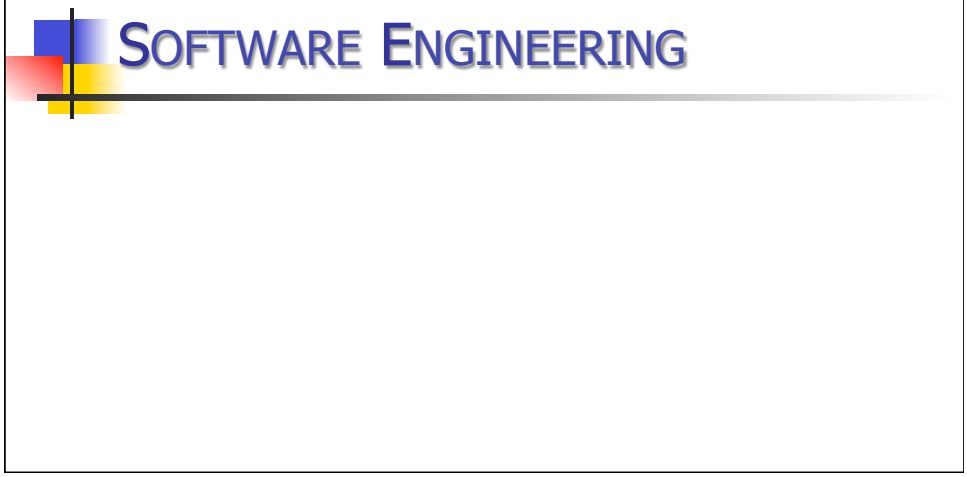
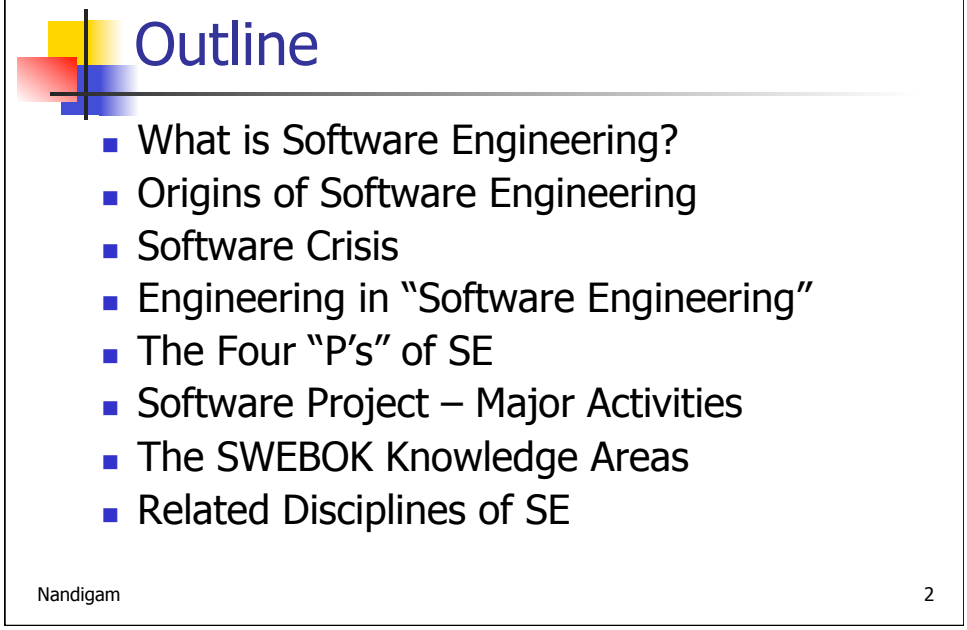

SOFTWARE ENGINEERING


Outline
What is Software Engineering?
Origins of Software Engineering
Software Crisis
Engineering in "Software Engineering"
The Four "P's" of SE
Software Project – Major Activities
The SWEBOK Knowledge Areas
Related Disciplines of SE
Nandigam
2

What is Software Engineering?

- The IEEE defines Software Engineering as:

1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software, that is, application of engineering to software.

2) The study of approaches as in 1)

Origins of Software Engineering?

- NATO Software Engineering Conference, Garmisch, Germany, 1968.
  - First users of the phrase "Software Engineering"
  - Deliberate and provocative use of the term in the title for the conference.
  - Participants met to discuss the seriousness of the problems with software development (some willing to term this "Software Crisis")

Software Crisis

- Practices that resulted in software
  - delivered late
  - over budget
  - doesn't meet stated customer requirements
  - buggy and unreliable
  - lower quality
  - performance doesn't meet expectations
  - difficult to use and/or maintain

Software Crisis – Major Cause

- Edsger W. Dijkstra, ACM Turing Award Lecture, 1972

  "The major cause of the software crisis is that the machines have become several orders of magnitude more powerful! To put it quite bluntly: as long as there were no machines, programming was no problem at all; when we had a few weak computers, programming became a mild problem, and now we have gigantic computers, programming has become an equally gigantic problem."

Software Crisis – Common Causes

- Unrealistic or unanticipated 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 stakeholders
- Inability to handle the project's complexity
- Sloppy development practices
- Inappropriate (or lack of) software process
- Stakeholder politics

Reading: "Why Software Fails" by Robert Charette, IEEE Spectrum, September 2005.

Standish Group Chaos Reports

- Standish Group – A research firm that analyzes software development projects.

Standish project benchmarks over the years

Year	Successful (%)	Challenged (%)	Failed (%)
1994	16	53	31
1996	27	33	40
1998	26	46	28
2000	28	49	23
2004	29	53	18
2006	35	46	19
2009	32	44	24

Standish Group Chaos Reports

- Project Successful
  - project completed on time and on budget
  - offered all features and functions as initially specified.
- Project Challenged
  - project completed and operational but over budget and over the time estimate
  - offered fewer features and functions than originally specified.
- Project Failed
  - project cancelled at some point during the development cycle
- Note: Others have questioned reports by Standish Group
  - "The Rise and Fall of the Chaos Report Figures" by J. Laurenz Eveleens and Chris Verhoef, January/February 2010 IEEE Software

Why Engineering in "Software Engineering"?

- Think of engineering disciplines such as civil engineering…
- How are bridges and buildings built?
  - Start from theoretical basis
  - Use sound and proven design and construction techniques
- Shouldn't it be possible to build software similar to other engineering fields?

The Four "P's" of Software Engineering

- Production of large software systems can be extremely complex and challenging.
- PEOPLE
  - Project stakeholders
  - Business, project, development, customer, and end users
- PRODUCT
  - Software product (source and object code)
  - Associate documents
- PROJECT
  - Activities carried out to produce the product
- PROCESS
  - Framework for carrying out the activities of a project in an organized and disciplined manner.
  - Software Process Model - a specific implementation of a software process.

Major Activities of a Software Project

- Planning
  - Plan, monitor, and control the software project
- Requirements analysis
  - Define what to build
- Design
  - Describe how to build the software
- Implementation
  - Program the software
- Testing
  - Verify and validate the software
- Maintenance
  - Corrective, adaptive, perfective, preventive

The SWEBOK Knowledge Areas

- Knowledge Areas (KAs) as per "Guide to the Software Engineering Body of Knowledge (SWEBOK), 2004"
  - Software requirements
  - Software design
  - Software construction
  - Software testing
  - Software maintenance
  - Software configuration management
  - Software engineering management
  - Software engineering process
  - Software engineering tools and methods
  - Software quality

Related Disciplines of Software Engineering

- Disciplines with which software engineering shares a common boundary (per SWEBOK):
  - Computer engineering
  - Computer science
  - Management
  - Mathematics
  - Project management
  - Quality management
  - Software ergonomics
  - Systems engineering