# Glossary

| Term | Meaning |
|---|---|
| **Class** | A means for organizing frames for the user, a container of frames. Not to be confused with a class in object oriented computer programming. The Quaestor core does not use the classes at all. A container can contain a container, can contain.... |
| **Constraint** | A defined limitation that determines if a relation can be used or not. |
| **Dataset** | |
| **Domain Expert** | A Domain Expert is an End User. In addition, he/she can generate new models based on existing knowledge bases. A Domain Expert cannot modify the knowledge in a knowledge base. |
| **End User** | An End User is a user that uses existing solutions with new data. An End User cannot generate new models based on an existing knowledge base. Also, this user cannot modify the knowledge base. This type of user is comparable to someone that uses an existing spreadsheet for computations with new data. |
| **Entity** | An Entity is a generic description of an Object (e.g. a ship structure) or a Process (e.g. the design of a propeller or a seakeeping analysis). |
| **Expression** | An Expression is a combination of operators with parameters, constants and functions, to compute something. |
| **Frame** | An item of knowledge, which can be a parameter, a relation or a constraint. |
| **KCMA** | Knowledge-based Computational Model Assembling: the creation of a model based on a network of knowledge components. From relations, applications, data etc. a model is created in collaboration by system and user, to compute the requested Top Goal. |
| **Knowledge** | Technical knowledge is all information that can be captured in data, parameters and relations (formulas, equations, procedures, applications), their properties and networks built from these concepts. |
| **Knowledge Base** | |
| **Knowledge Engineer** | A Knowledge Engineer is a Domain Expert. In addition, he/she can create, expand and modify knowledge bases. |
| **Object** | An object in Quaestor consists of a collection of Parameters or 'value containers'. |
| **Parameter** | Value container. Knowledge is created by creating relations between parameters. |
| **Process** | |
| **Project** | A collection of solutions. |
| **Quaestor** | ***[Q]uaestor*** is a computer program that allows a user in a simple way, to enter relations and to couple existing data, applications and databases. The inference engine then combines, in an intelligent way, the collections of relations, applications, data and databases to solutions for a given problem.This is an interactive process, where Quaestor searches suitable alternatives and proposes these to the user. Depending on the choices a user makes, the result is a model to solve the problem at hand. The finally generated model can be used over and over again with new data, but fixed knowledge. |
| **Reference** | Extra information about a parameter. Not to be confused with references in programming languages. |
| **Relation** | A means of defining the value of a parameter by an expression (parameter = expression) |
| **Solution** | A solution is the model that is generated as a result of the dialog between system and user. In the Knowledge Base, one or more Top Goals can be selected. After that, the user starts the computation. By entering values on Quaestor's request and by accepting or rejecting relations, a model is generated that can compute the Top Goal(s).This model can then be used over and over again with new input data. A solution can be regarded as programmed knowledge. |
| **Taxonomy** | A Taxonomy is a hierarchical workflow. It consists of Entities, each representing an Object that contains its private data, goals and model fragments and its sequence represents a work flow. |
| **Telitab** | **Te**xt-**Li**st-**Tab**le. A data structure type used inside Quaestor. |
| **Top Goal** | A Top Goal is a parameter for which a (range of) value(s) must be computed. |

| | |
|---|---|
| **Value** | A value can be (a vector of) numerical, nominal (string/document), binary or Object. |