Software Engineering

Prof Ravi Prakash Gorthi

Jan-April, 2011

Topics

1. Introduction
2. Software Engineering Models
3. BPM, BRMS, SOA
4. Software Requirements Engineering
5. Software Analysis Models
6. Software Project Management
7. Software Design Concepts, Principles and Models
8. Software Coding Practices
9. Software Testing Techniques
10. Software Quality Assurance
11. Emerging Trends in Software Engineering

Text & Reference Books

Text Books:

- Software Engineering: A Practitioner's Approach by Roger Pressman, McGraw-Hill, 6th edition
- Software Engineering by Ian Sommerville, Pearson Education LPE, 8th or 9th edition
- An Integrated Approach to Software Engineering, Narosa Publishing, 2nd edition

Acknowledgement: Contents of some of the slides are taken from Ian Sommerville's course ppt

Reference Books:

4. Software Engineering by S. L. Pfleeger, MacMillan Publishing
5. Research papers in the area of SE – to be distributed by Prof Gorthi

About my SE experiences

1978 to ’81: Software Engineer and team member of the Implementation Team (size ~40) of Air-India’s Airline Reservation System; around 1000 Fortran and Assembly language programs of 5000 LoC using ~ 250 Networked database tables were customized, integrated, tested, parallel runs and made online on Univac’s the then largest main-frame taking around 2.8 years;

1982-83: In New Zealand, as a SSE and Team Lead, was part of a TCS team (size ~50) that converted around 800 COBOL programs from Burroghs COBOL to FACOM COBOL; Burroughs used Hierarchical DBMS while FACOM used Networked database tables (~300 tables); project took around 1.5 years;

1985-86: As a PM of a TCS team in Holland, did a very successful Software Re-engineering project for a major bank in Holland; reduced ‘GoTo’ statements from ~ 25% to less than 1% from 10 PL/1 based modules;

1987-88: As Asst Consultant and PM, was part of a TCS team, that designed and developed AI based software to correlate hundreds of parameters of GSLV rockets of ISRO; the s/w was used for over 10 years by ISRO;

About my SE experiences

1995 to ’01: As a Group Technical Manager at IBM GS, Bangalore, offered Administration and Performance Tuning of DB2/400 and OS/400 systems to some of the reputed IBM clients in India and Asia-Pacific region;

2001 to ’03: As a VP of a Start-up company in Bangalore, participated in the strategizing, designing, developing and field-testing a new software package for OSS & BSS integration; this was an experience in software product line engineering;

2003 to 2010: I was a Principal Consultant to 3 major BPM, BRMS, SOA projects;

The purpose of listing these experiences here is to bring to bear relevant examples at appropriate discussions during this course!

Evaluation

The purpose of this course is to share with you whatever knowledge I have in the area of software engineering and not to evaluate you!

I am sure there will be learning and joy! But, if you miss classes and copy the project work, you will get a FAIL grade!!!

Type of Test	Weightage
Surprise quizzes + Mid-terms (no final exam)	40%
Project (monthly review)	4 * 15%
Innovative Ideas / High Quality Implementation	20%
Total	For 120 marks

Topics

- Introduction
  a. The need for SE
  b. Quality Attributes expected of software and its engineering
  c. Software Crisis of 1968
  d. Current State
  e. Software Engineering Ethics

Introduction to SE

- The need for Software Engineering

  What is software?

  What is meant by engineering?

  Why should we engineer software?

Introduction to SE

- The need for Software Engineering

  What is software?

  Computer programs and associated documentation.

  Software may be developed for a particular customer or may be developed for a general market.

  Why should documentation be part of software?

Introduction to SE

- The need for Software Engineering

  What is software?

  Computer programs and associated documentation.

  Software may be developed for a particular customer or may be developed for a general market.

  Why should documentation be part of software?

  Often software works in a specific atmosphere, that is intentionally set-up. This atmosphere includes, but not limited to,

  - The interfaces of other software / hardware in a specific type / version
  - The values in the 'Configuration Tables' or specific Database Tables

  Without such a documentation, you can't deliver s/w!

Introduction to SE

➢ The need for Software Engineering

What is software?

What is meant by engineering?

Engineering is the art or science of making practical application of the knowledge of pure sciences, as physics or chemistry, as in the construction of engines, bridges, buildings, mines, ships, and chemical plants (www.dictionary.com)

Engineering is the process of manufacturing a product using a set of well defined principles, techniques and methodologies, such that the process is repeatable (independent of humans involved) and offers certain estimated values of quality of the product being manufactured.

E.g. Pieces of 'art' are NOT the result of engineering; the automobile cars, cell phones, office furniture, medicines, OS/400 software, electric bulbs, etc are the result of engineering.

Introduction to SE

The need for Software Engineering

What is software?

What is meant by engineering?

Why should we engineer software?

Who all use software?

Many of the vertical and horizontal business segments CANNOT do business WITHOUT their critical software packages working (the incur loss to the tunes of a few millions of dollars / euros / rupees per hour). Can you name a few such business houses?

Many of the scientific research, art and social organizations critically depend upon certain software packages.

More and more of electro, electrical, mechanical, aero, chemical, nuclear, medical, pharmaceutical systems are software controlled!

Introduction to SE

- The need for Software Engineering

  What is software?

  What is meant by engineering?

  Why should we engineer software?

  Almost all age group people have started to use software!

  The economies of ALL developed nations are dependent on software.

  Is there any other engineering product that is so widely used across???

  There is a NEED to ENGINEER software!!!

Introduction to SE

- Introduction
  a. The need for SE
  - Quality Attributes expected of software and its engineering
  c. Software Crisis of 1968
  d. Current State
  e. Software Engineering Ethics

Introduction to SE

Quality Attributes expected of software

- Reliability
- Extensibility / Maintainability
- Performance
- Usability
- Security
- Availability
- Portability

Quality Attributes expected of software engineering

- Productivity
- Cost

Introduction to SE

Quality Attributes expected of software

Reliability

- Probability of failure-free operation for a specified time in a specified environment for a given purpose
- Informally, reliability is a measure of how well system users think it provides the services they require
- Not all faults are equally serious. System is perceived as more unreliable if there are serious faults (e.g. Toyota cars are considered very reliable as the probability of their engine not requiring a repair / replacement in the first 5 years is 99%; most of the other parts too have a high probability;)
- A failure corresponds to unexpected run-time behaviour observed by a user of the software
- One measure of Reliability is the MTF (mean-time-between-failures)
- OS/400 has a MTF of 1.3 years and MS Windows 5.0 had a MTF of 1.3 hours; incidentally, OS/400 is the ONLY software to have won the Malcolm-Baldrige Quality Award so far!!

Introduction to SE

Quality Attributes expected of software and its engineering

Extensibility / Maintainability

Software maintenance is the process of modifying a software system or component after delivery, to correct faults, enhance functionality, improve performances or other quality attributes, or adapt to a changed environment, etc.

Different modifications require different person-days of effort; badly analyzed / designed / coded / tested software products require more person-days of effort to fix simple, moderate and complex extensions / modifications as compared to the industry standards; modularity quotient of a software has a direct impact on the extensibility / maintainability.

One of the measures of Extensibility / Maintainability is the average person-days of effort to extend / maintain a software.

Introduction to SE

Quality Attributes expected of software and its engineering

Performance

Performance of a software is the average turn-around-time taken by the software for each logically different type of service under normal and worst-case scenarios (What are normal and worst-case business scenarios of typical banking services?)

For OLTP software, often, transactions are clustered as simple, moderate and complex and the performance is measured as the average turn-around-time for these normal, moderate and complex transactions.

For software algorithms, the performance is measured using asymptotic analysis under best and worst-case scenarios.

Performance of a software system (including its h/w and n/w environments) is measured by the average throughput it offers per unit of time

Introduction to SE

Quality Attributes expected of software and its engineering

Usability

Informally, usability is the ease with which a user of a software package can utilize its services.

Usability is a culturally sensitive subject; compare the usability of IBM Mainframe PDE (professional development environment) and UNIX software development environment.

A measure of usability is the average satisfaction rating of a suitably sampled user group.

(Note: Attempts by industry to mine texts of blogs, news-groups to get a measure for usability of their software)

Introduction to SE

Quality Attributes expected of software and its engineering

Security

Security of a software is the degree of its capability to protect its services and data from unauthorized users;

Authentication (to log-in) and authorization (to use only certain services) are used to protect the services and encryption to protect the data.

Introduction to SE

Quality Attributes expected of software and its engineering

Availability

Software packages contain many modules, which are integrated and work together; e.g Web servers, App servers, DBMS servers, application software modules, OS, N/W layers put together offer Infosys Finacle based banking services!

Availability refers to the percentage time that such as a complex software package is continuously available measured over a 100 units of time!

Many banks of reputation demand 99.9% availability for every 1000 days from the IT service provider!

Introduction to SE

Quality Attributes expected of software and its engineering

Portability

Portability of a software is the degree of ease with which one can install and use that software in different environments; for example, java based programs run on all OS that support JVM without modifying even one line of code.

Typically, software is designed in such a way that, except for one layer of that software, all other layers do NOT need any modifications, when that software is ported onto different OS or DBMS or other h/w components.

Introduction to SE

Quality Attributes expected of software engineering

Productivity

Average number of LoC per person taken over the complete life-cycle of a software project; well engineered projects eliminate or minimize re-design or re-coding or re-testing and thereby achieve high productivity; use of engineering models, methodologies, standards, tools and training of personnel etc. contribute towards high productivity;

Cost

Appropriate balance between high productivity and high cost is very essential;

High productivity and low cost is the dream of every software project manager!

Topics

- Introduction
  a. The need for SE
  b. Quality Attributes expected of software and its engineering
  - Software Crisis of 1968
  d. Current State
  e. Software Engineering Ethics

Introduction to SE

Software Crisis of 1968

During 1960's, there was so much progress in the h/w technology and production (engineering) that the business community was offered more powerful h/w boxes at lower prices;

BUT the area of SOFTWARE ENGINEERING miserably failed to meet the expansion plans! What would the business community do with the powerful h/w boxes without the suitably matching (reliable, high performance) software packages being available; many of the software projects had time and cost overruns!

The situation was declared as a "software crisis" by F. L. Bauer at the first NATO Software Engineering Conference in 1968 at Garmisch, Germany. An early use of the term is in Edsger Dijkstra's 1972 ACM Turing Award Lecture.

There has been great focus on software engineering since then!!!

Introduction to SE

Software Crisis of 1968

The problems and progress of software engineering in 1968:

1. Large (over 10K LoC) programs with 'Go To' statements and global variables existed; Dijkstra had brought in the 'Go To' less Block Structured Programming concept!
2. Re-usable functions with parameters and local variables were introduced leading to modular programs and reentrant libraries
3. Use of files for storing and retrieving data was replaced by database tables (whether hierarchical, networked or relational), which enabled locking at the table or row level, thereby paving way for concurrent use of software packages
4. Relational database concepts had lead to a scientific way to design databases (eliminated the artistic flavor of software engineers of 1970's and 1980's)

There has been great focus on software engineering since the Software Crisis event!!!

Introduction to SE

Software Crisis of 1968

The problems and progress of software engineering in 1968:

5. Software Engineering models (SSAD, OOAD, MDSE) evolved; CASE tools emerged leading to IDEs (Eclipse);
6. Object-Oriented SE was a major break-through offering encapsulation and software services, leading to major improvements in quality
7. BPM, BRMS, SOA, JMS, Web had offered the next set of improvements

Problems that we are still facing:

a) Design of program modules (as opposed to database) has still a part of art component (not desirable)

b) Lack of international standards (as compared to ECE) is coming in the way of 'Pulg-and-Play' capability of software packages

There has been great focus on software engineering since the Software Crisis event!!!

Topics

- Introduction
  a. The need for SE
  b. Quality Attributes expected of software and its engineering
  c. Software Crisis of 1968
  - Current State
  e. Software Engineering Ethics

Introduction to SE

Current State

There are at least 1 billion people in today's world who depend upon throughout 24 x 7 on cell phones, ATMs, Air and Train travel, Google, TV channels, Retail shops, electronic devices, etc, where a software package is continuously working!

Some of the software packages have 30M LoC, 30K Classes, 1K modules interacting with each other using 1K database tables, 100GB data and processing 50K transactions per day with more than 99% availability!

Still, only one software package in the last 40 years has won the Malcolm Baldrige Quality Award so far!

Next set of BIG things:

- Building software from tested building blocks – Integrating with ease a set of required services (classes) into a software package with no or minimal testing (just as we assemble compatible h/w parts into a system and start using it)!
- Use and pay of software packages with both software and data on the cloud!

Topics

- Introduction
  a. The need for SE
  b. Quality Attributes expected of software and its engineering
  c. Software Crisis of 1968
  d. Current State
- Software Engineering Ethics

Software engineering ethics

- Software engineering involves wider responsibilities than simply the application of technical skills.
- Software engineers must behave in an honest and ethically responsible way if they are to be respected as professionals.
- Ethical behaviour is more than simply upholding the law but involves following a set of principles that are morally correct.

Issues of professional responsibility

- ✧ Confidentiality
  - Engineers should normally respect the confidentiality of their employers or clients irrespective of whether or not a formal confidentiality agreement has been signed.
- ✧ Competence
  - Engineers should not misrepresent their level of competence. They should not knowingly accept work which is out with their competence.

Issues of professional responsibility

- ✧ Intellectual property rights
  - Engineers should be aware of local laws governing the use of intellectual property such as patents, copyright, etc. They should be careful to ensure that the intellectual property of employers and clients is protected.
- ✧ Computer misuse
  - Software engineers should not use their technical skills to misuse other people's computers. Computer misuse ranges from relatively trivial (game playing on an employer's machine, say) to extremely serious (dissemination of viruses).

The ACM/IEEE Code of Ethics

Software Engineering Code of Ethics and Professional Practice

ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices

PREAMBLE

The short version of the code summarizes aspirations at a high level of the abstraction; the clauses that are included in the full version give examples and details of how these aspirations change the way we act as software engineering professionals. Without the aspirations, the details can become legalistic and tedious; without the details, the aspirations can become high sounding but empty; together, the aspirations and the details form a cohesive code.

Software engineers shall commit themselves to making the analysis, specification, design, development, testing and maintenance of software a beneficial and respected profession. In accordance with their commitment to the health, safety and welfare of the public, software engineers shall adhere to the following Eight Principles:

Ethical principles

1. PUBLIC - Software engineers shall act consistently with the public interest.

2. CLIENT AND EMPLOYER - Software engineers shall act in a manner that is in the best interests of their client and employer consistent with the public interest.

3. PRODUCT - Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.

4. JUDGMENT - Software engineers shall maintain integrity and independence in their professional judgment.

5. MANAGEMENT - Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.

6. PROFESSION - Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.

7. COLLEAGUES - Software engineers shall be fair to and supportive of their colleagues.

8. SELF - Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.

Ethical dilemmas

- ✧ Disagreement in principle with the policies of senior management.
- ✧ Your employer acts in an unethical way and releases a safety-critical system without finishing the testing of the system.
- ✧ Participation in the development of military weapons systems or nuclear systems.

Topics

Introduction

a. The need for SE
b. Quality Attributes expected of software and its engineering
c. Software Crisis of 1968
d. Current State
e. Software Engineering Ethics

Summary: Key Points

1. A wide range of business, scientific and social organizations depend upon complex software packages for the continuity of their operations
2. We cannot afford another 'Software Crisis'; we need to engineer software guaranteeing high quality at reasonable cost;
3. Software Engineering is the process of manufacturing software using a set of well defined principles, techniques and methodologies, such that the process is repeatable (independent of humans involved) and offers certain estimated values of quality of the product being manufactured

Topics

Introduction

a. The need for SE
b. Quality Attributes expected of software and its engineering
c. Software Crisis of 1968
d. Current State
e. Software Engineering Ethics

Summary: Key Points

4. We need to be aware of and follow current software engineering models and research into better models
5. We need to be simultaneously ethical software engineers