Testing Quaestor and KB-Systems

Background

For good global information about software testing, please visit [Software testing on Wikipedia.org].

To summarise, in general testing can be done on the following levels:

1. **Unit testing**. Testing the minimal software component, or module. Each unit (basic component) is tested to verify that the design for the unit is correctly implemented;
2. **Integration testing**. This exposes defects in the interfaces and interaction between components;
3. **System testing**. This tests a completely integrated system to verify it meets its requirements;
4. **System integration testing**. This verifies that a system is integrated to any external or third party systems defined in the system requirements.

Before shipping the final version of software, *alpha* and *beta* testing is done:

1. **Alpha testing** is simulated or actual operational testing.
2. **Beta testing** comes after alpha testing. Versions of the software, known as beta versions, are released to a limited audience. The software is released to groups of people so that further testing can ensure the product has few faults or bugs. Sometimes, beta versions are made available to the open public to increase the feedback field to a maximal number of future users.

Finally, **acceptance testing** can be conducted by the users, customers, or clients to validate whether or not to accept the product. Acceptance testing is usually performed as part of the hand-off process between any two phases of development.

If we project this generic procedure on a Quaestor knowledge based system, we have to take into account some aspects in the architecture of the total system.

A Quaestor knowledge based system is based on three parts:

The knowledge base + project + and Quaestor

Depending on the position you take within the components of the total knowledge based system, the procedure can be used for the total knowledge based system and for each part individually. Please realize that, after delivery, the responsibility for the knowledge base (and projects based on it) becomes the responsibility of the knowledge engineer. In other words, for testing of the total knowledge based system, there is a separation in testing done by Qnowledge prior to release of a new Quaestor version and testing done by the knowledge engineer for release of a specific knowledge based system within an organization.

As we are involved in the majority of knowledge base developments. We are able to perform the total test procedure for the majority of the operational knowledge based systems. And as such, we also test the majority of use cases related to System integration, Alpha and Beta testing.

Because we recognize the need for organizations to perform alpha and beta testing on their knowledge based system themselves, below we have provided the generic description of our test procedure, followed by tools and advice to adopted for a procedure within your organisation.

Testing Quaestor

We have divided our testing procedure into three levels:

1. **Unit testing**: this is the same as the generic procedure above and closest to the source itself; testing on the level of modules, classes, sub-routines and functions;
2. **Sub-systems or general functionality testing**: this is the same as the integration and system testing of the generic procedure above. We test the main components of Quaestor;
3. **Global knowledge system testing**: this is the same as the system integration testing and alpha testing of the generic procedure above; testing of the total system, including the use of realistic and complex knowledge bases and use cases.

From the generic procedure perspective, the result is a *Beta* version. When we did not test your knowledge based system as part of our test procedure, this might be correct for the total knowledge based system. However, from the Quaestor (Qnowledge) perspective this is a final release version for Quaestor.

Unit testing

Unit testing is part of the software development environment and can only be carried out by the Qnowledge software department.

Sub-systems or general functionality testing

For Quaestor we recognize the following sub-systems:

1. The modeler/inference engine: creating of models, reasoning with the knowledge;
2. The solver: for determination of iterative problems;
3. Interpreter: interpretation of expressions;
4. Data management: loading and saving of projects, working with the Quaestor project file database;

5. TeLiTab/Object generation, manipulation and management;
6. Knowledge engineering;
7. File operations: loading and saving of file types;
8. GUI behavior;

To test these parts we have developed several tools.

A. For classic applications and scenario applications we have developed the following:

1. A knowledge base **qnowledge.qkb** containing most of the intrinsic functions;
2. A project with the version number of the release it corresponds to. This project contains solutions which can be repeated in the new release to test most of the above aspects;
3. A document with a procedure to assist testing the different aspect above;

Note that, although related to classic and scenario applications, this test is relevant for all knowledge based systems.

B. For Taxonomy applications we have developed some additional tools:

1. A knowledge base **qnowledgeConfigurator.qkb** containing all important Taxonomy related aspects;
2. A project with the version number of the release it corresponds to. This project contains solutions which can be repeated in the new release to test most of the above aspects;
3. A document with a procedure to assist testing the different aspect above;

When both tools are used and results are compared and equal to the original project the of the sub-systems is passed.

Go to testing tools page to download the above knowledge bases, projects and documents.

Global knowledge system testing

This final testing is done by means of about 15 operational knowledge based systems. We test the systems as delivered or on the basis of the last version we have received from the responsible knowledge engineer.

It is clear this is proprietary information that cannot be shared outside Qnowledge.

A good alternative (within an organization) is to test all the knowledge bases within the organization. To do this properly, you should test the new Quaestor release with knowledge bases that worked properly in the previous release. You can basically copy the above approach with a validation projects and test knowledge bases for your own system(s).

Testing the knowledge based system

Testing of the knowledge based system after testing Quaestor basically means testing the modified knowledge base and compare behavior and results with the previous knowledge base and the validation project.

If any strange behavior or result is discovered you can return to your previous knowledge base an check whether changes are due to modifications in the knowledge base or the Quaestorprogram (although you should already have tested this in the Global knowledge system testing).