# Use of external or satellite programs

## External programs or 'Satellites'

In general Quaestor is able to "see" satellite programs as objects / parameters, providing values to the calculation process. For every process there is a generic approach with respect to the file and data management. The way Quaestor is actually communicating with the external program depends on the program itself.

**File and data management for external programs**

Quaestor has two concerns while using programs:

1. How to find the external program;
2. How this program should find its own additional files;

By default Quaestor expects programs in the Applic directory as specified in the standard directory structure. As working directory Quaestor will set the project directory as specified in the standard directory structure. As Knowledge Engineer you are able to specify other directories to find files from withing Quaestor. However, you cannot manipulate the way the external project will find its files. All the required files should always be available in the working directory set by Quaestor.

By means of a special @COPYFIRST attribute for the GET$ function you are able to copy any file prior to running a program. In this way you can make sure any additional data files or dll's are copied to the working directory to be find by the external process.

**Program interaction**

We distinguish the following program interaction:

1. Command-line programs using standardized input and output file names (easiest implementation);
2. Command-line programs requiring specific additional arguments;
3. Programs included as Quaestor functions (Word: WINWORD$(), Excel: EXCEL#() );
4. Scriptable programs with an application programming interface (using the MACRO() function: such as Visio, Rhinoceros, AutoCAD, MatLab);

Note that, with some effort, most of the programs can be integrated with Quaestor.

Please realize however that a connection itself is one thing. What type of information a program requires and is providing, and how intelligent this program is, will determine its usability in a process (or the effort to make it work) to a far greater extent. A smart program can be incorporated with very little effort while a dumb program with equal functionality may require a lot of pre- and post-processing in Quaestor in order to create input and/or get results.

## 1 Command-line programs using standardized input and output file names

These are the most straight forward programs. Usually these programs expect the input file to have a standard name and to be at the location in the active directory (the working directory, which is actively managed by Quaestor) and the program generates the output file with standard name in this active directory.

In Quaestor the following steps are required:

1. use TEMPLATE$() to create the input as specified by the program based on available parameters in your knowledge base;
2. use PUT$() to write it as the specified standard file name and include the relation with the TEMPLATE$() to create the input;
3. use GET$() to read the standard output file and include the specify the program to run and the relation with the PUT$() two write the input file first;
4. use PARSE#() to parse the output file you have written by means of the GET$() function, to get your required resulting data set.

You might not yet realize, but by the fact that parameters associated to the parse script, are associated to the GET file, which is associated to the program and the put action, which is associated to the template for the input, you have connected the input to the output through the program.

Quaestor manages the setting of the working directory and the execution of the program if required (hence, Quaestor will only run the program when the input changes... which makes it highly efficient!)

## 2 Command-line programs requiring specific additional arguments

These programs are already a little more complex, although this is mainly related to getting them executed and not to the input and output actions.

Most of the time these types of programs are executed:

- with switches to define the type of calculations by the program;
- a specification of the input file name;
- optional specification of the desired output file name;
- ...

In Quaestor, the definition of the input file is equal to the first method. However, for execution it might be interesting to create a batch file in addition to the input file. In the batch file you write the specific command line options and in the GET$() relation you do not specify the program itself as process to be executed, but the batch file (which executes the program).

The whole thing still works in the same way but now you are able to parameterize complex batch files enabling all kind of specific actions, such as calling the program from a specific working directory, executing several programs after each other in order to get the result, etc. etc...

## 3 Programs included as Quaestor functions

See the WINWORD$() and EXCEL#() functions for further details.

## 4 Scriptable programs with an application programming interface

See the MACRO() function for further details.