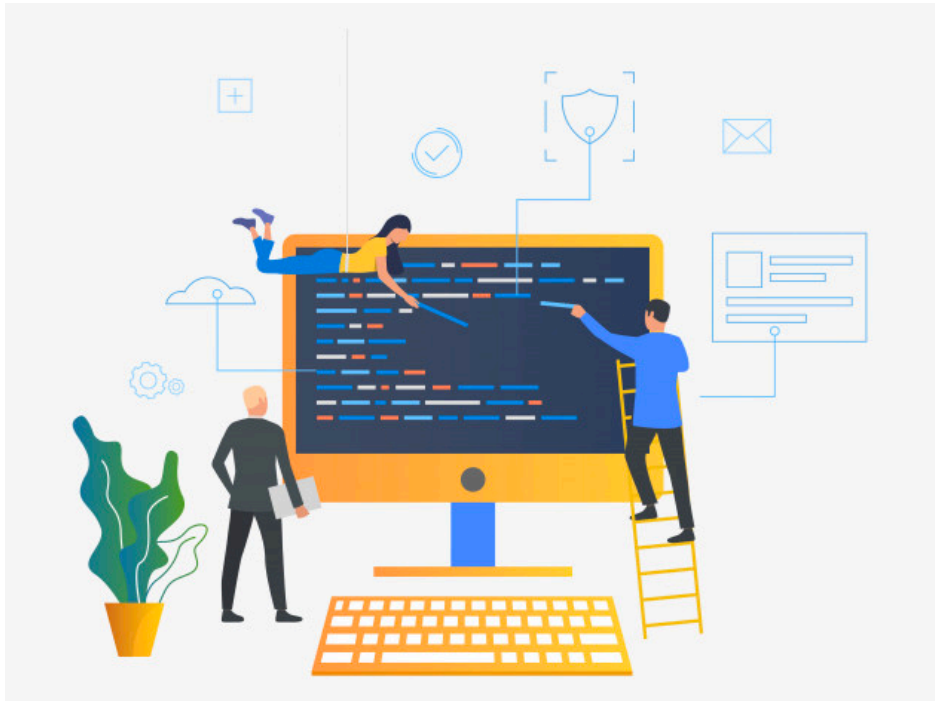
Lecture 1



Software Engineering – ADipIT01

Introduction to Software Engineering

Lecturer and Tutors

- MR. Nirmal Thapa
  - Course Leader – IT (HCK)
  - MBA, London Metropolitan University
  - BSC. IT (Hons), London Metropolitan University
- Expertise
  - Software Engineering, System Analyst,
  - Key Field: Web Application Development, E-Commerce System, QA (UI, UX)

Study Format

- Lecture hours and workshop hours
  - Lecture: 2 Hours
  - Workshop: 2.5 Hours
- Workshop followed by Lecture

Assessment Detail

- 50% coursework (Submission: Week 13)
- 50% Unseen Examination (Week 14)

Lectures	Topics
1	Introduction to Software Engineering
2	Software Development Process (SDLC Activities) - SDLC Activity: Specification or Requirement Engineering (Week 3) - SDLC Activity: System Modeling/Design (Week 4, 5, 6) - SDLC Activity: Implementation (Week 7, 8, 9) - SDLC Activity: Testing (Week 10) - SDLC Activity: Evolution (--) - SDLC Activity: Deployment/Installation (Week 11) - SDLC Activity: Maintenance (week. 12)
3	SDLC Activity: Requirement Engineering - Requirement Elicitation - Requirement Analysis and Management - Requirement Validation
4, 5, 6	SDLC Activity: System Modeling/Design - Context Modeling - Data Modeling - Structural/Architectural Modeling - Process Modeling - UI/UX Modeling
7,8,9	SDLC Activity: Implementation (Coding, tools, GIT – Version management, IDE, RESTFUL architecture)
10	SDLC Activity: Testing (Old slides)
11	SDLC Activity: Deployment (tools to deploy, cloud computing)
12	SDLC Activity: Maintenance (old slides)

Agenda

- What is Software Engineering
- Software Engineering Diversity
- Types of Software (In terms of product and application)
- Causes of Software Project Failure

1.0 Introduction to Software Engineering

1.1 What is software Engineering

Software engineering is concerned with the following aspects for professional software development.

1. Theories
2. Methods
3. Tools

1.0 Introduction to Software Engineering

Theories	Methods	Tools
• Types of requirements • System Modeling • System design patterns • Testing approaches	• Software development processes (Generic SDLC) • Software Development approaches • Software development frameworks • Software Management	• GIT • IDEs • Deployment Tools • Testing Tools

1.0 Introduction to Software Engineering

In Summary,

- Software engineering is an engineering discipline that is concerned with **all aspects of software production from the early stages** of system specification through to maintaining the system after it has gone into use

1.2 Software Engineering Diversity

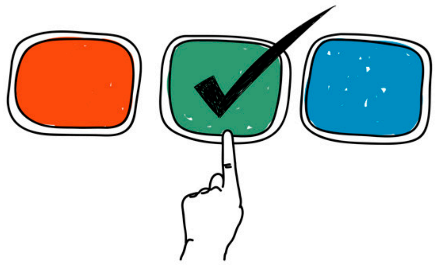
What are the best software Engineering techniques and methods?

While all software projects have to be professionally managed and developed, different techniques are appropriate for different types of system.

1.2 Software Engineering Diversity

For example,

- games should always be developed using a series of prototypes *(Probably, Prototype method would be best)*
- safety critical control systems require a complete and analyzable specification to be developed. *(Probably DSDM method would be best)*



Hence:

You can’t, say that one method is better than another

1.3 Types of Software (in terms of product)

1. Generic / General Purpose Product

2. Customized Product

Note:

- These different types of software may result in different architecture based on future scalability, security -
- General purpose software might need to be more modular for future feature, it might need to run in multiple platforms and suitable architecture might need to be planned during software engineering)

1.3 Types of Software (in terms of product)

1. Generic / General Purpose Product

   - Example: word, accounting software, CAD software

1.3 Types of Software (in terms of product)

2. Customized Product

- Software that is commissioned by a **specific customer to meet their own needs.**
- Example: Embedded control system, air traffic control software, traffic monitoring systems

1.4 Types of Software (in terms of applications)

1. Stand Alone Application
2. Interactive Based Transaction based Application
3. Embedded Control System
4. Entertainment System
5. System for Modeling and Simulation
6. Systems of Systems

1.4 Types of Software (in terms of applications)

1. Stand-alone application

- Do not need to be connected to a network

1.4 Types of Software (in terms of applications)

2. Interactive Transaction based applications

Applications that execute on a remote computer and are accessed by users from their own PCs or terminals (Web Applications)

Example: Zoho Book, Google Products

1.4 Types of Software (in terms of applications)

3. Embedded Control systems

Control, manage and monitor hardware devices

1.4 Types of Software (in terms of applications)

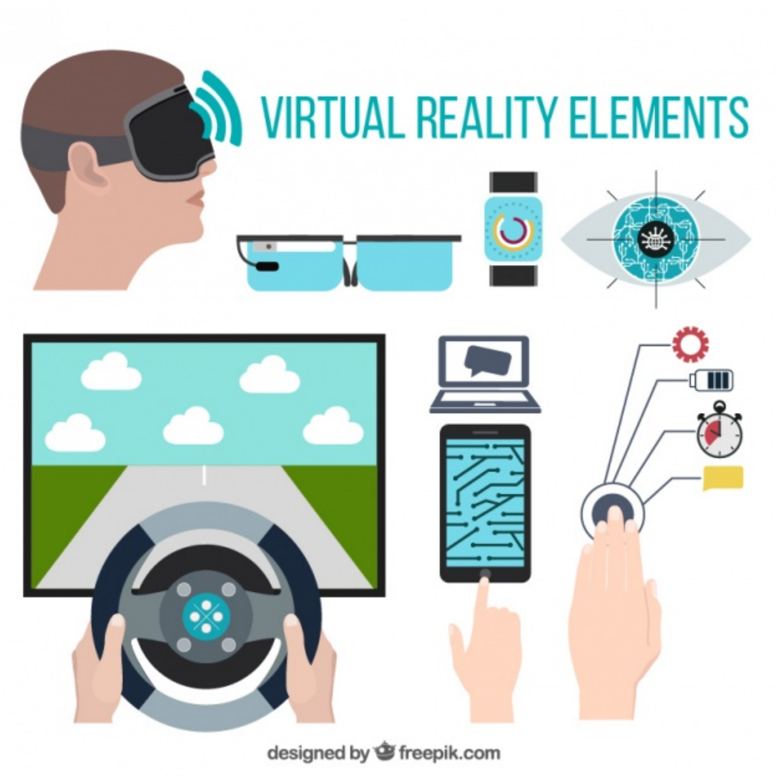
4. Entertainment Systems

Gaming, Music, Video

1.4 Types of Software (in terms of applications)

5. Systems for modeling and simulation

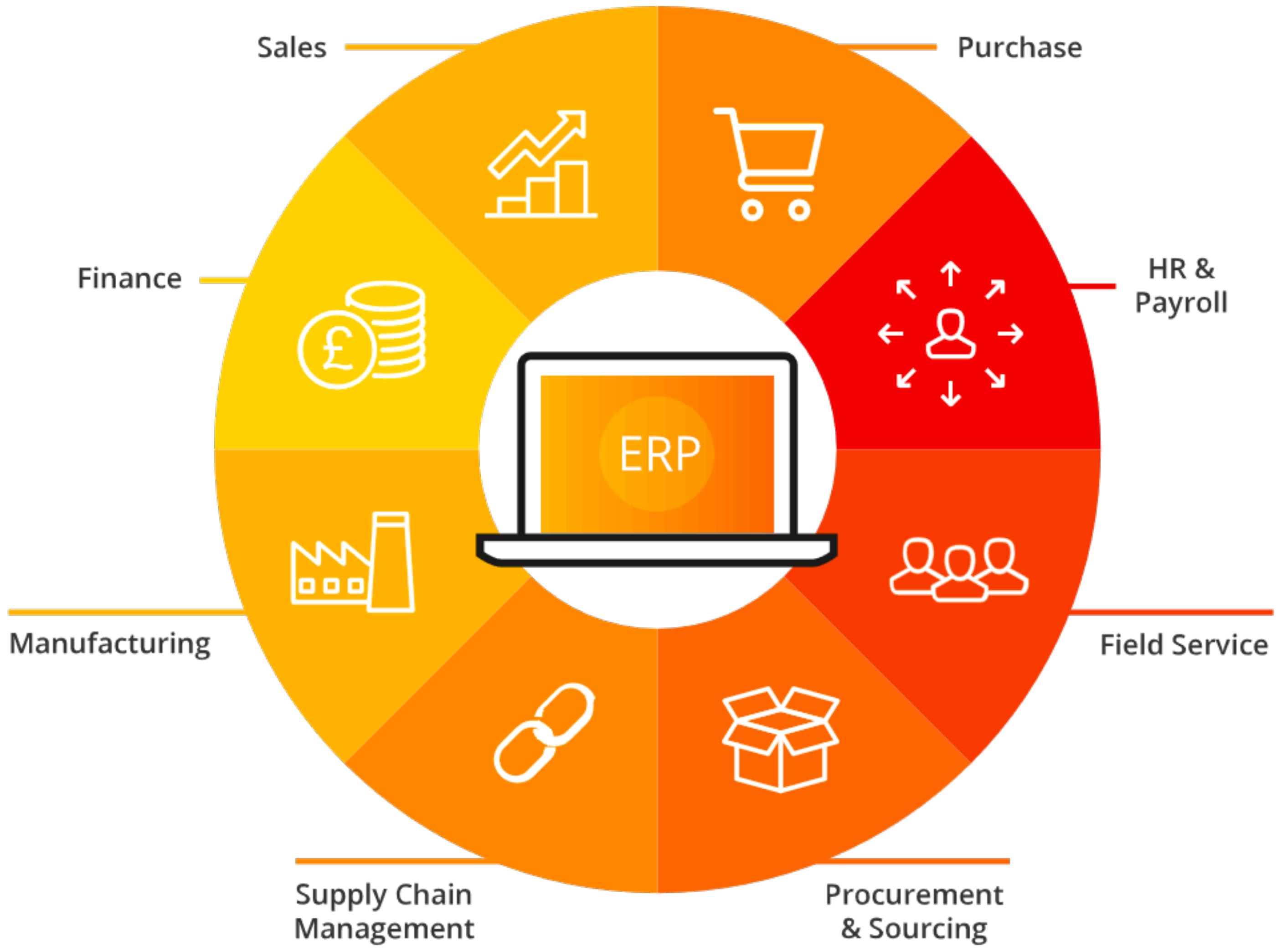
Simulator software (Flight Simulator)



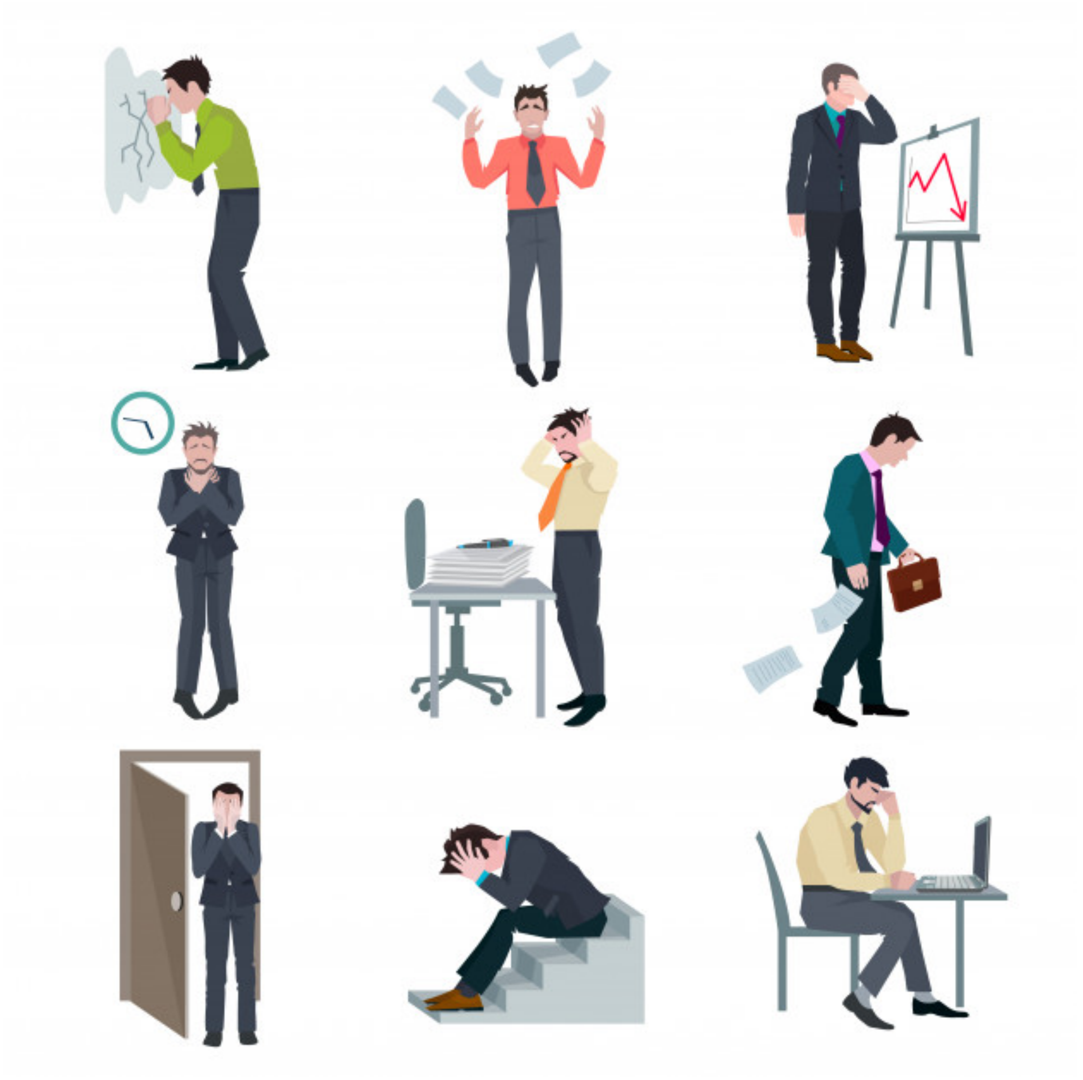
1.4 Types of Software (in terms of applications)

6. Systems of systems

ERP-Enterprise Resource Planning System (Large integrated System), system consist of large independent sub systems.



1.5 Cause of Software Project Failure



We need the understanding of software engineering to avoid software project failure. The reason behind increasingly failure of software project are as follows:

1. Increasing system complexity
2. Failure to use software engineering methods and processes

1.5 Cause of Software Project Failure

1. Increasing system complexity

As new software engineering techniques help us to build larger, more complex systems, the demands change. Systems have to be built and delivered more quickly. larger, even more complex systems are required. systems have to have new capabilities that were previously thought to be impossible.

1.5 Cause of Software Project Failure

2. Failure to use software engineering methods

It is fairly easy to write computer programs without using software engineering methods and techniques. Many companies have drifted into software development as their products and services have evolved. They do not use software engineering methods in their everyday work. Consequently, their software is often more expensive and less reliable than it should be.

Thank you