CS3219
AY24/25 SEM 1
github/SelwynAng

1 Software Applications, Deployment & Development Processes

1.1 Cloud Computing

- Software infrastructure hosted on an external data center with services delivered over the internet
- Different models
  1. **On-site:** User manages applications, data, runtime, middleware, OS, virtualisation, servers, storage, networking
  2. **IaaS:** User manages applications, data, runtime, middleware, OS | service provider manages the rest
  3. **PaaS:** User manages applications, data | service provider manages the rest
  4. **SasS:** Service provider manages everything
- Cloud native is the software approach of building, deploying and managing modern applications in cloud computing environments

1.2 Deployment (Software Delivery)

- Deployment comprises activities that make the software available for use after development (process between software acquisition and execution)
- **Deployment Issues:** Integration of the internet and related advances (Portability), Large-scale content delivery (Availability, Performance), Heterogeneous platforms (Interoperability), Dependency and change management (Maintainability), Coordination and communication among components (Performance), Security
- **Deployment Mechanisms:**
  1. Bare metal: (+): Complete control, physical isolation | (-): Wasted hardware resources, cost, scalability issues
  2. Virtual machine: (+): Improved resource utilization, flexible, scalable | (-): Vulnerable to side-channel attacks, noisy neighbor problem
  3. Container: (+): Lighter than VM, write once run anywhere, granular control | (-): Not suitable for all apps, not suitable for performance-critical applications
- **Container VS Orchestrator VS Serverless:**
  - Container: Provide the platform for building & distributing services
  - Orchestrator: Separate software that integrate & coordinate many parts, scale up/down deployment, provide fault tolerance, provide communication among containers
  - Serverless: Cloud provider dynamically manages the allocation and provisioning of servers (used for small, stateless, event-driven workloads, e.g., processing an image upload, API endpoints)

1.3 Software Development Process:

- Use waterfall model when requirements are well-understood, fixed, and effort predictable | Use iterative development for fuzzy and evolving requirements
- **CI/CD Pipeline:**
  - Continuous Integration: Development practice that requires developers to integrate code into shared repo several times a day $\rightarrow$ Each check-in is then verified via automated build
  - Continuous Delivery: Ensuring that every good build is potentially ready for production release (Manual deployment to production)
  - Continuous Deployment: Automating release of a good build to production environment (Auto deployment to production)
- **DevOps:** Blends software development & operations staff and tools $\rightarrow$ Reduce time between committing change to system and the change being placed into production while ensuring high quality