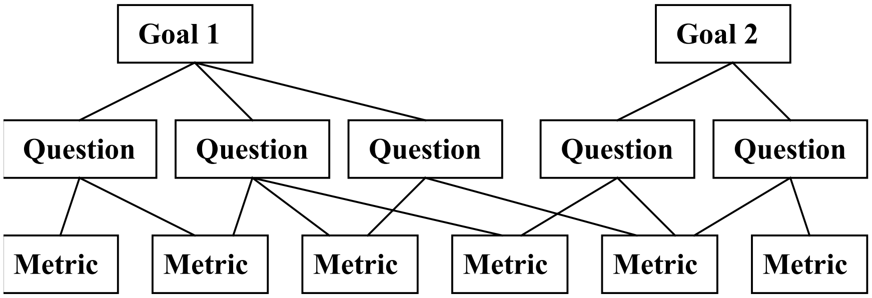
Software Measurement Exercise

To help you develop a set of metrics to collect for you project, you may it useful to carry out a GQM analysis.

GQM (Goals-Questions-Metrics) is a technique for identifying suitable measurements to collect during a software development project. It is based on the observation that data collection for its own sake is usually a waste of time and effort. Hence, it is only worthwhile measuring things to satisfy specific goals. The technique starts by identifying some high level goals related to the project (e.g. "keep the project on schedule", "sell more units than any other team", "get all the bugs out", etc). For each goal, it identifies a number of questions that need to be asked if the goal is to be achieved (e.g. "how far behind schedule are we?", "how many units are the other teams selling?", "how many bugs are left?"). Then, for each question, it identifies some measurements that can be taken to answer the question. A question may be related to more than one goal, and a measurement may help to answer more than one question, so the result is a graph like this:



As an example, imagine we have identified a goal "Improve the timeliness of handling change requests from users". We might identify a number of questions that need to be addressed:

- "What is the current speed at which we handle requests" which we could answer by measuring the average cycle time for change requests, the standard deviation, the percentage of requests that are above some upper limit, etc.
- "Is the current performance improving?" which we could answer by measuring the current average cycle time over some baseline average cycle time, or perhaps just a subjective assessment of the manager's or users' satisfaction.

Each goal refers to a particular object, which may be a product (e.g. a specification, a design, a program unit, a test suite), a process (e.g. specifying, designing, testing, interviewing, selling), or a resource (e.g. personnel, hardware, software, office space). For example, the goal described above refers to a process: handling change requests. Questions identify the information needs associated with goals.

Metrics are specific items that can be measured to answer the questions. Metrics may be:

- algorithmic—if they do not depend on the viewpoint of the person or machine making the measurement
- subjective—if they involve some subjective judgement, and hence may vary depending on who is doing the measuring

Each metric will have a scale of units associated with it, which may be one of the following:

Nominal—A nominal scale is an unordered set of named categories, such that the only comparison that make sense is equality.

Ordinal—An ordinal scale is an ordered set of categories, such that tests of relative size ('greater than', 'less than') make sense. For example, a five point scale where each point on the scale is somehow 'bigger' than the previous point

Interval—An interval scale is an ordered scale where the intervals between the points on the scale are constant, so that addition and subtraction make sense. For example, temperature measured in centigrade: we can add and subtract temperatures, but it does not make sense to multiply them (i.e. 40˚C is *not* twice as hot as 20˚C)

Ratio—A ratio scale is an ordered scale where the intervals between points on the scale are constant, and there is an absolute zero, so that multiplication and division make sense. For example, temperature measured in Kelvin (for which it does make sense to say that 40˚K really is twice as hot at 20˚K).

Absolute—An absolute scale is a count of the number of occurrences of something. This is like a ratio scale, except that the only meaningful points on the scale are zero and the positive integers.

Notes on software measurement can be found in chapter 6 of van Vliet. There is also a paper on GQM available on the course website.