# Use Case

Memolist is a list of objects the user wants to remember. He can select objects in data table and drawing by clicking on objects in the memolist. So the memolist is like a fixed selection controlled by the user. Objects can be added and removed by the user at any time.

# User Interface

## Memolist pane

The memolist should be implemented as a dockable pane. There are two reasons why a pane is more useful as a window.

* First, the memolist is probably open for most of the time. The user needs a position where the memolist doesn’t overlap important interface parts and where it can be accessed rapidly.
* Second, the memolist is HCV specific. Each hcv has its own memolist. That’s different compared to the Search dialog, which works on all open HCVs.

Because it’s dockable and HCV specific it has to be re-initialised when drawing is changed/opened/closed. Just like the modules pane.



Abbildung 1: Memolist pane

Memolist pane contains a tree of objects. The tree should look like the search result tree. It has the same logic like search result tree:

* When a module is deselected by the module configuration, the corresponding parts are greyed out in the tree
* Objects are grouped in categories
* One click selects the object in memolist
* Double click selects the object in data table and drawing
* Empty categories need to be hidden, to reduce the length of the list. This can also be applied to the search result tree.

There are also some differences to between memolist and search result:

* checkbox who helps the user to keep track of his work process
* textfield for user comments
* Multiselection is enabled

Context menu in memolist:

* “Remove“: removes the current selected objects, this should also be possible by pressing “Del”-key. Together with the multiselection several objects can be removed by this feature.
* “Remove all”: removes all objects from memolist
* “Uncheck all”: deactivates all checkboxes
* “Check all”: activates all checkboxes

## Changes in data table

There are also some extensions to the data table context menu:

* “Add selected to memolist”: add the current selected object (a single object) to memolist
* “Add highlighted to memolist”: add all currently highlighted objects (multiple objects) to memolist, including the selected object. This menu item needs to be separated from all others, e.g. by an menu separator, to clarify that it concerns not only the currently selected object.

## Changes in ribbon menu

We need a new group “Memolist” in the edit menu. Important buttons are “Export” and “Import”. The buttons “Remove all”, “Check all” and “Uncheck all” trigger the same functions like the context menu in the memolist. The functions are represented in the ribbon because they don’t need an object. They are applied to all objects in the memolist.



Abbildung 2: Memolist ribbon group

The “Export“ button exports the current memolist to a file in xml. “Import” imports such an xml file. Together with the object ids, the checkbox status and the user comment needs to be exported. The file ending could be “mmlst”.

# Persistence

The memolist must be persistence. In general it should behave similar to the redlining.

This means:

* When Memolist is changed (objects added/removed, checkbox changed, comment changed), hcv should be marked as dirty (save enabled)
* When hcv is saved, the current memolist is saved within the hcv, together with all checkbox status and the user comments. Just like the redlining.
* Memolist can be exported to an xml file (see Changes in ribbon menu)