# ADVANCED SOFTWARE ENGINEERING

CSE 805

MSC

BY

Dr. Chinwe Peace Igiri

Course Content

| S/N | Topic | Description |
|---|---|---|
| 1 | Introduction | Software engineering and its place as an engineering discipline. |
| 2 | Life cycle of software system | Requirements analysis, development, operation and maintenance. |
| 3 | Software metrics | Portability, Re-usability, Correctness, Reliability, Efficiency, Usability, Integrity, Maintainability and Flexibility. |
| 4 | Software quality and testing. | Software quality and testing. |
| 5 | Software architecture | Architecture description languages, pattern-oriented software architecture, component-based development, distributed software architecture using middleware, enterprise application integration, architecture for mobile and pervasive systems and model driven architecture. |
| 6 | Advanced modeling | UML extension mechanisms, object constraint language and model checking. |
| 7 | Software project management | Study of interpersonal process decision making styles, problem solving concepts and procedure, creative effort, conflict resolution, leadership and assessment. Concepts of motivation, team work and group dynamics. |
| 8 | Software engineering and law | Intellectual property law, professional ethics and code of conduct. Patents, trademarks, copyright, trade secrets, privacy and confidentiality , contracts and licensing, government regulations, global legal issues including Internet law and cyber crime. |
| 9 | Overview of Open Source Software. | |

# INTRODUCTION:
# SOFTWARE ENGINEERING AS AN ENGINEERING DISCIPLINE

**Software?**

Collection of instruction and data that tells the computer what to do

**Engineering?**

Application of scientific principles to design and build a product

**S/E?**

Application of theories, methods and tools to design and build a cost-effective, efficient and quality software product

# Software Engineer’s responsibilities

**Users**

Lack knowledge of their expectation from the software
Only interested in simplicity of operation and response time

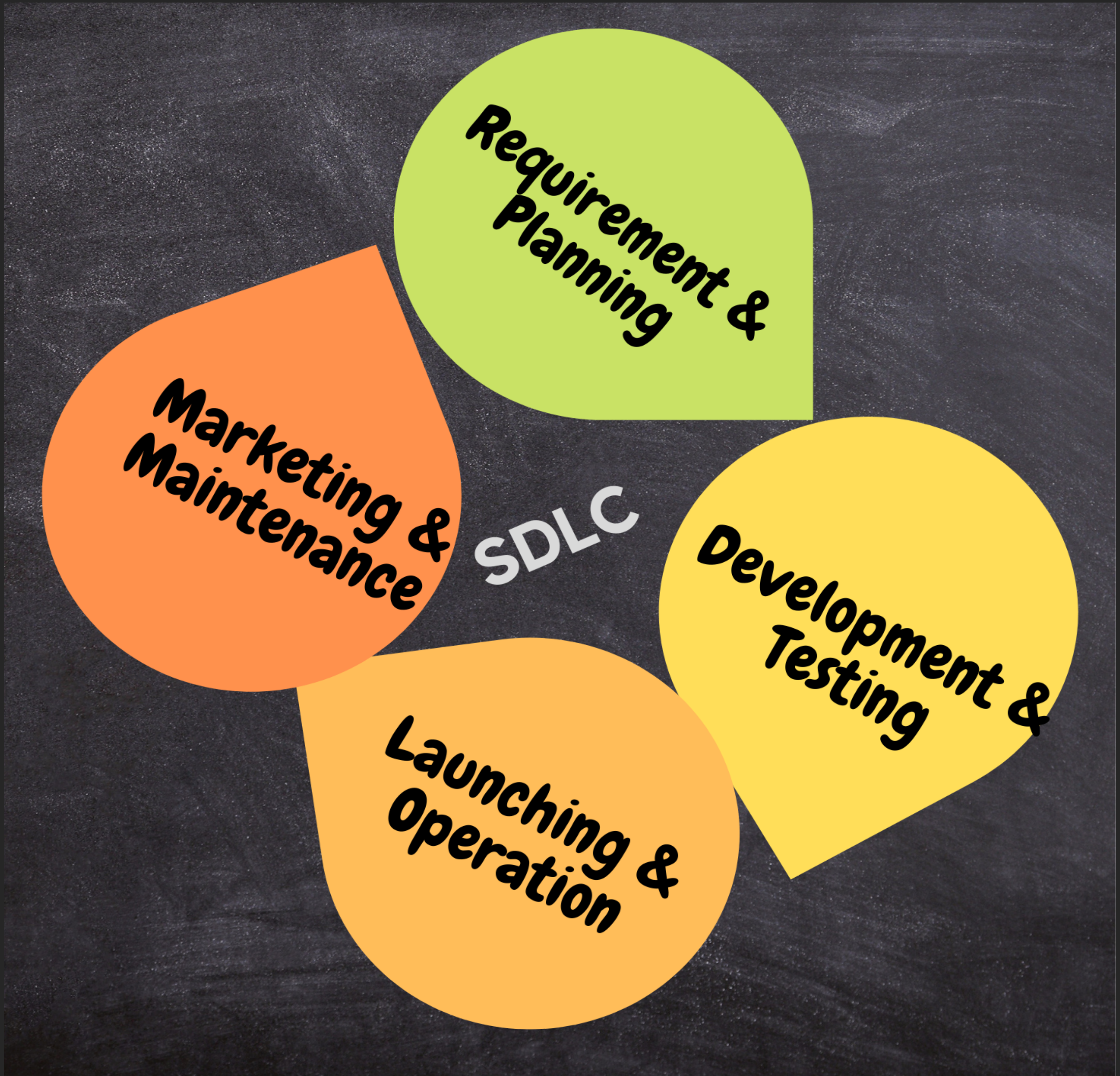
**Technical people**

Interested in functionalities and data base

**Management**

Interested in profit and return on investment

# SDLC


Requirement &
Planning
Development &
Testing
Launching &
Operation
Marketing &
Maintenance
SDLC

# SDLC: REQUIREMENT ANALYSIS AND PLANNING

- **Problem identification**
- **Idea generation –** *Elevatr, milanote, Squadhelp, Stache*
- **Planning and feasibility study**
- **Market research and analysis – gather information form client and their preferences –** *AYTM, Proved, google trends, CrunchBase, Statista*
- **Naming -** *Club, Squadhelp,, NameChk, Naminum, Name Robot*
- **Mockup & Wireframe –** *Lucidchat, mockingbird, mockup, Balsamiq, etc.*

# SDLC: DEVELOPMENT AND TESTING

- Designing – Agile methodology for instance use tools like:
  - JIRA Agile
  - Agile manager
  - Active Collab
  - Agile Bench
- Writing the source code and documentation
- Testing – testing is a process to find and fixed bugs.
  - Selenium
  - Testing Whiz
  - Test Complete

# SDLC: LAUNCHING AND OPERATION

- Launching is bring the software to the market
  - Publicize
  - PRX
  - BuzzSumo
  - JustReachOut
  - PressFarm
  - Hey Press

Operation – using the product

# SDLC: MAINTENANCE AND MARKETING

**What is it?**

The process of changing, modifying, and updating software to keep up with customer needs. It is usually done after the product has launched for some  reasons:

- improving the software overall,
- correcting issues or bugs,
- to boost performance,
- Remain competitive and relevant.

# SDLC: TYPES of MAINTENANCE

- Corrective Software Maintenance
- Preventative Software Maintenance
- Perfective Software Maintenance
- Adaptive Software Maintenance

# SDLC: CORRECTIVE MAINTENANCE

Corrective software maintenance is the typical, classic form of maintenance (for software and anything else for that matter). Corrective software maintenance is necessary when something goes wrong in a piece of software including faults and errors. These can have a widespread impact on the functionality of the software in general and therefore must be addressed as quickly as possible.

# SDLC: PREVENTIVE MAINTENANCE

Preventative software maintenance is looking into the future so that your software can keep working as desired for as long as possible. This includes making necessary changes, upgrades, adaptations and more.

Preventative software maintenance may address small issues which at the given time may lack significance but may turn into larger problems in the future. These are called latent faults which need to be detected and corrected to make sure that they won’t turn into effective faults.

# SDLC: PERFECTIVE MAINTENANCE

As with any product on the market, once the software is released to the public, new issues and ideas come to the surface. Users may see the need for new features or requirements that they would like to see in the software to make it the best tool available for their needs. This is when perfective software maintenance comes into play.

Perfective software maintenance aims to adjust software by adding new features as necessary and removing features that are irrelevant or not effective in the given software. This process keeps software relevant as the market, and user needs changes.

# SDLC: ADAPTIVE MAINTENANCE

Adaptive software maintenance has to do with the changing technologies as well as policies and rules regarding your software.

These include operating system changes, cloud storage, hardware, etc. When these changes are performed, your software must adapt in order to properly meet new requirements and continue to run well.

# SDLC: MAINTENANCE TOOLS

1. Bugzilla: it is defect tracking system
2. HP Quality Center : Web based
3. IBM Rational Quality Manager: Web based
4. Micro Focus SilkPerformer