scan and repair…” database. Selecting this item from the tools menu opens the “Scan database” window shown in Figure 53. In this window the user can determine whether logging information messages are shown while searching the database for missing cross-references.

The button “Save log” starts a file dialog asking for the place where the log file should be stored. The location where to store the log file can be determined before and after scanning the database.

The scanning of the database and rebuilding of cross-reference is executed by clicking on the “Start” button. Alternatively, the window can be closed without scanning the database by clicking the “Cancel” button.
After the database scan is completed the log messages are displayed if this option was selected by checking the option “Log information messages while processing” (Figure 54).

The window is closed by clicking the "✔️ Done" button.

4.9. Export database

The current instance of the CRAFT knowledge base which is used by the CRAFT Editor can be exported from the relational database to a file using the “Export database…” entry from the tools menu. If this entry is selected from the tools menu the CRAFT Editor Export knowledge base dialog is displayed on the screen (Figure 55).
Starting the export by clicking the “Start” button opens a file dialog (Figure 56). In this dialog the user determine the output file.

After termination of the export process a brief status message on the successful export of the knowledge base is shown (Figure 58).
Figure 58  Message about successful export of the knowledge base

All log messages generated in course of exporting the CRAFT knowledge base can be saved to a file. The file can be specified using the file dialog which opens if the “Save log” button is clicked.

If the knowledge base contains reaction rules without a likelihood model and likelihood category the export of the knowledge base fails. A message with details on the error is displayed (Figure 59). The application of a reaction rule without any likelihood information can not be evaluated by the CRAFT Explorer.

Figure 59  Example for an error message during the export of the knowledge base

In addition, the user gets the message that the export failed (Figure 60) after the error message box was terminated by clicking the “OK” button.

Figure 60  Failure notice for the unsuccessful export of the CRAFT knowledge base

4.10.  **Import database**

The CRAFT knowledge base can be imported from a file and saved in the relational database which is currently used. The “Import database...” command from the tools menu opens the CRAFT Editor Import knowledge base window (Figure 61). The import function adds all records to the current knowledge base and might duplicate entries. After pressing the “Start” button the user is asked to specify the file in the dialog box shown in Figure 62. During the import process the progress is displayed (Figure 63). Finally, a message about the successful import of the knowledge base is shown (Figure 64).

The Import knowledge base dialog then can be closed by clicking the “Done” button.
Figure 61  CRAFT Editor Import knowledge base dialog

Figure 62  CRAFT Editor Specify input file name dialog for the knowledge base to be imported into the CRAFT Editor

Figure 63  Display of the progress of the import of the CRAFT knowledge base. a) import of the different database tables. b) dangling of the cross-references.
Figure 64  Message about successful import of the knowledge base
5. CRAFT Annotations

5.1. Introduction

The CRAFT project employs a CML-based format to represent both the chemical structures and the reaction type information. A choice in favor of CML [Chemical Markup Language, P. Murray-Rust World Wide Web Journal, 1997, pp 135-147] was made in spite of other possibilities like:

- Reaction SMILES [www.daylight.com/dayhtml_tutorials/languages/smiles]
- SMARTS [www.daylight.com/dayhtml_tutorials/languages/smarts] or
- SMIRKS [www.daylight.com/dayhtml_tutorials/languages/smirks]

The above alternatives all represent the well-known and widely accepted formats for storing chemical structures and/or reactions. However, their application in the CRAFT project was seen as limited since the available open-source tools and libraries are either of poor quality and do not support all the necessary functionality, or are licensed under incompatible licenses (CRAFT software is licensed under GNU LGPL license [www.gnu.org/licenses/old-licenses/lgpl-2.1.html]). Additionally, the need to provide a graphical editor for the reaction types also made the choice of a CML-based format more favorable since the chemoinformatics platform used for the project – the Chemistry Development Kit (CDK) v 1.0.2 [cdk.sourceforge.net] – already contained the JChemPaint graphical editor that is CML-aware. Thus, the own CML-based format for structure and reaction representations allowed an implementation of a graphical editor tool that can be used to manage the reaction type’s information. Moreover, using such a format as CML that can be easily extended supports the ability to re-employ the developed CML format in other applications.

In order to provide the CRAFT applications with all the required information about the reaction types, an extension of the basic CML format has been implemented, that allows one to annotate chemical structures with various annotations including:

- informative annotations that simply bear a tagging function e.g. to mark a molecule as a structure alert
- query annotations that define various substructure related query features
- transformation annotations describing the changes in a chemical structure in order to yield a predesigned transformation

The extension of the CML format used in the CRAFT project – the CRAFT reaction generation engine annotation format (CRAFT-RGE) implements all the abovementioned features and provides a flexible architecture allowing for its easy extendibility.

5.2. Annotating model

The CRAFT-RGE annotations define the way to describe reaction types and reaction rules and specify the syntax for them. The annotations introduce different elements, like informative items, query flags or transformation rules:

- Informative items include descriptive annotations, e.g. title of a given reaction type or a reaction rule implementation details, global structure flags, etc
- Query annotations that mark certain structure features, e.g. if a bond should be aromatic or if an atom must match a set of elements in the target structure etc
- Transformations that describe the changes made in the structure in the course of reaction, e.g. if a found matching bond should be broken or which predefined
structure fragments will be attached to an atom in the matching target structure after a bond has been broken to form a reaction product.

The CRAFT-RGE annotations currently provide all the features needed to formally describe the substructure queries of all necessary reaction types currently supplied with the CRAFT applications. Additional annotations may become necessary if new reaction types should be added to the CRAFT knowledge base, therefore, the implementation must be flexible enough to enable the changes in the annotation set quite easily.

5.3. Implementation of CRAFT-RGE annotations

The annotation model used in the CRAFT-RGE annotations is based on <scalar> elements supported by the CML standard. In order to distinguish the customized <scalar> elements from the standard ones, a CML convention named rge:annotation is used.

The implementation of reaction annotations consisted of the two major steps:

1. Implementation of the annotation syntax, including:
   - Design of the annotation syntax and grammar
   - Design of parsing mechanisms
   - Implementation of necessary syntax and grammar and corresponding Java classes

2. Implementation of an initial set of annotations, including:
   - Describing the initial annotations using the developed grammar and syntax
   - Implementation of parsing Java classes and mechanisms of employing the annotations in CDK

A short description of the XML schema defining the CRAFT-RGE annotations is given below. For more information about the CRAFT-RGE annotations please refer to the API documentation.

5.3.1. Syntax and grammar definitions

Since the CRAFT-RGE annotations are used within the CML format, it is natural to employ an XML schema definition for the annotations specification itself. To ensure the flexible and error-prone concept of the annotations, the XML schema definition has been developed for the CRAFT-RGE annotations. The designed XML schema definition ensures that the annotations are syntactically correct and correspond to the grammar developed. This makes it much easier to parse the dictionary of the supported annotations and also prevents users from writing their own annotations and reaction types in a wrong way. The XML schema definition is employed when loading the dictionary of supported annotations stored in an XML file. It also makes it easier to write own annotations when employing any XML-validating editor - some of them even auto-complete the elements being typed based on the XML schema information.

The following Figure 65 presents a diagram that roughly reflects the structure of the XML schema definition for a single annotation element used in the CRAFT-RGE annotations.
According to Figure 65, each annotation includes the following attributes and elements:

- **title** – an obligatory attribute defining the name of the annotation
- **dataType** – an obligatory attribute defining the type of the data stored in the annotation value
- **domain** – an obligatory attribute defining the domain where the annotation is meant to be used
- **implementation** – an obligatory attribute defining the name of the Java class that implements the logic of annotation
- **priority** – an obligatory attribute defining the priority of an annotation in the group of annotations
- **minValue** – an optional attribute defining the minimal allowed value that an annotation may have
- **maxValue** – an optional attribute defining the maximal allowed value that an annotation may have
- **label** – an obligatory element containing a string that will be used to denote the annotation element in a GUI application
- **description** – an obligatory element containing the description of the annotation
- **comment** – an optional element containing the comments about the annotation, e.g. the implementation details, help texts etc
- **value** – one or more optional elements that define all the allowed values for an annotation, e.g. a list of bond orders defining single, double, triple and aromatic bonds as annotation values
• **calculator** – an optional element that defines a Java class that shall be used to compute the annotation value, if necessary.

According to this schema, the CRAFT-RGE annotations are described in a dictionary file which is then used by all the CRAFT applications that deal with reaction types. The XML schema also defines three groups of annotations – **informative**, **query** and **transformation** annotations, and each of these groups is described in the annotations dictionary.

The data types used by the annotations include Booleans, numerical (integer and double precision) and string values.

The CRAFT-RGE annotations are split into six different domains – **molecule**, **atom**, **bond**, **reaction**, **reaction-type** and **reaction-rule** – defining the annotations for all these different entities correspondingly. Thus, it is easy to filter out only those annotations that are applicable to a certain chemical object.

The heart of the **annotation** element is its **implementation** attribute value. As defined in the CRAFT-RGE, each annotation provides a ‘sticky’ element that is stored together with the annotated object – e.g. an atom – within the CML description of the atom. This element is honored by the CRAFT applications and is constantly used. However, the annotation itself does not define any logic, e.g. a query annotation XML element does not say anything about how exactly the substructure query is influenced by an annotation. The logic of the annotation is separately represented by a Java class named in the **implementation** attribute.

Such an approach might seem making things too complicated; however, the separation of the logic and of the description of the annotation elements has certain advantages. First of all, the annotations dictionary, once prepared, will not be changed unless a new annotation is needed. Any changes in the logic of the annotation do not influence the dictionary itself. Secondly, the dictionary of the annotations speeds up searching of the specific annotation elements since it can be preloaded at startup time and statically stored in memory. On the other hand, modifications of the dictionary do not necessarily require the changing of the source code of the annotation implementing classes.

The **priority** attribute defines the order in which the annotations are evaluated. This is especially important for the transformation annotations. The highest priority value is 0 (zero), the larger is the number, the lower is the priority. That is, the annotations with the priority 0 zero are processed first. It is not recommended to have two annotations with the same priority value within the same group of annotations.

Both **minValue** and **maxValue** attributes of an annotation element optionally define the boundaries for a value that may be assigned to the annotation. They make sense, e.g. for the annotations describing the number of bonds, valence states, ring membership etc.

The **label** element of an annotation defines the GUI label used to represent the annotation. The **description** and **comment** elements of an annotation both have rather informative character and may be used to comment on the annotation implementation details, i.e. the logic it is implying, syntax rules etc. The difference between the **description** and **comment** elements is that the CRAFT applications use comments for the inline help while descriptions are not visible to the end users.

The **value** element may be repeated several times and is used to limit the annotation to a set of certain values. It is useful, e.g. to define a list of acceptable bond orders in a bond order query annotation. These elements are optional so that if no annotation **value** element is defined for an annotation, it may have any value, including an empty one.

The optional **calculator** element, which must be unique, if present, describes a Java class that is used to compute, e.g. a descriptor value in order to evaluate the logic of an annotation. As an example, a query annotation defining the partial atomic charge may be
named. Its logic could impose that an atom must be treated as matching in a substructure query only if its partial charge is within the given boundaries. To compute the partial charge of a candidate atom, the calculator element should be used to define the calculator implementation.

The annotation implementations define only one feature that is not described by the XML dictionary – the plurality of the annotations. Certain annotations – depending rather on the implementation details than on the formal definition – may or may not be specified multiple times for a chemical object. Therefore, a class that implements the logic of annotation defines if it can be used multiple times for annotating the same chemical object or not.

Certainly there are more details about the CRAFT-RGE annotations syntax and grammar. Please refer to the API documentation for getting more information about them.

5.3.2. Default dictionary of annotations

The CRAFT applications use a default set of the annotations. This set currently supports all the informative, query and transformation annotations that were needed to describe the set of reaction types provided by the UM-BBD database. This section describes the provided annotations and gives some examples on how they can be employed.

Informative annotations

**Structure alert**

The structure alert annotation is used to indicate the structure alert – in other words, the substructure feature that is required to enable a certain reaction type. This annotation can be used once for a reaction type, applies to a molecule and marks the (sub)structure that must be present in the candidate chemical to allow the reaction type to be used. As this is a Boolean annotation, the CRAFT Editor shows a check box for it. A molecule annotated with this annotation is highlighted with blue color in the reaction type editor.

The annotation is global for a molecule and can be set on any bond or atom of this molecule; still it will be recognized for the whole molecule.

**Substructure exception**

The substructure exception annotation is used to indicate a substructure feature that must be absent in a candidate chemical in order to allow a reaction type to be used. Again, this is a Boolean annotation applicable to molecules as a whole. It marks a substructure fragment as an exception. It is required to reflect the fact that a reaction type is only applicable to a chemical structure in the absence of a certain substructure feature.

A reaction type may define multiple substructure exceptions.

The annotation is global for a molecule and can be set on any bond or atom of the molecule; still it will be recognized for the whole molecule.

**Full structure exception**

The full structure exception is necessary to define a complete molecule that should be excluded from a reaction even if it matches the structure alert. This annotation might be specified several times within a reaction type. It is defined for a whole molecule.

The annotation is global for a molecule and can be set on any bond or atom of the molecule; still it will be recognized for the whole molecule.

**Atom-atom mapping number**

The atom-atom mapping annotation is the informative annotation of an atom to indicate the atom pairs that will be connected later on in the course of the transformations. It differs
from the default definition of the atom-atom mapping in a reaction where the mapping
numbers indicate the positions of specific atoms in the reactants and in the products
correspondingly. In the CRAFT-RGE annotation, the atom-atom mapping number must be
assigned to an atom pair that will be connected by a specified bond during the
transformations prescribed.

Figure 66 Atom-atom mapping annotation

As shown in Figure 66, there is an atom-atom mapping number 1 set on one of the
oxygen atoms and on a carbon atom so that the reaction generation engine knows that
these atoms should be connected. The same has to be done for the other oxygen atom
and carbon atom which are involved in this reaction type. The bond type that the atoms
should be connected with is specified by the make bond transformation annotation (see
below).

Distinct reaction name

The distinct reaction name annotation (also sometimes referred to as the
implementation title) is used to name a specific reaction, e.g. Ester hydrolysis. It is a
purely descriptive annotation.

Name of reaction type

The name of reaction type annotation assigns the family name to the group of reaction
types, e.g. it defines a domain of reaction types in the hierarchy of reactions. It is a purely
descriptive annotation.

Query annotations
Query annotations are used to specify the various substructure query features. Like in SMARTS, certain features like the aromaticity of atoms or bonds, the valence state or connectivity of atoms, element lists or functional groups can be specified by means of the query annotations. The current implementation does not yet support all the standard SMARTS features, however, the already implemented features allow quite advanced querying of substructures.

Please note that the CRAFT applications, when doing a substructure matching to detect the reaction types applicable to a candidate chemical, employ the CDK UniversalIsomorphismTester class that has certain limitations (see CDK documentation for more information).

It is important to specify all the explicit hydrogen atoms where they are needed in the structure alert in order to enable certain chemical reactions.

The specifying of a substructure query with the CRAFT-RGE annotations has one peculiarity that has to be taken into account when creating a query. Namely, when a substructure is being annotated in the reaction type editor, if at least one query annotation has been set on an atom or bond, only the annotations are used to judge about matching or not matching when comparing the structures. For example, if on a carbon atom an any aromatic atom annotation is set, then this atom will match any aromatic atom, not only the aromatic carbon atoms. In other words, specifying the query annotations on an atom or a bond should define all the features that have to be met in order to allow a substructure match.

If several query annotations are set on an atom or a bond, they all are logically combined with an AND operator. This means, that in order to hit a substructure match, a candidate atom or bond being compared to the annotated one must fulfill all the query annotations specified.

Query annotations of atoms

When specifying a structure alert for a reaction type, the atoms of the structure alert can be either regular atoms – like carbon, oxygen etc. – or they can be atoms with annotations. In case there are no annotations set on an atom, it is used in the substructure query on as is basis, meaning – exactly as it is specified. The annotated atoms, however, are compared based on the annotation values only, disregarding explicitly specified elements.

Any aromatic atom

This is a singular query annotation that marks an atom in the query substructure as matching any other one if, and only if they both atoms belong to an aromatic system. The element of an atom is not honored. To limit the matches to a specific element, e.g. carbon, also set the matching elements annotation (see below) on the atom.

Any aliphatic atom

This is a singular query annotation that marks an atom in the query substructure as matching any other atom if, and only if they both are not aromatic. This definition is slightly wider that the standard definition of aliphatic compounds as non-aromatic hydrocarbons, but it allows as well specifying, e.g. non-aromatic nitrogen atoms.

If this annotation is the only one set on an atom, this atom will match any aliphatic atom disregarding the elements. To limit the elements, also set the matching annotations on the atom.

Any ring atom

This is a singular query annotation of an atom that matches any atom belonging to a ring system. The number of rings the atom is member of is disregarded. To limit the ring
membership to a certain number of rings, consider the use of the ring membership annotation (see below).

If this is the only one annotation assigned to an atom, the element of the atom is disregarded when doing the substructure match. To limit the elements, also set the matching annotations on the atom.

**Ring membership**

This is a singular query annotation of an atom that matches any atom belonging to a given number of rings in a molecule. The number of rings must be set explicitly. It may be also set to 0 (zero) meaning that an atom must not belong to any ring system. The ring membership annotation value cannot be negative.

If this is the only one annotation assigned to an atom, the element of the atom is disregarded when doing the substructure match. To limit the elements, also set the matching annotations on the atom.

**Valency**

This is a singular query annotation of an atom that matches any atom in the given valent state. The valent state of the atom has to be specified and cannot be negative or equal to zero. Unlike the connectivity annotation, this annotation checks the valent state, i.e. the total sum of orders of the bonds adjacent to an atom.

If this is the only one annotation assigned to an atom, the element of the atom is disregarded when doing the substructure match. To limit the elements, also set the matching annotations on the atom.

**Connectivity**

This is a singular query annotation of an atom that matches any atom with the given connectivity. The connectivity of an atom is evaluated as the count of the bonds adjacent to an atom, not the sum of the bond orders. The connectivity must be set as non negative number.

If this is the only one annotation assigned to an atom, the element of the atom is disregarded when doing the substructure match. To limit the elements, also set the matching annotations on the atom.

**Matching elements**

This is a plural query annotation of an atom. The atom annotated with this annotation will match in a substructure query all the atoms that are given in the list of the annotation values. The size of the list is not limited, and it is enough that the atom in a candidate molecule matches any of the elements in the list of annotation values.

The annotation values for this annotation may be:

- Any real elements symbols, e.g. C, S, O, Cl etc.
- A wildcard symbol * (asterisk) meaning – matching any element
- An element symbol followed by a sequence of plus or minus signs to indicate the charged species, e.g. O- or S++
- A pattern like #<xx><T><nn> where <xx> is the atomic number of an element that should match, <T> is the type of the pattern and <nn> is a numeric value depending on the pattern type. These patterns reflect a SMARTS-like functionality supported by CRAFT-RGE annotations.

Patterns that can be used in the list of matching elements it their turn can be of three different types (the symbols are case-sensitive!):

- H pattern to indicate the number of attached hydrogen atoms
- X pattern to indicate the connectivity of an atom
- \( v \) pattern to indicate the valency of an atom

Here are some examples for the patterns. The pattern \( \#7H2 \) matches a nitrogen atom with two attached hydrogen atoms – also -\( \text{NH}_2 \) – only, while \( \#7H1 \) matches both -\( \text{NH}_2 \) and -\( \text{NH} \). The pattern \( \#7X2 \) matches the nitrogen in nitroso group of 4-nitrosoanilin, but not the amino group, while the pattern \( \#7v3 \) will match both the nitrogen in amino group and in nitroso group (Figure 67). Neither of these patterns, however, will match nitrogen in nitro group -\( \text{NO}_2 \).

![Figure 67 4-Nitrosoaniline](image)

**Matching anything except**

This is a plural query annotation of an atom. The annotated atom will match anything except the elements and patterns given in the list of the annotation values. The number of elements in the list is not limited. The elements in the list may be the same as for the list of matching elements (see above).

This annotation, thus, allows one to specify the list of exceptions in a substructure. Please note, that it makes no sense to set both lists of matching elements and exclusions. It will take much more time to compare both than evaluating only one list, so instead of setting both lists it is better to configure a more comprehensive list either of matching elements or of exclusions.

**Formal charge of atom**

This is a singular query annotation of an atom that matches any atom with the given formal charge. The formal charge should be a positive or negative integer, it may be also set to 0 (zero).

If this is the only one annotation assigned to an atom, the element of the atom is disregarded when doing the substructure match. To limit the elements, also set the matching annotations on the atom.

**Query annotations of bonds**

The bonds in the structure alert again can be either annotated or not. If a bond is not annotated, it is compared to a candidate bond using the bond order only. Alternatively, if there are some annotations set, only they decide whether a bond matches the candidate or not, explicit bond order is disregarded in this case.

**Aromatic bond**

This is a singular query annotation of a bond. The annotated bond in a query substructure will match any candidate bond that is aromatic.
Note: despite CDK v.1.0.2 actually has a separate bond order value for aromatic bonds - CDKConstants.BONDORDER_AROMATIC - it is practically unused even internally in CDK own routines. Instead, an aromaticity flag is set over the bonds that are found to be in an aromatic system. Therefore, when specifying a query bond that should be aromatic, there is no need – and in fact, it is even not recommended – to draw the query substructure as an aromatic one. It works better when the query substructure defines single bonds with aromatic bond annotation set on them (see examples below).

Please note that if this is the only annotation set for a bond, the actual bond order is disregarded when matching a substructure and the annotated bond will match any bond having the aromatic flag set. Setting additionally a list of matching bond types does not make sense for this annotation since the aromatic bonds in fact have the same bond order even if they are depicted using Kekule representation.

**Aliphatic bond**

This is a singular query annotation of a bond. The annotated bond in a query substructure will match any candidate bond that is aliphatic. Internally, all the bonds that are not aromatic are treated as aliphatic, thus, this annotation is simply neglecting aromatic bonds.

Please note that if this is the only annotation set for a bond, the actual bond order is disregarded when matching a substructure. The annotated bond will match any aliphatic bond. To limit the matches, please set also the list of matching bond types (see below).

**Ring bond**

This is a singular query annotation of a bond. The annotated bond in a query substructure will match any candidate bond that belongs to a ring system. The actual bond order is disregarded, if this is the only annotation set. To limit the matches, please set also the list of matching bond types (see below).

**Matching bond types**

This is a plural query annotation of a bond. The list of values for this annotation allows for the specifying the bond types (bond orders) that should match the annotated bond. The number of elements in the list is not limited, but there are only four allowed values that can be combined in any order:

- Single bonds
- Double bonds
- Triple bonds
- Any bonds – denoted with ~ wildcard symbol

It is allowed to combine these values in any combination, despite adding “Any bond [~]” value to the list will make the bond matching any candidate bond as well. The actually depicted bond order is disregarded when matching a substructure.

**Transformation annotations**

The transformation annotations describe the chemical modifications that should be applied to the substructure found by a substructure query so that the designated reaction logic is executed. It is important to emphasize that the specification of the structure alert for a reaction type only defines the substructure feature needed to undergo the given reaction. To complete the reaction itself, it might be necessary to provide some additional molecules and/or fragments that will be then attached to the modified structure of a candidate chemical.
**Break bond**

This is a singular transformation annotation of a bond. The annotated bond will be broken during the transformation. This annotation has the highest priority, thus, the bonds to be broken will be processed always first.

Please note that this annotation can only be used if set on bonds belonging to the structure that matches the structure alert. Specifying this annotation on bonds belonging to other disconnected fragments – not to the structure alert – will have no effect!

**Delete atom**

This is a singular transformation annotation of an atom. The annotated atom will be deleted from the structure during the transformation. This transformation is the second one after the breaking bonds, i.e. the atoms that should be deleted are removed after all the bonds to be broken are broken.

Please note that this annotation can only be used if set on atoms belonging to the structure that matches the structure alert. Specifying this annotation on atoms belonging to other disconnected fragments – not to the structure alert – will have no effect!

**Delete fragment**

This is a singular transformation annotation of an atom. The fragment of the candidate structure that contains the atom with this annotation will be completely deleted in the course of the transformation. This annotation is a convenient mean to remove several bonds and atoms from the reacting molecule. It has the third priority in the default dictionary.

Please make sure that the fragment that should be deleted in the reaction is disconnected from the rest of the molecule, i.e. all the bonds connecting the fragment to be removed are annotated with **break bond** annotation. Otherwise, the entire molecule will be deleted!

Please note that this annotation can only be used if set on atoms belonging to the structure that matches the structure alert. Specifying this annotation on atoms belonging to other disconnected fragments – not to the structure alert – will have no effect!

**Make new bond of type**

This is a singular transformation annotation of an atom. Together with the atom-atom mapping number annotation it defines the creation of new bonds of given type.

The annotation value can be one from the list of values for matching bond annotations: namely **Single**, **Double** or **Triple**. The new bond with **aromatic** order is not present in the list – see the note to the aromatic bond annotation.

To denote a new bond to be created, it is necessary to mark two atoms – does not matter if they are in the same structure fragment or not – with the same atom-atom mapping number, and to set on either of these two atoms the new bond order. This annotation must be set only on one of the two atoms with the same mapping number. Otherwise an error message will be shown by the editor. Please refer to the examples below to get more details on this annotation.

This annotation can be set both on atoms belonging to the structure that matches the structure alert or on atoms belonging to any disconnected fragments.

**Replace with bond type**

This is a singular transformation annotation of a bond. The annotated bond is replaced with a new bond of the specified type.
The annotation value can be one from the list of values for matching bond annotations: namely **Single**, **Double** or **Triple**. The new bond with **aromatic** order is not present in the list – see the note to the aromatic bond annotation.

Please note that this annotation can only be used if set on bonds belonging to the structure that matches the structure alert. Specifying this annotation on bonds belonging to other disconnected fragments – not to the structure alert – will have no effect!

**Set new atomic charge**

This is a singular transformation annotation of an atom. The annotated atom will receive a new formal charge specified in the annotation value after the transformation.

Please note that this annotation can only be used if set on atoms belonging to the structure that matches the structure alert. Specifying this annotation on atoms belonging to other disconnected fragments – not to the structure alert – will have no effect!

**Break conjugated system**

This is singular transformation annotation of a bond. The annotated bond will be marked as not belonging to a conjugated or aromatic system, i.e. the aromaticity or conjugation flags will be removed from the bond.

This annotation is necessary to indicate that a certain bond will loose its aromatic or pi-conjugated character after a transformation. The need in such an annotation is again caused by the limitations of used in CDK valence bond representation model.

Please note that this annotation can only be used if set on bonds belonging to the structure that matches the structure alert. Specifying this annotation on bonds belonging to other disconnected fragments – not to the structure alert – will have no effect!

**Reconfigure conjugated systems**

This is a singular transformation annotation of a bond. Being set on any bond in the molecule, it will make the application to try to rebuild completely the entire set of all the pi-systems in it. This annotation has the priority value of 999999999 to ensure that the corresponding transformation is always done at last.

It is only needed to specify this transformation once on a bond in the reacting molecule to rebuild all the pi-systems in it.

This transformation annotation is required in certain cases when a molecule's pi-system gets destroyed and/or changed in the course of chemical reaction. Please refer to the examples below for more details on this transformation.

**5.3.3. Hints and examples**

The following section contains some examples and hints on the use of certain annotation whose meaning might not be obvious from the above description only.

**Querying the aromatic rings**

To illustrate the aromaticity annotation, the following task is chosen: implement the following structure alert (Figure 68):
Figure 68   Biotransformation rule bt0005 “vic-unsubstituted Aromatic $\rightarrow$ vic-Dihydroxyaromatic”

In this example, the aromatic rings are meant not to be fused at 2,3 position or containing heteroatoms [see umbbd.msi.umn.edu/servlets/rule.jsp?rule=bt0005 for more details]. The position depicted as (H1) must have explicit hydrogen atoms to allow the reaction.

To specify the structure alert for this reaction type, the following structure should be drawn and the following annotations are to be set (Figure 69).

Figure 69   Annotating the structure alert for the reaction type show on Figure 68

Note that in the rule description it is explicitly said that the reaction happens only at aromatic rings containing only carbon atoms and which are not fused at carbon atoms 2 and 3. If there were no such limitations and also the reaction type would not be limited to the carbon aromatic rings only, one could express such a pattern as shown on the Figure 70:

Figure 70   Annotating the structure alert for the reaction type shown in Figure 68 without taking into account the limitation about fused rings and hetero atoms
Querying carboxylic acids and anions

To query a carboxylic acid in both neutral and charged (anionic) forms, the following structure alert could be annotated (Figure 71)

![Figure 71 Annotating carboxylic acid in both neutral and anionic forms](image)

Creation of new bonds

To create a new bond, there are several annotations to be set:

- Annotate an atom in the structure alert with a unique atom-atom mapping number. The number must be unique within the current reaction type definition
- Annotate an atom in a fragment to be connected to the structure alert with the same atom-atom mapping number
- Annotate the atom in the fragment or the atom in the structure alert with the “make new bond of type” annotation indicating the type of the bond being created

Several new bonds

The following figure (Figure 72) illustrates the creation of two single bonds when oxidizing the benzene ring according to the UM-BBD rule bt0005 [see the web site of UM-BBD umbbd.msi.umn.edu/servlets/rule.jsp?rule=bt0005 for more details].
Several new bonds to the same atom

Sometimes it is necessary to create several bonds to one and the same atom. This is also possible as shown on the Figure 73.

Rebuilding a conjugated system

In certain cases it is necessary to indicate that the pi-system of the molecule is changing in the course of the reaction. Because of the valence bond representation used in CDK, it is sometimes impossible to unambiguously specify the structure alert that should match
an aromatic system that then becomes destroyed or needs to be rebuilt. As an example, look at the UM-BBD rule bt0367 vic-unsubstituted Aromatic → vic-Dihydroxyaromatic [http://umbbd.msi.umn.edu/servlets/rule.jsp?rule=bt0367] (Figure 74):

![Figure 74](image)

**Figure 74** The UM-BBD rule bt0367

This rule implies that an aromatic system will be destroyed after the reaction is complete. Thus it requires the rebuilding of the aromatic system in the molecule. Simply specifying the change of bonds from single to double and vice versa will not work: in one of the resonance forms of the benzene ring this will correctly replace the bonds while in the other the replacement simply will not take place (see Figure 75)

As shown on Figure 75, the aromatic ring annotated as the structure alert will match both resonance forms of the Kekule representation of benzene. Accordingly, for this example the structure alert uses **replace bond** annotations to construct the new molecule.

![Figure 75](image)

**Figure 75** Switching of the bonds in an aromatic ring does not always work

As can be seen, depending on in which initial resonance form the benzene ring is input, the results of the transformation will be different. If the benzene ring is given in the form 1 as shown on Figure 75, the transformation **replace bond with single** will correctly replace the matched double bond and the resulting structure will be as expected. However, if the input structure has resonance form 2, the matched bond is already single, therefore the **replace bond** transformation will not be doing anything and the resulting molecule will be invalid.

This is common limitation of the valence bond based representation of pi-systems. To overcome it, all the resonance structures of the candidate molecules have to be
enumerated to make sure the structure alert query will match. Secondly, the reconstructing of the pi-system after transformation is necessary to re-evaluate the bonds orders and their aromaticity after the transformation is completed – i.e. all the bonds needed to be broken are disconnected, all the bonds to be changed are already replaced, and all the new bonds are created. To do this, both break conjugated system and reconfigure conjugated systems annotations are to be used in conjunction.

The break conjugated system annotation should be set on the bond that should be excluded from further conjugation. In our example, this would be the bond previously annotated with replace bond annotation on Figure 75. Then, any bond in the structure alert should also be annotated with reconfigure conjugated systems annotation; it can be the same bond again. Finally, no other annotations of type replace bond should be assigned.

The resulting annotation of the transformation looks like it is shown on Figure 76.

![Figure 76 Annotating transformations for the UM-BBD rule bt0367](image)
6. Troubleshooting

6.1. Memory problems

The application may fail with an Out of memory error. If this happens, try to start the CRAFT Editor manually requesting more memory for the JAVA virtual machine as described in the following part.


The CRAFT Editor application itself is stored in the executable JAR file `mn-kbe.jar` located in the installation directory. The binary executable `CRAFT.Editor-v.1.0.exe` is simply a launcher application that starts the Java virtual machine and loads the CRAFT Editor application into it. The launcher application tries to configure the Java runtime so that it requires at least 512MB of the operating memory (RAM) for the operation. If a PC is equipped with a greater amount of RAM, it might be useful to not to use the executable launcher but to call the CRAFT Explorer directly with a customized batch file. An example content of such a file might look like that:

```bash
@echo off
javaw.exe -Xms1024M -Xmx1024M -splash:kbe-splash.png -jar mn-kbe.jar
```

Note that the location of the `javaw.exe` file – the Java runtime executable on Windows® platforms – must be either given explicitly or be indicated in the system PATH variable. The highlighted value `1024M` indicates a memory amount of 1GB that the Java runtime can use. Please make sure that these numbers match to the really available memory of a PC.

To use such a custom launcher, save the text above in a batch file, e.g. `craft-kbe.bat` in the same directory where the JAR file `mn-kbe.jar` is located. Now this batch file can be used to run the CRAFT Editor application.

6.3. Log file location

The CRAFT Editor application uses a flexible logging system to keep track about the events happening during the run-time. The initial releases of the application still may produce very detailed logs (so-called debug mode logs) while the final version will only keep track of the critical errors and warnings.

Either way, the log file of the CRAFT Editor is quite useful in case the application behaves unexpectedly or there are error messages shown, or the application crashes etc.

In such cases, please inspect the log file which can be found in the temporary folder under the name `craft.log`. In order the location of temporary folder is not clear, simply search for the file named `craft.log` using the Windows Explorer or other available tools.

6.4. Reporting an error

In case the CRAFT Editor application crashes or there is a reproducible error, please do the following before reporting the error:
• Close the CRAFT Editor application, if it is still running
• Locate and delete the log file *craft.log*
• Start the CRAFT Editor application
• Reproduce the situation when the error occurs
• Close the application if it is still running
• Locate again the log file *craft.log* and attach it to the error report
• Describe the error, the circumstances how it occurs and how to reproduce it
• If possible, attach also the chemical structures that were used
• If a modified CRAFT knowledge base has been used for the run, please prepare the database export file with the CRAFT Editor and attach it to the error report