Running Head: GOAL QUESTION METRICS

Defining Metrics Effectively: Goal Question Metrics (GQM)

Jered McClure

Walden University

Defining Metrics Effectively: Goal Question Metrics (GQM)

Goal Question Metrics (GQM) is a technique where specific aims are defined and then questions towards those aims are applied from which metrics are assembled (Basili, Caldiera, & Rombach, 1994). This is important as it helps to address key software development issues such as early lifecycle defects and changing requirements. Both of these are important issues to resolve as soon as possible during any software project. Moreover, each is a necessary byproduct of any development project which, in general, cannot be bypassed.

Most projects are plagued by the issue of requirements bloat. That is, where requirements are constantly added to the original project scope such that time is wasted simply reworking development planning to cater for the new requirements. GQM addresses this issue through a thorough breakdown of why the requirements are being added or changed, and for whom the changes are being made. In essence, GQM acts as a high level method of change control.

For instance, GQM denotes that a goal must be defined for each issue. In this regards a requirement must be added to the project. Ergo, the Goal is broken into the purpose, issue, object, and viewpoint of the specific question: “Add a requirement to the software project for an employee or group.” From this, questions can be defined which then lead to metrics that can be physically or subjectively measured. Fundamentally, each requirement that needs to be added to the software project must first answer each goal question and be measured as passing before it can be added.

“Resolve early lifecycle defects on the software project for the project manager.” This is the goal statement for the lifecycle defect issue. From this goal statement questions are defined, such as, “Is this defect connected to other software objects?” and “Does the defect adversely change the project timeline.” These two questions both have clear metrics, the first is a Boolean true false, and the second is an estimated fix time. Further questions can be defined, the point is that each question must be answered per defect, thereby ensuring no defect is overlooked and thus causing issues further in the development lifecycle.

 The key elements of GQM are in its ability to dig down to the measureable metrics through a hierarchical Goal-Question-Metrics tree. The Goal of each GQM group sits at the top, broken out into the issue, object, viewpoint, and purpose. Beneath this are the questions (there can be more than one), and all fall into one of three areas: 1. characteristics of the object, 2. the attributes of the object, and 3. the evaluation of the characteristics from the goal’s viewpoint (Basili, Caldiera, & Rombach, 1994). Finally at the bottom of the tree are the metrics which answer the questions previously defined and are usually broken into subjective and objective areas. Subjective metrics are those that can only be measured from a specific viewpoint, objective metrics are specific to the goal object in question and can be measured from any viewpoint.

GQM’s primary strengths are in its ability to help define what metrics actually are, given any specific software development project. This is exceedingly important as understanding what to measure early on in any project can be difficult without guided brainstorming. The downfall to this is that GQM is specific to software development. It is possible to bend the rules of GQM a bit to make it adaptable to other process areas, but at its heart it is a software development metrics definition model.

To ensure GQM is followed through by employees, a motivational incentive program needs to be setup. First and foremost, make completing GQM programs easy and second nature. That is, the process of answering each GQM question and providing measurement should be melded into any current development processes. For instance, during software delivery, documentation must be provided, which includes the GQM metrics data. This removes any additional re-documenting which would occur otherwise.

Secondly, staff should be made aware that metrics measurements are about improving product performance, and not about measuring employee activity. This changes the focus of metrics gathering from a personal attack, to one of a project oriented tool. Metrics are there to increase quality and provide a better tool for the customer, not monitor and micromanage employees.

Management can assist in both of these through a campaign of actively using and requesting said metrics to monitor the software development status. This needs to be done in a way that is public to staff, and anonymous so that staff are aware metrics are about the software and not about employee appraisal. Furthermore, management should make it clear that GQM completion is about providing a better product to the customer. This in turn needs to be a key organizational value, thus allowing employees to internalize GQM completion as a method of meeting their own values (Evans, 2010).

Goal Question Metrics helps management and employees effectively define and measure metrics to satisfy software development goals. It clearly helps alleviate the issue of determining what defects are, and how to measure them early in the development lifecycle. Further to this, it allows for a layer of change management between requirements bloat and the software development timetable. In essence, GQM is a methodology aimed at increasing software quality.

Reference

Basili, V. R., Caldiera, G., & Rombach, H. D. (1994). *The Goal Question Metric Approach.* Institute for Advanced Computer Studies, Department of Computer Science. College Park: University of Maryland.

Evans, J. R. (2010). *Quality & Performance Excellence: Management, Organization, and Strategy* (6th ed.). Mason, OH, USA: Cengage Learning.