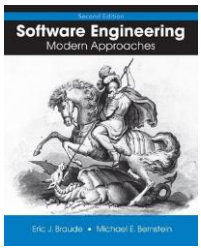
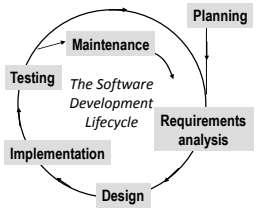
Software Engineering

Modern Approaches



Eric Braude and Michael Bernstein

Chapter 1. Introduction



- Why is software engineering important?
- Who and what does it consist of?
- What are its main activities?
- What are the principles of software engineering?
- What ethics are involved?
- What sorts of case studies will be used to illustrate the subject?

Goal of Software Engineering

- Creation of software systems that are
  - Reliable
  - Efficient
  - Maintainable
  - Meet the needs of customers
- Production of system meets
  - Schedule
  - Budget

What is Software Engineering?

- Engineering discipline
  - *the design, analysis and construction of an artifact for some **practical** purpose*
- IEEE definition:
  - *"the application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software; that is the application of engineering to software."*

NATO Study Group

- NATO Study Group on Computer Science (1968)
  - one of the first uses of the phrase *software engineering*
- "Programming management will continue to deserve its current poor reputation for cost and schedule effectiveness until such time as a more complete understanding of the program design process is achieved."

NATO Study Group (cont.)

- "Today we tend to go on for years, with tremendous investments to find that the system, which was not well understood to start with, does not work as anticipated. We build systems like the Wright brothers built airplanes — build the whole thing, push it off the cliff, let it crash, and start over again."

Software Disasters

- Numerous examples of software disasters
  - Ariane Project
  - 1990 AT&T Disaster
  - Radiation Overdose
  - The link below has a list with several disasters due to software faults

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

Software Failure

- What is it?
  - Failure to meet expectations
- What expectations are not achieved?
  - Over budget
  - Exceeds schedule and/or misses market window
  - Doesn't meet stated customer requirements
  - Lower quality than expected
  - Performance doesn't meet expectations
  - Too difficult to use

Software Failure (cont.)

- Reasons for failure:
  - Unrealistic or unarticulated project goals
  - Poor project management
  - Inaccurate estimates of needed resources
  - Badly defined system requirements
  - Poor reporting of the project's status
  - Unmanaged risks
  - Poor communication among customers, developers, and users
  - Inability to handle the project's complexity
  - Poor software design methodology
  - Wrong or inefficient set of development tools
  - Inadequate test coverage
  - Inappropriate (or lack of) software process

Software Engineering Activities

4 P's of Software Engineering

- **People**
  - *Project stakeholders*
- **Product**
  - *The software product plus associated documents*
- **Project**
  - *The activities carried out to produce the product*
- **Process**
  - *Framework within which the team carries out the activities necessary to build the product*

People

Stakeholders

- ***Business Management***
- ***Project Management***
- ***Development Team***
- ***Customers***
- ***End-Users***

The Software Product Artifacts

- Project documentation

  Documents produced during software definition and development
- Code

  Source and object
- Test documents

  Plans, cases, and results
- Customer documents

  Documents explaining how to use and operate product
- Productivity measurements

  Analyze project productivity

Project

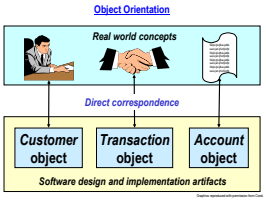
Software Project Activities

-- which produce a software product: Mainly…

- **Planning**
  - plan, monitor and control the software project
- **Requirements analysis**
  - define what to build
- **Design**
  - how to build the software
- **Implementation**
  - program the software
- **Testing**
  - validate software meets the requirements
- **Maintenance**
  - resolve problems; adapt software to meet new requirements;

Project (cont.)

- Development paradigm
  - e.g. object-oriented

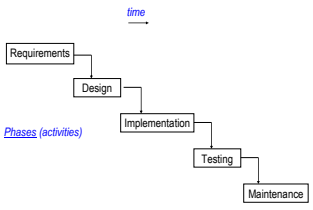


Process

- Framework for carrying out the activities of a project in an *organized* and *disciplined* manner.
- Imposes structure
- Waterfall or Iterative

Waterfall Process

The Waterfall Software Process



- Simplest process
- Sequential
- Basis for others

Iterative Process

- Software projects rarely follow ***strict*** waterfall
- Some ***iteration*** between specifications, design, implementation and test
- Requires discipline
  - e.g. update specifications when design changes

Software Engineering Principles

Software Engineering Principles

1. ***Make Quality Number 1***
2. ***High Quality Software is Possible***
3. ***Give Products to Customers Early***
4. ***Use an Appropriate Sofware Process***
5. ***Minimize Intellectual Distance***
6. ***Inspect Code***
7. ***People are the Key to Success***

Source: 201 Principles of Software Engineering, Alan Davis

Software Engineering Ethics

- Most disciplines operate under a strict set of *ethical* standards
- The *Merriam-Webster* online dictionary defines *ethics* as:
  - **1**: the discipline dealing with what is good and bad and with moral duty and obligation
  - **2:** a set of moral principles

Software Engineering Ethics (cont.)

- ACM/IEEE-CS Joint Task Force - *Software Engineering Code of Ethics and Professional Practices* (Version 5.1):
  - "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:

A Portuguese version can be found here:
https://www.computer.org/cms/Computer.org/professional-education/pdf/doc.pdf

Software Engineering Ethics (cont.)

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:

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.

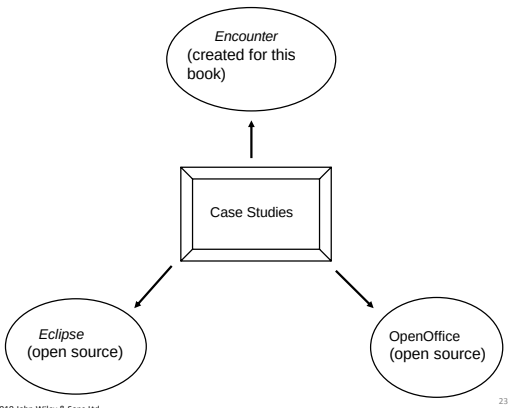
4. JUDGMENT - Software engineers shall maintain integrity and independence in their professional judgment.

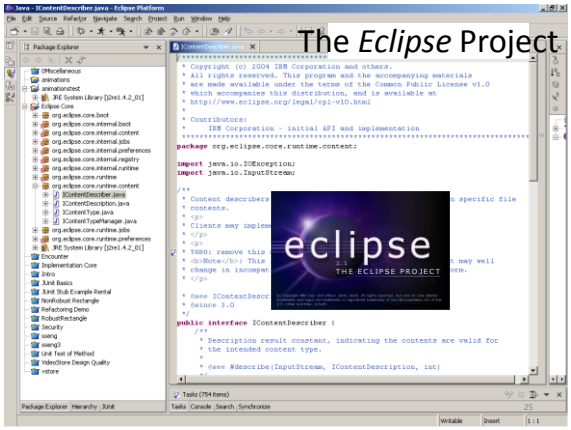
5. MANAGEMENT - Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.

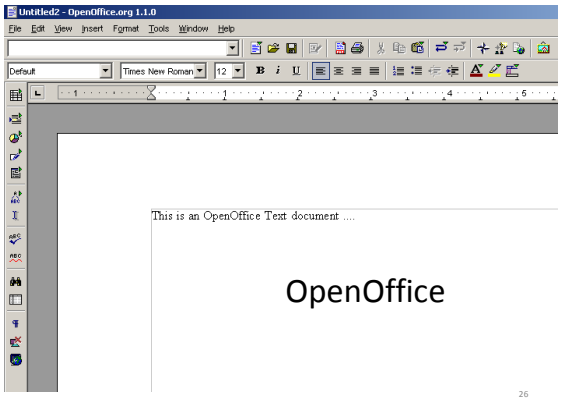
6. PROFESSION - Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.

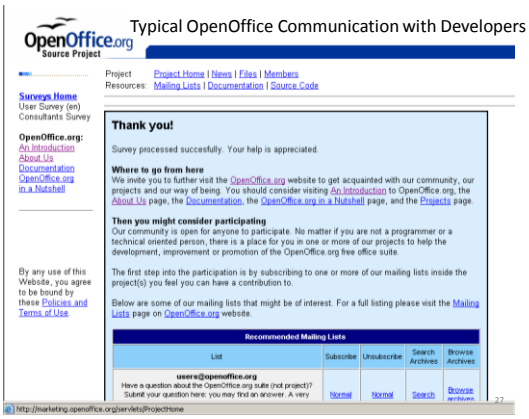
7. COLLEAGUES - Software engineers shall be fair to and supportive of their colleagues.

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.









Project	Lead/Co-Lead	Short Name	Description
API	Michael Hoennig, Juergen Schmidt	api	The application programming interface.
Application Framework	Mathias Bauer, Carsten Driesner	framework	The framework for applications.
Build Tools and Environment	Martin Hollmichel	tools	The tools used in build process and the build environment.
Database Access	Frank Schoenheit, Dirk Grobler	dba	The database access for the applications.
Documentation	Scott Carr	documentation	End-user documentation for the various components making up OpenOffice.org.
External	Martin Hollmichel (acting)	external	This project will host all the external code allowed.
Graphic Applications	Kai Ahrens, Christian Lippka	graphics	The graphic applications such as Draw and Impress.
Graphic System Layer	Christof Pintaske, Hennes Rohling	gsl	The Visual Class Library and other modules.
Installation	Dirk Voelzke	installation	Creating the installation set.
Linguicomponent	Kevin Hendricks	lingucomponent	Creating dictionaries, thesauri, and other related tools.

OpenOffice Accepted Projects

http://projects.openoffice.org/accepted.html

Localization	Dieter Loeschky, Nils Fuhrmann Pavel Janik	l10n	Localization. This project includes Internationalization (i18n).
Marketing	Jacqueline McNally	marketing	The project furthering the growth and use of OpenOffice.org technology. Efforts include: developing collateral, logos, public outreach.
Porting	Martin Hollmichel, Kevin Hendricks	porting	Porting to new platforms.
Quality Assurance	Michael Bemmer, Scott Carr, Gordon Shum	qa	Quality Assurance: testing and qualifying all builds of OpenOffice.org.
Spreadsheet	Niklas Nebel, Eike Rathke	sc	The spreadsheet application.
Universal Content Broker	Matthias Huetsch, Andreas Bille	ucb	Allows the applications to transparently access content with different structures.
UNO Development Kit / Component Technology	Kay Ramme, Kai Sommerfeld	udk	Object model development and component technology. Includes the old OI and Scripting projects.
User Interface	Oliver Specht	ui	Common user interface for OpenOffice.org applications.
Utilities	Hennes Rohling	util	Utilities used in development.
Website	Louis Suárez-Potts, Kay Schenk,	www	The OpenOffice.org website; the project for establishing the appearance of the Project.
Word Processing	Andreas Martens, Caolan McNamara	sw	The Word Processing Application

OpenOffice Accepted Projects 2

http://projects.openoffice.org/accepted.html