PRACTICAL 3

AIM:

To learn how to select a Software Quality Assurance Tool and to study the benefits and costs associated with Automated Software Quality Assurance Tools.

SOFTWARE QUALITY ASSURANCE

Software quality assurance (SQA) consists of a means of monitoring the software engineering processes and methods used to ensure quality. The methods by which this is accomplished are many and varied, and may include ensuring conformance to one or more standards, such as ISO 9000 or a model such as CMMI.

SQA encompasses the entire software development process, which includes processes such as requirements definition, software design, coding, source code control, code reviews, change management, configuration management, testing, release management, and product integration. SQA is organized into goals, commitments, abilities, activities, measurements, and verifications.

ADVANTAGES

- Domain-specific languages allow solutions to be expressed in the idiom and at the level of abstraction of the problem domain. Consequently, domain experts themselves can understand, validate, modify, and often even develop domain-specific language programs.
- Self-documenting code.
- Domain-specific languages enhance quality, productivity, reliability, maintainability, portability and reusability.
- Domain-specific languages allow validation at the domain level. As long as the language constructs are safe any sentence written with them can be considered safe.

DISADVANTAGES

- Cost of learning a new language vs. its limited applicability.
- Cost of designing, implementing, and maintaining a domain-specific language as well as the tools required to develop with it (IDE).
- Finding, setting, and maintaining proper scope.
- Difficulty of balancing trade-offs between domain-specificity and general-purpose programming language constructs.
- Potential loss of processor efficiency compared with hand-coded software.
- Proliferation of similar non-standard domain specific languages, i.e. a DSL used within insurance company A versus a DSL used within insurance company B.
- Non-technical domain experts can find it hard to write or modify DSL programs by themselves.
COSTS ASSOCIATED with Software Quality Assurance

Conformance Costs: The conformance costs comprise all costs that need to be spent to build the software in a way that it conforms to its quality requirements. This can be further broken down into prevention and appraisal costs. Prevention costs are for example developer training, tool costs, or quality audits, i.e. costs for means to prevent the injection of faults. The appraisal costs are caused by using various types of tests and reviews.

Nonconformance Costs: The nonconformance costs come into play when the software does not conform to the quality requirements. These costs are divided into internal failure costs and external failure costs. The former contains costs caused by failures that occurred during development, the latter describes costs that result from failures at the client.

COST FACTORS

Setup Costs: The costs to set up the application of a specific technique are one main part of the costs of analytical QA. The importance of this factor is directly derived from the quality cost model and it can be significant considering the effort needed to build a suitable test environment. The personnel effort is also dependent on the labour costs.

Execution Costs: In addition to the fixed setup costs, we have variable execution costs. This factor is also directly derived from the quality cost model. The execution costs are, as discussed above, mainly personnel costs and hence dependent on another factor, the labour costs.

Fault Removal Costs: When a defect is detected, the second step is to remove that defect. This has also different costs mainly consisting of personnel costs. Hence, we have an influence from the labour costs. Many other factors have an effect on this: inspections detect faults whereas tests only detect failures and leave the fault isolation as additional step. Hence, this has an influence on how laborious the removal is. Furthermore, it is dependent on the type of document and phase in which the defect is detected. A found defect during requirements analysis involves only to change the requirements document. Detecting the same defect during system test might require to change several documents, including the code and the design, and to re-inspect and re-test the software. When the defect is revealed in the field, we may have more costs for support staff. It is common software engineering knowledge that defect removal is the more expensive the later the defect is revealed.

Effect Costs: This is the most difficult to determine factor in the quality economics of defect-detection techniques. It contains all further costs that a defect in the field has apart from the removal costs. This may include compensation costs in case the user of the software had any damage because of the software defect. Other factors such as lost sales because of bad reputation belong to this factor as well but are not further considered in this dissertation.
**Labour Costs**: The labour costs are a secondary factor because they do not influence the quality costs directly but have an effect on several of the other cost factors. Therefore, it is worthwhile to include it in the list and analyse it later on. Most of the costs incurring in analytical QA can be attributed to personnel effort. Hence, we can often measure the time effort spent for different activities and multiply them with the labour costs. Those need not only to contain the salary of the staff but all additional 334 Quality Economics costs caused such as sick days or training. This is often called a loaded labour rate. We assume this to be available as this is a standard economics approach.

**SELECTING a Software Quality Assurance Tool**

A Software Quality Assurance Automation Tool shall be selected such that it:

- Manages requirements
- Aligns testing priorities based on risk
- Defines test plans
- Monitors quality across releases and cycles
- Schedules and runs tests
- Tracks defects
- Uses versioning and baselining