Knowledge base version and release management

Knowledge based system version management

This document is a guideline for Knowledge Engineers on development and maintenance of kernel protected knowledge bases. It addresses the most important issues concerning version and release management and provides advice on file and knowledge base organization. It is assuming you use Quaestor version 2.46.2 or later.

Because specific organization of your ICT infrastructure will vary, we advise also to discuss these issues with the responsible persons within your organization.

1 Backup functionality

Although we do our utmost avoiding Quaestor to crash, disaster can strike (also due to other reasons than Quaestor). To avoid loss of information, Quaestor is able to save backups of knowledge bases and projects at regular intervals (see also Backup File).

Moreover, and very important as a tool for developers, you can save these backups with a time stamp, enabling several versions of our backups to be stored. This does not only save time when Quaestor or your computer crashes, it also enables you to go back to a previous version of your development version.

You can set the intervals and the time stamp options in Tools>Options>Files.

2 Knowledge base version management

A Quaestor knowledge base is a binary file, and as such all aspects as facilitated by the MS Windows file management are available. This means that the file can be read and write protected, be added to a backup procedures, etc.

As it is common practice to have a development area and a production area, we will only mention the fact that it is important to have a good separation between your development area and your production area in order not to overwrite a released knowledge base with a development version.
In Quaestor you are able to work with personal profiles (please read about startup behavior for some specific start up behavior and default setting issues). These profiles enable a knowledge engineer to efficiently work on different knowledge bases in different area's requiring different Quaestor directory settings by switch between these profiles.

When a good separation between development and production is guaranteed, there are several ways to keep track of versions of a knowledge base:

2.1 File name and date

The most straight forward way to keep track of knowledge base versions is by means of their file name and modification date.

However, please realize that that Quaestor will use the file name to associate a project file to the accompanying knowledge base. Therefore, changing file names between versions of knowledge bases is a good approach for major releases (in which it might make sense to break the association between an existing project file and this new knowledge base), but for minor releases this is not convenient.

It should be noted that it is always possible to open an old project with the new knowledge base. When no frames are deleted (we will come to that in the next chapter) the project will also work in the new knowledge base and after saving the project, it will start up with the new association.

2.2 Information provided by Quaestor

After opening a knowledge base, in the "About Quaestor" window Quaestor and knowledge base version information can be found.

You will find information on your current Quaestor version (see http://qnowledge.nl/pub/version number convention.htm), your license information and user level.

Furthermore you can see information on the latest modification date and time, who has done this modification to a knowledge base and with which Quaestor version this was done.

2.3 QKnowledgebaseVersion parameter

In addition to this generic information, as a knowledge engineer you can actively manage versions by means of knowledge in the knowledge base.

Read in QKnowledgebaseVersion for more detail.

2.4 General remark

Please note that Quaestor will automatically store the following information as part of a solution:

@CREATEDON:Date\Time
@CREATEDBY:LoginName
@SOLUTIONTYPE:SolutionType
@CREATEDINDOMAIN:Domain

In which Date is his/her system date and Time is his/her system time, LoginName is the Windows login name of the user, SolutionType can be STANDARD, SCENARIO, TAXONOMY and Domain is the domain, in most cases the name of the knowledge base.

3 Knowledge bases development

Below several important knowledge base development issues are discussed.

3.1 Knowledge base protection

See Knowledge base protection for more detailed info. It is important to know that by default knowledge bases are protected against deleting frames

3.2 Deleting knowledge

When you have protected knowledge base, the password is required to open the knowledge base in KE mode (a DE ore EU will not see any password dialog).

In KE mode you can make any modification you want with one exception: you cannot delete frames without removing the delete protection.

The reason is that in Quaestor all knowledge is represented by frames with a frame number. Thus, every parameter, relation, constraint, function is stored in a frame. In the present architecture (based on a long history of Quaestor use and development), Quaestor will renumber the frames the moment you delete one of them. This is no problem as long as you did not use the knowledge in solutions. However, the moment you have made projects based on the knowledge base, removing frames in the knowledge base will result into corrupt project because the relations between frame numbers in your solution will not correspond with the actual content of these frames.

In Quaestor we have made checks in order for you to still load old projects with the modified knowledge base. And most data will still be available in the project. However, the solutions might be broken and have to be recalculated. Realize that, when relations are removed from the knowledge based that are used in a solution, this would be the case anyway.

So deleting of frames is only possible when you actively remove the delete protection through the File menu item "Allow deleting frames". Furthermore, you should only delete frames you have just added and you are sure of that are not yet used in any project.

As a result of this restriction, you might end up with a lot of parameters, relations, constraints, etc. that are obsolete. This is not necessary. By simply reusing the obsolete frames (parameters, relations and constraints) you can simply virtually remove these parameters. You can rename parameters into new names and you can edit and save relations and constraint to fit your needs. Without going into too much detail, there is a small recycling restriction: you cannot change string parameters into value parameters and vice versa.

3. Adding and modifying knowledge

While adding and modifying knowledge it is important to document its reference. In the reference and data slots of the Frame Viewer you can add this content. For a parameter we advise at lease to provide its dimension and one small description, ideally followed by some more back ground text (the first line, until the first carriage return, can be presented as identification in the Workbase). See also Documentation of knowledge in the Wiki.

Please note that the same can be done for relations. In this case the reference slot can be accessed by typing in the second text window of the Frame Viewer below the “-X-“.

When you have modified data, you can document the modifications in this same area. When you want to keep information on modifications separate from the reference text, you can also use the data slot. However, do not use full Quaestor syntax and the @ symbol as this can be recognized as syntax in the data slot.

While adding and modifying relations and constraints, Quaestor will add change log information automatically. In the data slot of relations and constraints, Quaestor will provide to lines:

@ENGINEER:LoginName
@LASTCHANGED:Date at Time

In which LoginName is the Windows login name of the Knowledge Engineer, Date is his/her system date and Time is his/her system time.

You can search on these attributes in using the most right select box in the knowledge browser and select under the filter options for relations or constraint “Input/changed after”. This will initiate an input box where you can provide a date resulting in a filter showing all modifications after this provided date identified by @LASTCHANGES.

3.4 Comparing kernels

When you have several version of a (kernel) knowledge base, you are able to compare knowledge base versions by opening both knowledge bases and choosing Tools>Compare Kernel KB’s. (Please note this is only enabled when you have opened two kernel knowledge bases...).

After you have selected this option, for each knowledge base you will only see the frames in that knowledge base containing differences compared to the other knowledge base. Note that this can be any difference, so also only the @ENGINEER or @LASTCHANGED attributes. The Web Browser should be automatically opened to report the specific differences (if not open the Web Browser with View>Web Browser). The information identified as Source is always the knowledge base that is not being in focus at the moment.

4 Proposed release procedure

Based on our experience and the provided information above, we advise the following release procedure:

1. Check whether you have no red crosses and red relations or constraints;
2. Make sure you have modified your version number(s);
3. Be certain your modifications are functioning properly. For this purpose use a known benchmark and re-run this benchmark checking all results;
4. When this validation is satisfied, open (a copy of) an old project and start an old solution to check for possible compatibility problems;
5. When this is satisfied, write a release document to inform your users about the changes and what they may and may not expect. Optionally add these notes to your knowledge base before releasing;
6. Distribute the knowledge base with all documentation to the appropriate person(s) (ICT department etc.) and archive the released knowledge base version with documentation and a copy of your Quaestor executable in a separate Release folder on your development area.

When there are any additional steps or conflicts with the release procedures within your organization, please discuss this with the responsible persons. The above 6 steps are advise only.