Engenharia de Software

2º Semestre 2016

Cronograma

	Data	Conteúdo
1	01-ago-16	Software e Engenharia de Software
2	08-ago-16	Conceitos Básicos de Orientação a Objeto
	15-ago-16	Feriado
3	22-ago-16	Herança e Polimorfismo
4	29-ago-16	Diagrama de classes
	05-set-16	Semana da Patria - Não há aula
5	12-set-16	Casos de Uso
6	19-set-16	Engenharia de Requisitos
7	26-set-16	Prova 1
8	03-out-16	Generalização, agregação
9	10-out-16	Processo para diagramas de desenvolvimento
10	17-out-16	General Hierarchy; Player-Role; Singleton; Observer;
11	24-out-16	Adapter; Facade; etc.
12	31-out-16	Design de Interfaces de Usuários
13	07-nov-16	Principios de Design
	14-nov-16	Recesso - Não há aula
14	21-nov-16	Process Models, Cost Estimation, Teams
15	28-nov-16	Prova 2
16	05-dez-16	Sub

Avaliações

2 Provas – P1, P2

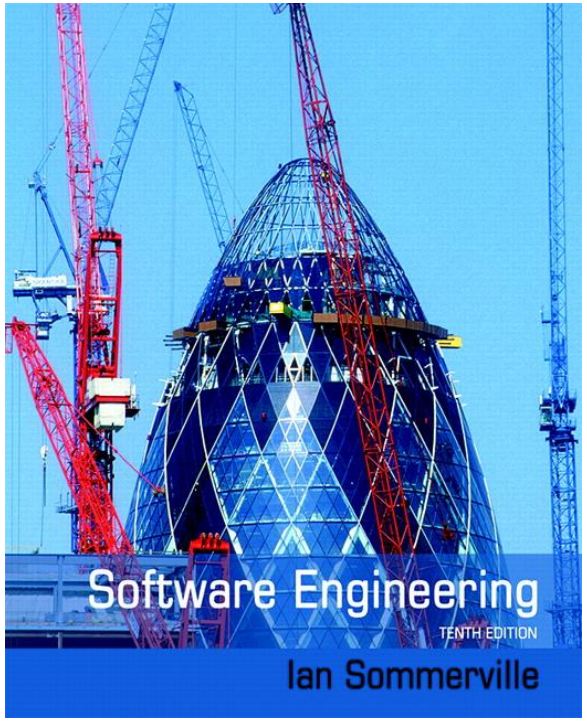
1 Prova Substitutiva - S

6 Trabalhos práticos – TP1, TP2, TP3, TP4, TP5, TP6

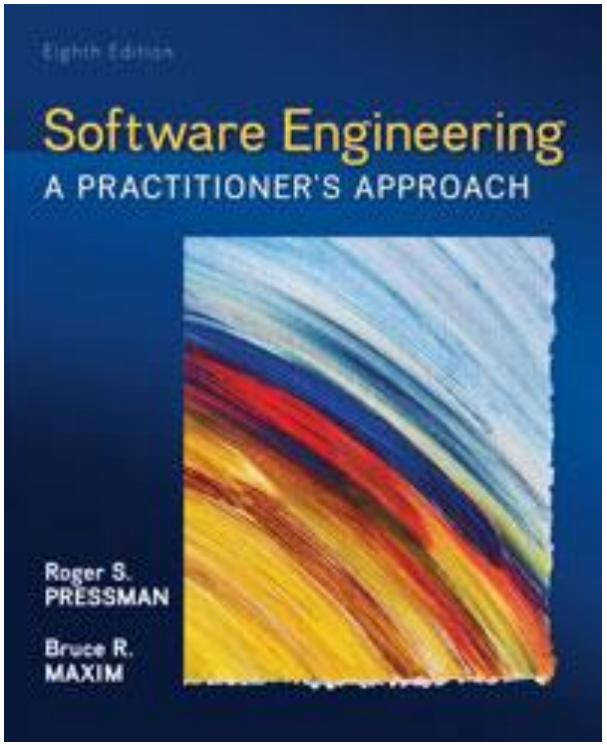
Tarefas ao final das aulas – T(i)

Média = a s d

Bibliografia



Software Engineering, 10th Edition
Ian Sommerville, University of St Andrews, Scotland
©2016 Pearson



Software Engineering: A Practitioner's Approach, 8/e
Roger S Pressman, R.S. Pressman & Associates, Inc
Bruce R. Maxim, University of Michigan
ISBN: 0078022126
Copyright year: 2015

1.1 The Nature of Software...

Software is intangible

- Hard to understand development effort

Software is easy to reproduce

- Cost is in its *development*

  —in other engineering products, manufacturing is the costly stage

The industry is labor-intensive

- Hard to automate

The Nature of Software ...

Untrained people can hack something together

- Quality problems are hard to notice

Software is easy to modify

- People make changes without fully understanding it

Software does not 'wear out'

- It *deteriorates* by having its design changed:

  —erroneously, or

  —in ways that were not anticipated, thus making it complex

The Nature of Software

Conclusions

- Much software has poor design and is getting worse
- We have to learn to ‘engineer’ software

Some types of Software

Real time embedded software

- E.g. control and monitoring systems
- Must react immediately
- Safety often a concern

Data processing software

- Used to run businesses
- Accuracy and security of data are key

Game software

Mobile device software

Web-based software

Etc.

1.2 What is Software Engineering?...

The process of solving customers' problems by the systematic development and evolution of large, high-quality software systems within cost, time and other constraints

What is Software Engineering?…

Solving customers' problems

- The goal
- Sometimes the solution is to *buy, not build*
- Adding unnecessary features often makes software worse
- Software engineers must *communicate effectively* to identify and understand the problem

What is Software Engineering?…

Systematic development and evolution

- An engineering process involves applying *well understood techniques* in a organized and *disciplined* way
- Many well-accepted practices have been formally standardized

  —e.g. by the IEEE or ISO
- Most development work is *evolution*

What is Software Engineering?…

Large, high quality software systems

- Software engineering techniques are needed because large systems *cannot be completely understood* by one person
- Teamwork and co-ordination are required
- Key challenge: Dividing up the work and ensuring that the parts of the system work properly together
- The end-product must be of sufficient quality

What is Software Engineering?

Cost, time and other constraints

- Finite resources
- The benefit must outweigh the cost
- Others are competing to do the job cheaper and faster
- Inaccurate estimates of cost and time have caused many project failures

Top Hat Monocle Question: SE Objectives

1.3 The Software Engineering Profession

The term Software Engineering was coined in 1968

- People began to realize that the principles of engineering should be applied to software development

Engineering is a licensed profession

- In order to protect the public
- Engineers design artifacts following well accepted practices which involve the application of science, mathematics and economics
- Ethical practice is also a key tenet of the profession

In many countries, much software engineering does not require an engineering licence, but is still engineering

Software Engineering Code of Ethics

Software engineers shall

- Act consistently with public interest
- Act in the best interests of their clients
- Develop and maintain with the highest standards possible
- Maintain integrity and independence
- Promote an ethical approach in management
- Advance the integrity and reputation of the profession
- Be fair and supportive to colleagues
- Participate in lifelong learning

1.5 Software Quality...

Usability

- Users can learn it and fast and get their job done easily

Efficiency

- It doesn't waste resources such as CPU time and memory

Reliability

- It does what it is required to do without failing

Maintainability

- It can be easily changed

Reusability

- Its parts can be used in other projects, so reprogramming is not needed

Software Quality and Stakeholders

Customer (those who pay):
solves problems at
an acceptable cost in
terms of money paid and
resources used

User:
easy to learn;
efficient to use;
helps get work done

Developer:
easy to design;
easy to maintain;
easy to reuse its parts

Development manager:
sells more and
pleases customers
while costing less
to develop and maintain

Software Quality: Conflicts and Objectives

The different qualities can conflict

- Increasing efficiency can reduce maintainability or reusability
- Increasing usability can reduce efficiency

Setting objectives for quality is a key engineering activity

- You then design to meet the objectives
- Avoids ‘over-engineering’ which wastes money

1.6 Software Engineering Projects

Most projects are *evolutionary* or *maintenance* projects, involving work on *legacy* systems

- Corrective projects: fixing defects
- Adaptive projects: changing the system in response to changes in
  - —Operating system
  - —Database
  - —Rules and regulations
- Enhancement projects: adding new features for users
- Reengineering or perfective projects: changing the system internally so it is more maintainable

Software Engineering Projects

‘Green field’ projects

- New development
- The minority of projects

1.7 Activities Common to Software Projects...

Requirements and specification

- Includes
  - —Domain analysis
  - —Defining the problem
  - —Requirements gathering
    - Obtaining input from as many sources as possible
  - —Requirements analysis
    - Organizing the information
  - —Requirements specification
    - Writing detailed instructions about how the software should behave

Activities Common to Software Projects...

Design

- Deciding how the requirements should be implemented, using the available technology
- Includes:
  - —*Systems engineering*: Deciding what should be in hardware and what in software
  - —*Software architecture*: Dividing the system into subsystems and deciding how the subsystems will interact
  - —*Detailed design* of the internals of a subsystem
  - —*User interface design*
  - —*Design of databases*

Activities Common to Software Projects

Modeling

- Creating representations of the domain or the software
  - —Use case modeling
  - —Structural modeling
  - —Dynamic and behavioural modeling

Programming

Quality assurance

- Reviews and inspections
- Testing

Deployment

Managing the process

Tarefa 1 – T(1)

1. **Leitura do Cap. 1 – Introdução do livro de ES - Sommerville.**
2. **Defina processo de software.**
3. **O que é CASE?**
4. **Quais são os quatro principais atributos que todos os produtos de software devem ter? Sugira quatro outros atributos que podem ser significativos.**