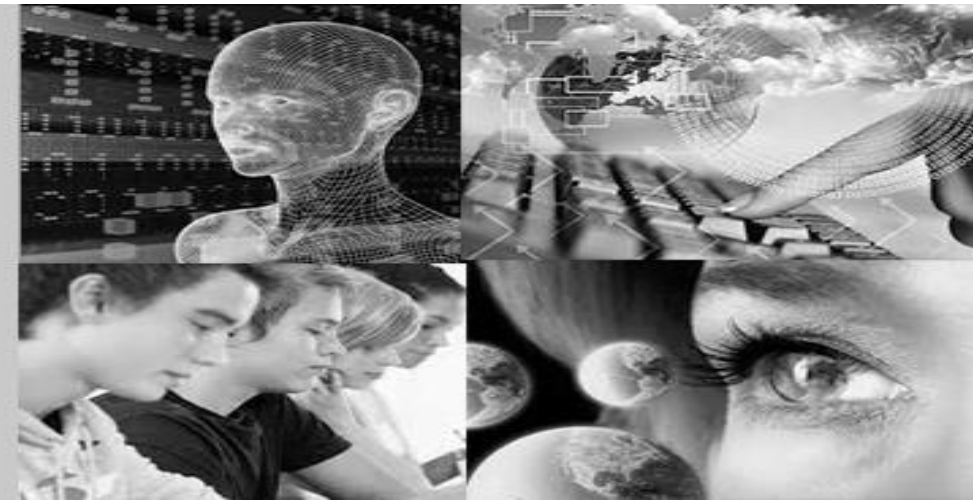
Software Engineering
(CSM 31212)

Lecture 12:Emerging Trends in SE



MM. Mohamed Mufassirin

Lecturer in Computer Science

Dept. of Computer Science

FAS/SEUSL

Objective:

- Adapting to Innovations and Design Patterns

Introduction

What are Emerging Trends?

Innovations shaping software engineering practices.

Driven by advancements in technology and changing user demands.

Why Are They Important?

Enhance efficiency, scalability, and maintainability of systems.

Equip engineers to meet modern challenges.

Key Emerging Trends in Software Engineering

- **Artificial Intelligence (AI) and Machine Learning (ML):**
  - Incorporating AI in software development tools for automation.
  - Examples: Bug prediction, code review automation.
- **DevOps and Continuous Delivery:**
  - Bridging development and operations for seamless delivery pipelines.
  - Tools: Jenkins, Docker, Kubernetes.
- **Low-Code and No-Code Platforms:**
  - Reducing coding complexity with drag-and-drop interfaces.
  - Suitable for rapid prototyping and business applications.

Key Emerging Trends in SE (Cont…)

Microservices Architecture:
•Decomposing applications into small, independent services.
•Ensures scalability and fault tolerance.

Cloud-Native Development:
•Designing applications specifically for cloud environments.
•Benefits: Scalability, cost efficiency, and resilience.

Cybersecurity Practices:
Embedding security within every phase of software development.
Approach: DevSecOps

Edge Computing:
Processing data closer to its source rather than in centralized data centers. Key for IoT and real-time applications.

The Role of Design Patterns in Modern Software Engineering

- **What Are Design Patterns?**
  - Proven solutions to common design problems.
  - Serve as blueprints for building robust software systems.
- **Importance:**
  - Promote code reuse, readability, and scalability.
  - Help standardize best practices across teams.

Commonly Used Design Patterns

1.Creational Patterns:

- Purpose: Deal with object creation mechanisms.
- Examples: Singleton, Factory Method, Abstract Factory.

2.Structural Patterns:

- Purpose: Define relationships between entities to form larger structures.
- Examples: Adapter, Composite, Proxy.

3.Behavioral Patterns:

- Purpose: Handle object communication and responsibility delegation.
- Examples: Observer, Strategy, Command.

Emerging Design Patterns for Modern Trends

1.Event-Driven Architecture Patterns:

1. Designed for asynchronous communication in distributed systems.
2. Example: Event Sourcing, CQRS (Command Query Responsibility Segregation).

2.Serverless Design Patterns:

1. Focus on building applications in serverless environments.
2. Example: Function-as-a-Service (FaaS) pattern.

3.Cloud Design Patterns:

1. Patterns tailored for cloud-based applications.
2. Example: Circuit Breaker, Retry Pattern.

Challenges in Adopting Emerging Trends

- **Rapid Technology Changes:**
  - Keeping up with tools and frameworks.
- **Integration with Legacy Systems:**
  - Merging modern practices with existing infrastructure.
- **Skill Gaps:**
  - Training teams to adopt new methodologies and technologies.

Conclusion

- **Key Takeaways:**
  - Emerging trends and design patterns are essential for modern software development.
  - Staying updated helps engineers remain competitive and innovative.
- **Future Outlook:**
  - Continued evolution with advancements in AI, cloud, and distributed systems.

Questions?