Software Engineering

CS305, Autumn 2020

Week 14

Class Progress…

- Last week..

  Topics in Software Construction

  - **Software Verification** – “checking for bugs”
    - Static analysis - Code coverage and demo of tools (Gcov, Eclipse)
  - **CI/CD** - Continuous Integration / Continuous Deployment and demo (CI) using GitHub Actions

Class This Week..

- Topics in
  - Software Quality
  - Project Management

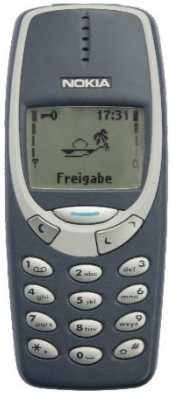
Quality

- Informally, quality of a product is the presence of all features that the customer requires and absence of those features that the customer does not require.
- ISO defines quality as:
  The totality of features and characteristics of a product or service that bears on its ability to satisfy stated or implied needs
- Today, the term quality includes:
  - Features and characteristics that affect suppliers, shareholders, employees, management, and larger community
  - Work life, workplace diversity, environmental conditions, and competitiveness.

Quality

- What about Software Quality?
  - It should perform intended function, perform under specified environment and constraints, be reliable, maintainable, and interoperable (*all at a cost that the customer can afford*).
    - Some of the above requirements may be stated or implied e.g. customer may not say anything about performance requirements or maintainability

Judging Quality



https://en.wikipedia.org/wiki/Nokia_3310

https://en.wikipedia.org/wiki/IPhone

Does lower cost mean lower in quality?

No. It depends on stated and implied needs of current and potential customers.

- Make an apples-to-apples comparison of attributes

Improving Quality

- Follow software processes
  - Recall that software process is an ordered sequence of activities that transform an idea to software
    - Inputs are software requirements
    - Output is the delivered software
    - Logical grouping of activities under stages of requirements engineering, design, coding, testing, deployment, and maintenance.
  - Why do we need software processes?
    - They are best practices for yielding good results
    - Ensure consistency in software creation

Improving Quality

- Importantly, following processes, we can predict:
  - How many defects are likely to be in the final software delivered to customer?
  - When can we realistically deliver software?
  - How much is it going to cost when we deliver the software?

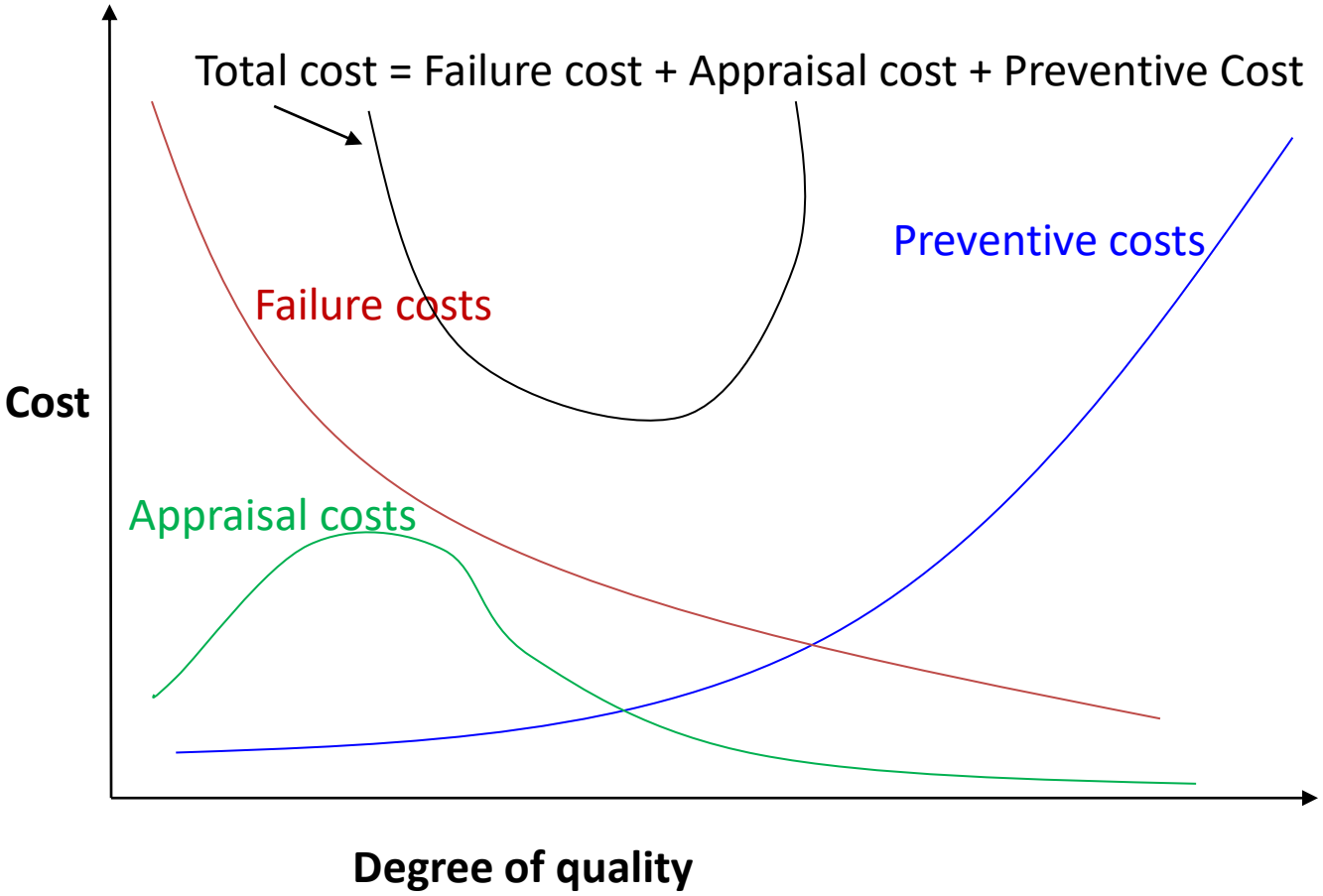
Why Improve Quality ?

- To maintain competitive edge
  - Software is no exception
- To grow business
  - satisfy customers and expand customer base
- To keep up-to-date with technological advances
  - Tools and methods arrive at a faster rate than it takes to train the staff on their use
- To control the cost of defects in delivered software

Cost of Quality (COQ)

- Is not equal to cost of defects in delivered software
- Is how you quantify quality in terms of money
  - Lower the COQ, better
- Uses:
  - Quantify cost of software that fails to be delivered
  - Quantify the cost of having defensive code to avoid problems rather than using sophisticated review and testing methods to catch them
- E.g. British Standard BS 6143: Guide to Economics of Quality describes Process Analysis Model (PAM) and Prevention, Appraisal, and Failure (PAF) model for recording quality costs

COQ – PAF Model



PAF Model

- Failure costs
  - Recall that failure, as per ISO definition, relates to coding errors/bugs (in the context of software). Failure costs result from the cost of fixing bugs (before and after deployment), handle customer complaints
- Preventive costs
  - Costs that are incurred towards preventing poor quality product. E.g. accurate documentation, requirement analysis, staff training
- Appraisal costs
  - Cost incurred towards finding problems e.g. code inspections, design reviews, black- white-box testing, beta testing, test automation, etc.

Software Quality Factors

- What are the *features* and *characteristics* in the ISO definition of Quality applicable to Software?
  - Correctness: extent to which program meets its specifications
  - Maintainability: effort required to locate and fix bugs or to introduce new features
  - Reliability: extent to which the software performs its intended function with the required precision
  - Portability: effort required to transfer a program from one program environment to another
  - Usability: effort required for learning and operating the software
  - Reusability: extent to which the product can be reused in other contexts

Total Quality Management (TQM)

- TQM loosely means approaches towards continuous improvement that lead to quality products
- Probably first adopted by US Naval Systems in 1985
- ISO definition:
  A management approach to an organization centered on quality, based on the participation of all its members and aiming at long-term success through customer satisfaction, and benefits to the members of the organization and to society.

TQM Summary

- Quality is an organization-wide process
- Quality is what customer says it is
- Quality is an ethical responsibility

How to implement TQM is up to the organization. However, the motivator should be pursuit of excellence.

TQM Implementation Guidelines

- Incorporate Essential Components
  - Effective planning
    - E.g. use of tools such as Microsoft Project
  - Measurement
    - Function Point estimation, COQ etc.
  - Control mechanisms
    - Change and version management

TQM Implementation Guidelines

- Create an objective that commits entire company to total quality and customer satisfaction as primary function of business
- Determine what customers want
- Deliver on promises made to the customer by
  - Building quality product that includes all the quality factors mentioned earlier (..ilities)

ISO Quality Standards

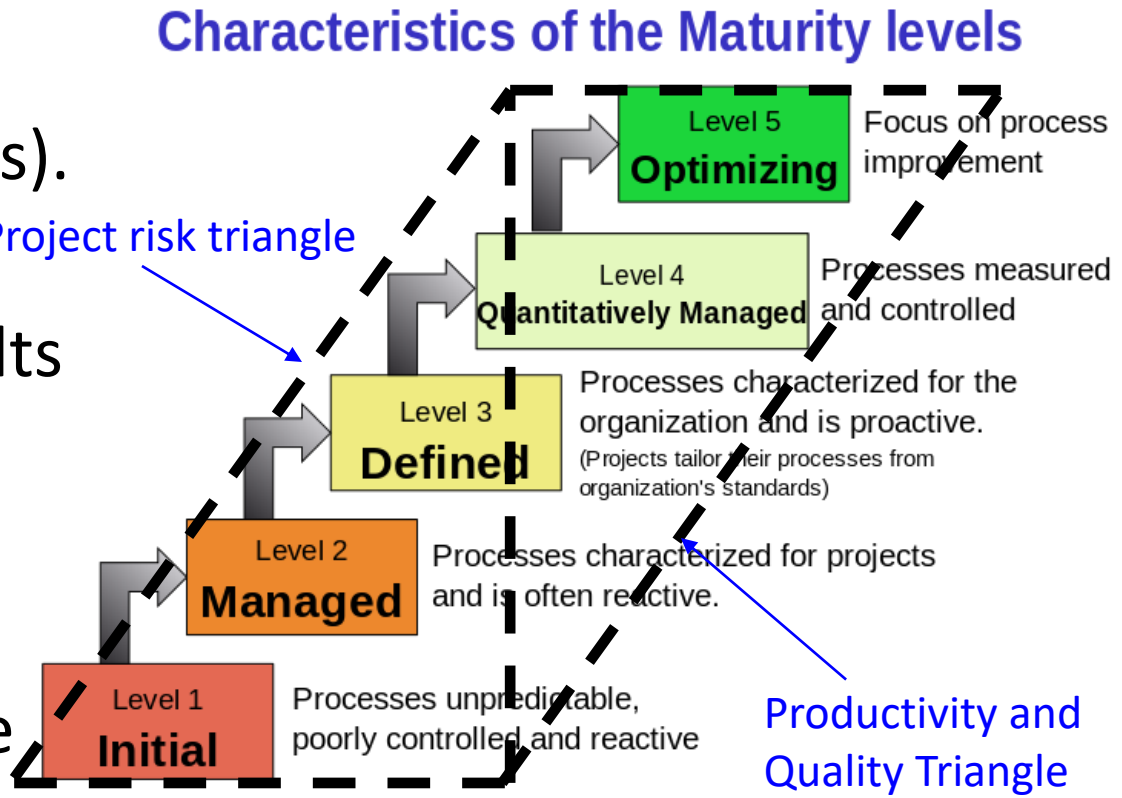
- Based on TQM
  - ISO 9000 is about processes
    - Consists of ISO 9000, ISO 9001, ISO 9004
    - ISO 9000 is the standards document prescribing roadmap for implementation of ISO 9001-9003
    - ISO 9001-2000 is about certifying quality in design/development, production, installation, and servicing.
    - ISO 9004 for quality management and assurance
  - ISO 14000 is about environmental management
- Can be applied to any organization / department, any product.
- External audits done once in 6 months. Internal audits not mandatory.

Capability Maturity Model (CMM)

- ISO is generic. CMM is specific to Software industry.
  - ISO is subjective at times, CMM is not. CMM is widely used in the US.
- Based on TQM
- Define how software organizations mature or improve in their ability to develop software
- Developed by Software Engineering Institute (SEI) at Carnegie Mellon University in late 80s.

CMM Levels

- 5 levels or key process areas (KPAs).
- Organization can deliver better results if the KPAs are implemented.
- Higher the level, more mature is the project / organization.



https://en.wikipedia.org/wiki/Capability_Maturity_Model_Integration

CMM Levels

- Level 1: Not systematic. Success dependent on people working in the project. Not scalable. Not consistent.
- Level 2: Can repeat success in similar projects. Focus on 'Project Management'. But best practices are not shared across organization.
- Level 3: Focus on standardizing best practices and processes at an organization level. Project history, data collected and shared through central repo.
- Level 4: Predictable process capability of the organization at this level. Projects use project data to benchmark quality goals, to plan, to improve. The processes are quantitatively established and published.
- Level 5: Focus area is 'continuous improvement'. Improvement means reducing variation, proactive assessment of weaknesses and strengthening (all organization-wide).

CMM Concluding Remarks

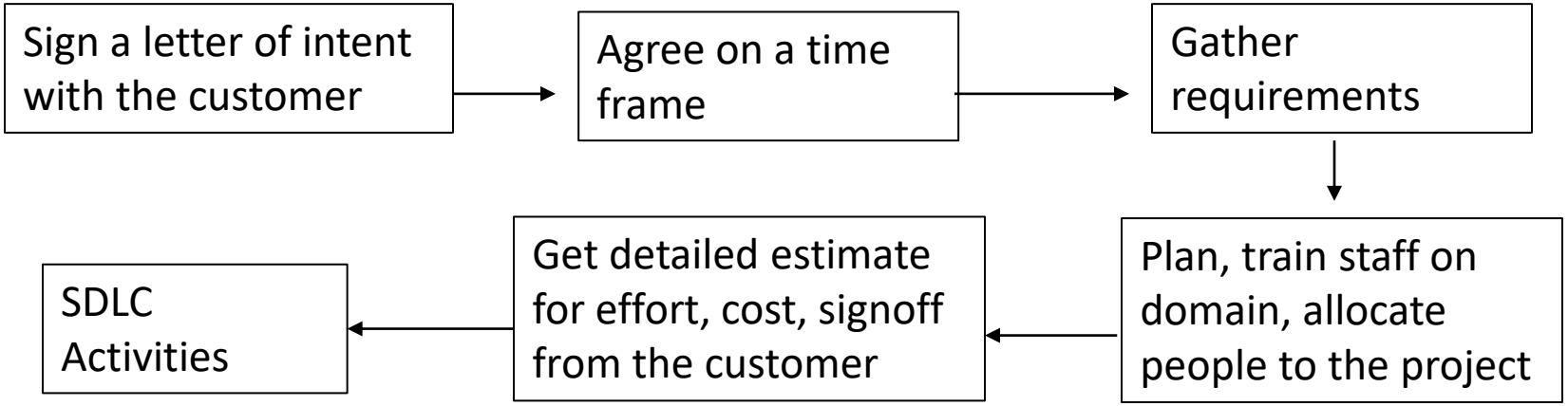
- No audits after CMM assessment
- It is only for software divisions unlike ISO
- Can be used as a roadmap for organizational improvement unlike ISO

Further Reading

- Lectures 21, 34,35,36 of https://nptel.ac.in/courses/106/101/106101061/

Project Management

- Optimal utilization or project resources to deliver the project according to specifications within the specified time and cost budget.



Also includes negotiating on delivery dates when project significantly deviates from plan:

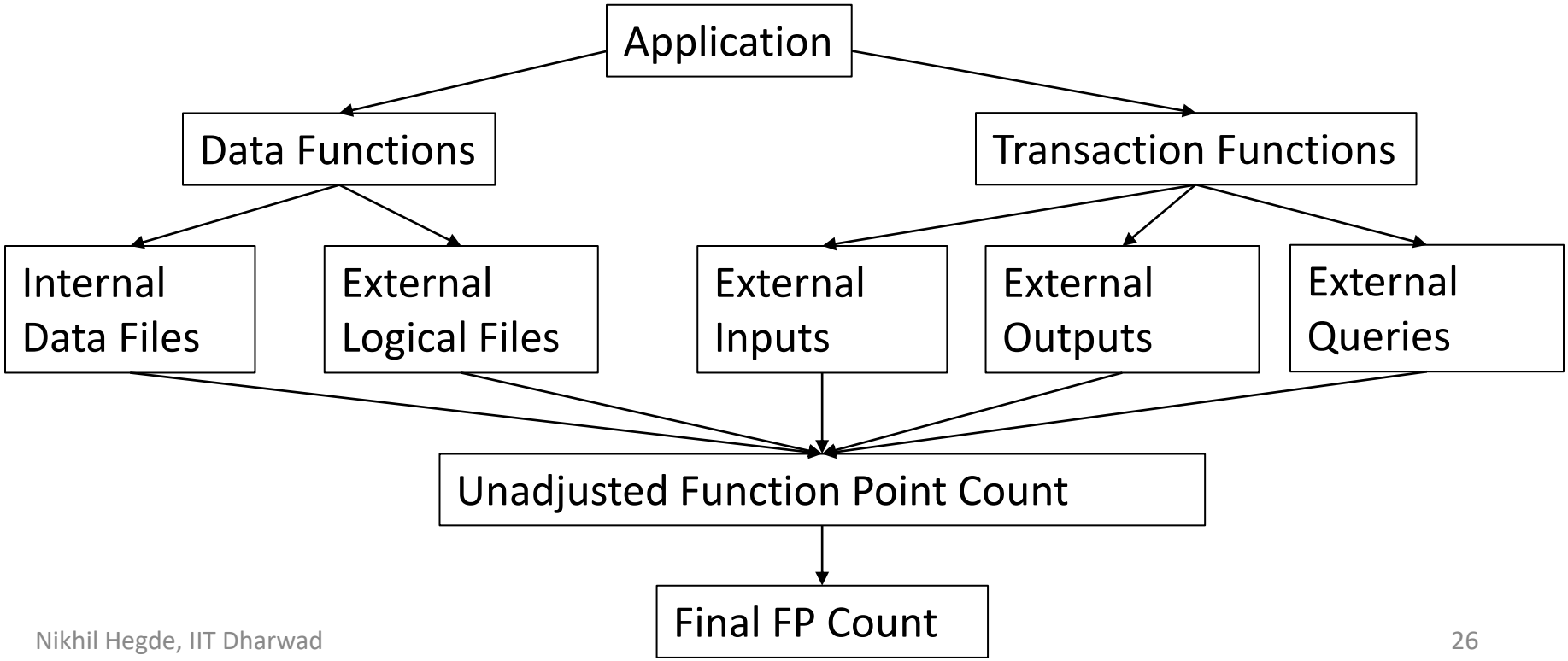
- *due to changes in customer requirements*
- *due to other reasons*

Effort Estimation

- Project's success / failure depends on how accurately the effort is estimated. Includes predicting:
  - Cost
  - Manpower
  - Schedule (timeline)
- Multiple methods exist:
  - Function Point (FP) estimation
  - Cost Constructive Model (COCOMO)
  - Simple-Medium-Complex (SMC)
  - …

FP Estimation

- Pioneered at IBM. Size of the application software is estimated based on the number of functionalities that the software provides.



FP When Not Applicable?

- When you want to estimate the time required
- When you want to estimate the effort for system software (and not application software)
- When you want a breakup of efforts required for Analysis, Design, Coding etc.

COCOMO Estimation

- Cost estimation model introduced by Barry Boehm in 80s.
- Fits a regression formula using historical project data
- Each of 15 attributes related to Product, Hardware, Personnel, and Project receives a 6-point rating.
- Requires lines of code as input
- Has 3 modes organic, semi-detached, embedded to estimate time and effort for different types of software

Other Project Management Processes

- Staffing
  - Earlier phases of SDLC require lesser number of people compared to coding and unit testing. Staffing follows Raleigh's distribution curve (basically non-linear)
  - Should have a blend of expert to beginner skill levels
- Scheduling
  - Typically, 1.2 to 1.5 times the effort / average staff size
- Change Management
  - Changes to requirements from customers and its impact
- Configuration Management
  - Artifact management. e.g. change in HLD should reflect in code and test plans.

Further Reading

- Lectures 29-33, 37-39 of https://nptel.ac.in/courses/106/101/106101061/