Click here to review OO Testing Strategies

Software Metrics

Chapter 4

SW Metrics

- SW process and product metrics are **quantitative measures** that enable SW people to gain insight into the efficacy of SW process and the projects that are conducted using the process as a framework.

SW Metrics Terms

- **Measure** –
  - provides a quantitative indication of the extent, amount, dimension, capacity or size of some attribute of a process or product.
  - Example: Number of defects found in component testing. LOC of each component.
- **Measurement** –
  - The act of collecting a measure.
  - Example: Collecting the defect counts. Counting LOC.

SW Metrics Terms

- **Metric** (IEEE Standard Glossary of Software Engineering Terms)
  - A quantitative measure of the degree to which a system, component or process possesses a given attribute. It relates measure in some way.
  - Example: defects found in component testing/LOC of code tested.
- **Indicator** –
  - A metric that provide insight into the SW process, project or product.
  - Indicators are used to manage the process or project.

SW Metrics Terms

- **SW Metrics** refers to a range of measurements for computer software that enable software people to gain insight into the project:
  - To improve the Process and the Product
  - Assist in Estimation
  - Productivity Assessment
  - Quality Control
  - Project Control

SW Metrics

- **How are metrics used?**
  - Measures are often collected by SW engineers and used by SW managers.
  - Measurements are analyzed and compared to past measurements, for similar projects, to look for trends (good and bad) and to make better estimates.

SW Metrics

- **Reasons for measuring SW processes, products, and resources:**
  - To characterize
    - To gain understanding of Product, Process, and ?
    - To establish baseline for future comparisons
  - To evaluate
    - To determine status within the plan
  - To predicate
    - So that we can plan. Update estimates
  - To improve
    - We would have more information "quantitative" to help determine root causes

Three domains of SW metrics

- **Product –**
  - These measurements relate to SW product and all related artifacts.
  - Examples: code, design docs, test plan, user manual …LOC, # of objects, # of pages, # of files.
  - Measures can also be used to evaluate the SW quality:
    - Cyclomatic complexity: a way to measure the complexity of a module.
    - It assigns a value V(G) to a module based on the control flow of the module. Some companies place a cap on V(G). If too high, the module must be redesigned.

Three domains of SW metrics

- **Process: -**
  - These measures used to quantify characteristics of the SW process.
  - Usually related to events or things that occur.
  - Examples: # defects found in test, # requirements changes, # days to complete task …
- **Project** –
  - used to manage the SW project “Tactic”.
  - Estimating cost is the first application of Project Metrics
  - Examples: estimates of SW development time based on past projects.

Two types of uses for process metrics

- **Private metrics –**
  - measures taken of an individual's software process. These are usually private to the individual or team. Used to improve an individual's performance or personal software process.
  - Example: defect rate for an individual.
- **Public metrics –**
  - measures taken at a team level. These are made public to the organization. Used to improve an organizations process maturity.
  - Example: defects found after release per KLOC.

Software Measurement

- Two categories of measurement:
  - **Direct measures –**
    - measurements that are more tangible.
    - Examples:
      - cost, time, and efforts are Direct Process measures
      - LOC, memory size are examples of Direct Product measures
  - **Indirect measures –**
    - measurements of things that describe the characteristics of a product or process. These are the "-abilities".
    - Examples: functionality, quality, complexity, reliability, maintainability…

Software Measurement

- Size Oriented Metrics
  - Derived from the size of the software (KLOC)
    - Errors per KLOC
    - Defects per KLOC
    - $ Per LOC
    - Errors per person
    - LOC per Person-per-Mont
  - Do you see any problem with Size Oriented Metrics (I.e thing of Assembly vs. C++)?

Software Measurement

- Functional-Oriented Metrics
  - use measures of the functionality delivered by the application.
  - Functionality is derived indirectly using other direct measures.
  - Function Points are derived using direct measures of software's information domain.
    - FPs are computed using a simple tabular form
    - Text Book Page 90.

Software Measurement

- Functional-Oriented Metrics

Complexity adjustment values

$$FP = \text{count Total} \times [.65 + .01 \times \sum (F_i)]$$

From table Text book page 90

Constants derived using historical data

i=1 to 14 Survey, see book page 91

Software Measurement

- Once FPs has been calculated we can:
  - Errors per FP
  - Defects per FP
  - $ per FP
  - FP per Person-per-Month

Metrics for SW Quality

- 4 suggested quality measures
  - Correctness –
    - the degree to which the SW performs its required function.
    - Usually measured by defects/KLOC.
    - A defect is defined as a verified lack of conformance to requirements. Usually counted over a standard period of time.
  - Maintainability –
    - the ease to which a program can be corrected, if an error is found, changed, for a new environment, or enhanced for a user request.
    - No direct way to measure this.
    - One measure is mean-time-to-change which measures that time from when the change request is analyzed to when it is distributed to the user

Metrics for SW Quality

- 4 suggested quality measures
  - Integrity –
    - a system's ability to withstand attacks to its security. Attacks can be made to a systems data, programs or documents.
    - Threat - the probability that an attack will occur in a given time period.
    - Security - the probability that an attack will be repelled.
    - These values are estimated or derived from empirical evidence.
    - Integrity = summation [(1-threat) * (1-security)] summed over each type of attack.

Metrics for SW Quality

- 4 suggested quality measures
  - Usability –
    - an attempt to measure "user-friendliness"
    - Can be measured in terms of 4 characteristics:
    1. The physical and/or intellectual skill required to learn the system
    2. The time required to become moderately efficient in the use of the system.
    3. The net increase in productivity, measured against the old process or system, measured after a user has gained moderate efficiency.
    4. A subjective measure of user attitude towards the system

Process Metrics

- **Suggested guidelines:**
  - Use common sense and organizational sensitivity when interpreting metrics data.
  - Provide regular feedback to the individuals and teams who collect measures and metrics.
  - Do not use metrics to appraise individuals.
  - Set clear goals and metrics that will be used to achieve them.
  - Never use metrics to threaten individuals and teams.
  - Metrics that indicate a problem area should not be considered a problem, but an indicator for process improvement.
  - Do not obsess on a single metric.

Project Metrics “Estimates”

- Estimation Model:
  - Use derived formulas to predict efforts as a function of LOC or FP.
  - All models take the form:
  - $E = A + B \times (ev)^C$ where
    - A,B, and C are derived constants
    - E is the effort in person-months
    - ev is the estimation variable (LOC or FP)
  - A model could also have “project adjustments”
    - Enable E to be adjusted by other project characteristics like:
      - Problem complexity, staff experience, environment.
    - Examples:
      - $E = 5.2 \times (KLOC)^{0.91}$ (LOC-Oriented)
      - $E = -91.4 + .355\ FP$ (FP-Oriented)

Project Metrics “Estimates”

- The COCOMO Model
  - It stands for COnstructive COst MOdel by Bary Boehm.
  - Most widely used
  - It’s a hierarchy of estimation models that address the following areas:
    - Application composition model –
      - Used during early stages - prototype
    - Early design stage model –
      - used once requirements have been stabilized
    - Post architecture stage model
      - Used during the construction of the software

Project Metrics “Estimates”

- The COCOMO Model
  - Uses the following sizing measures
    - LOC
    - Function Points
    - Object Points: computed using counts of the number of
      - Screens, Reports, and Components.
      - Each object criteria is classified as either SIMPLE,MEDIUM, or DIFFICULT
      - Each class is given a Weighting Factor as follows:
        - » Screen simple =1, Screen Medium = 2, Screen Difficult = 3
        - » Report simple = 2, report Medium = 5, Report Difficult = 8
        - » Component Difficult = 10
      - Object Point = Weighting Factor x Number of Object (i.e. 3 screen)

Project Metrics “Estimates”

- The COCOMO Model
  - Object Point = Weighting Factor x Number of Object (i.e. 3 screen)
  - NOP = (Object Points) x [(100 - %reuse) / 100]
  - PROD = NOP / Person-Month
  - Estimate Effort = NOP / PROD

NOP = New Object Point
PROD = Productivity Rate

Project Metrics "Estimates"

- The Software Equation
  - Dynamic multivariable model
  - Based on data collected for over 4000 software projects
  - Estimation model
  - **$E = [LOC \times B^{0.333} / P]^3 \times (1/t^4)$**

E : Effort in person-months
t : project duration
B : "special Skill factor"[16]
P : Productivity parameter

- **P=2000 for real-time embedded SW**
- **P=10,000 for telecommunication**
- **P=28,000 for business systems**

Software Quality Assurance

Chapter 8

SQA

- The goal of any SW is to produce high-quality software
- Its applied throughout out the software process
- A set of steps that ensures every product is of high quality
- Use metrics to develop strategies for improving the process and the quality of the final product
- Every one is responsible for quality
- Good quality means lower cost and reduction in time-to-market

SQA

- SQA activities should be defined at the beginning of the project
  - They will help to filter errors in the process and ultimately in the product.
  - They are set at each phase

SQA

- **SQA encompasses:**
  - Quality Management approach
  - Effective SW engineering technology
    - Methods and tools
  - Technical reviews (Formal)
  - Testing strategies
  - Change Control
  - Compliance procedures
  - Measurements and reporting mechanisms

SQA

- **Variations:**
  - No 2 products are a like
  - All engineered products exhibit variations
    - SW engineering products too
  - We need “equipments” to measure variations
  - In software we would like control variations
    - Between projects
    - Estimates
    - Defects found between projects
    - Time variations
    - Etc.

SQA

- **Quality:**
  - "a characteristic or an attribute of something"
  - Measure-able characteristic
    - Cyclomatic complexity
    - Cohesion
    - Coupling
    - LOC
    - Number of FP
    - Number of Object Points

SQA

- **SQA** are all the activities applied throughout the software process to ensure every work product developed in the process is of high quality.
- What is Quality, in respects to software and the process of constructing software?

Quality Concepts:

- **Quality of Design:**
  - includes the requirements specification and the design of the system.
- **Quality of Conformance:**
  - focuses on if the implementation follows the design and if the resulting system meets the requirements of the system.
- **Quality Control**
  - is variation control. This involves all the activities such as inspections, reviews and tests used throughout the software process to ensure each work product meets the requirements placed on it.

- There is a feedback loop that allows the process to be improved, if the work product falls short of what was expected.

- In terms of SW, a company strives to minimize the difference between the estimated resources needed to complete a project and the actual resources used.

Examples

- Company testing program always finds the same percentage of defects from release to release.
- Projects complete within the same margin of error from the projected completion date.
- Projects complete within the same margin of error using the projected human resources.

- A key concept of quality is that all work products have defined, measurable specifications to which we may compare the output of each process.

- Examples of this are the templates we used for each phase of the life cycle process.

Quality Assurance

- consists of the auditing and reporting functions of management.
  - The data collected is used by management to understand the quality of the product.
  - Management may change the process, if the data collected indicates a problem.
- Example:
  - The number of defects in new products is too high. Management may change black box testing procedures and add time to the projects in the test phase to help catch more of the defects before the product is released.

Cost of Quality

- is the sum of all costs incurred from all activities related to ensuring quality.
- A cost of quality study is conducted to produce a baseline for the current cost of quality.
- This number is used to manage against and evaluate possible changes to the process.

Costs can be categorized

- Prevention costs:
  - quality Planning
  - formal technical reviews
  - test equipment
  - training
- Appraisal costs:
  - (Gives us an idea of where we are the first time through a process.)
  - in-process and inter-process inspections
  - equipment calibration and maintenance
  - testing

Costs can be categorized

- Failure costs:
  - (Internal failure costs - prior to shipping)
  - rework
  - repair
  - failure mode analysis
  - (External failure costs - after product is shipped)
    - complaint resolution
    - product return and replacement
    - help line support
    - warranty work

- The earlier in the process defects are found, the less expensive they are to correct.

- See page 198 of text.

SQA Activities

- A set of activities designed to help the project team achieve high quality. (Basically, keep the team honest!)

1. Prepare an SQA Plan for the project. This describes the following:
   - Evaluations to be performed
   - Audits and reviews to be performed
   - Standards that are applicable to the project
   - Documents to be produced by the SQA team
   - Amount of feedback to be provided to the software team

SQA Activities

2. Team chooses a process to follow for the project. SQA reviews the process for compliance with company guidelines.
3. SQA reviews teams progress for compliance to the chosen process.
4. SQA performs the following:
    1. reviews selected work products
    2. ensures all reviews are performed
    3. ensures corrections are made
    4. reports results to management
5. Ensures deviations are handled properly.
6. Records any noncompliance and reports to management.

Software Review

- A filter for defects in the software development process.
- Reviews are performed at various points in the software process and serve to uncover defects that can be removed.
- **Formal technical review:**
  - An SQA activity performed by SW engineers to:
  - Uncover errors in any representation of SW (requirements, design or code)
  - To verify the SW under review meets it requirements
  - To ensure the SW has been represented according to predefined standards
  - To achieve SW that is developed in a uniform manner
  - To make projects more manageable.

Software Review

- The Review meeting:
  - 3-5 People
  - Advance preparation should occur, less than 2 hours per person
  - Meeting should last less than 2 hours
- Review Guidelines:
  - Review the product, not the producer.
  - Set an agenda and stick with it.
  - Log issues and problems, do not try and resolve them.
  - Take written notes.
  - Limit number of people. Too many is not productive.
  - Develop checklist for each product to be reviewed.
  - Allocate resources and time in project plan.
  - Conduct meaningful training for all reviewers.
  - Review your earlier reviews (checklists and results…) Constantly improve the review process itself.