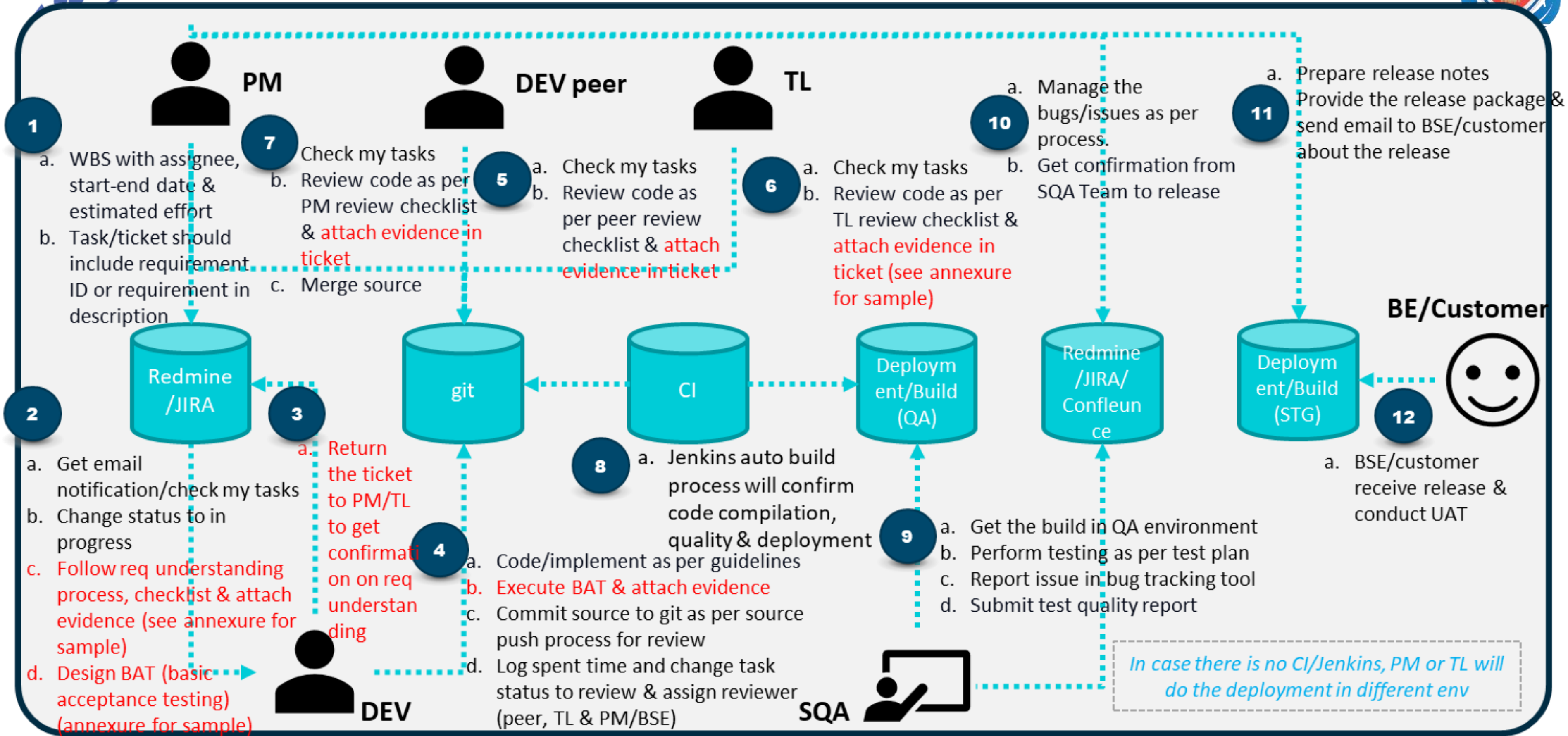
Introduction to Software Development

Process & Technology

Provide diversified training to brush up skills of employees

Typical Development and Release Process



1
a. WBS with assignee, start-end date & estimated effort
b. Task/ticket should include requirement ID or requirement in description

2
a. Get email notification/check my tasks
b. Change status to in progress
c. Follow req understanding process, checklist & attach evidence (see annexure for sample)
d. Design BAT (basic acceptance testing) (annexure for sample)

3
a. Return the ticket to PM/TL to get confirmation on req understanding

4
a. Code/implement as per guidelines
b. Execute BAT & attach evidence
c. Commit source to git as per source push process for review
d. Log spent time and change task status to review & assign reviewer (peer, TL & PM/BSE)

5
a. Check my tasks
b. Review code as per peer review checklist & attach evidence in ticket

6
a. Check my tasks
b. Review code as per TL review checklist & attach evidence in ticket (see annexure for sample)

7
Check my tasks
b. Review code as per PM review checklist & attach evidence in ticket
c. Merge source

8
a. Jenkins auto build process will confirm code compilation, quality & deployment

9
a. Get the build in QA environment
b. Perform testing as per test plan
c. Report issue in bug tracking tool
d. Submit test quality report

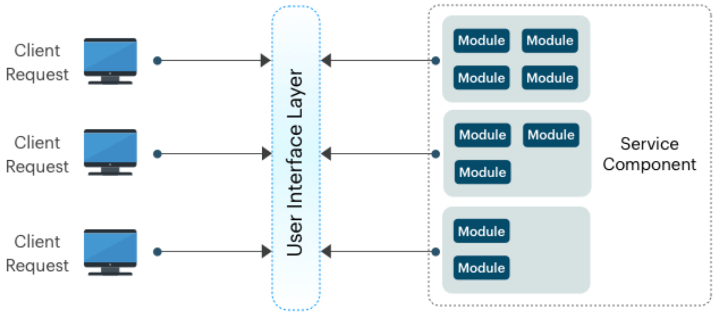
10
a. Manage the bugs/issues as per process.
b. Get confirmation from SQA Team to release

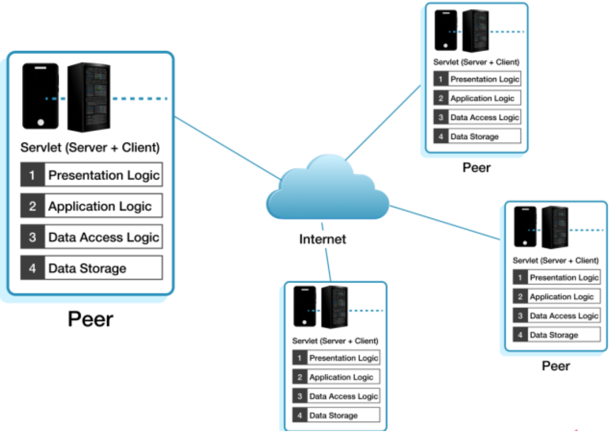
11
a. Prepare release notes
Provide the release package & send email to BSE/customer about the release

12
a. BSE/customer receive release & conduct UAT

In case there is no CI/Jenkins, PM or TL will do the deployment in different env

Software Architectural Patterns





An architectural pattern can be called an outline that allows you to express and define a structural schema for all kinds of software systems.

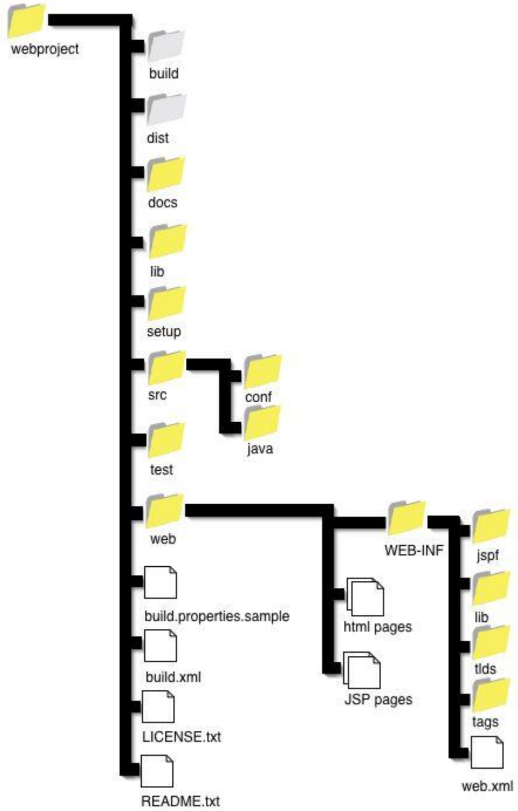
It's a reusable solution that provides a predefined set of subsystems, roles, and responsibilities, including the rules and roadmap for defining relationships among them.

An architectural pattern is a rough image or blueprint of your system, it's not the design pattern.

Famous examples of architectural patterns are **microservices, message bus, service requester/ consumer, MVC pattern, MVVM, microkernel, n-tier, domain-driven design components, and presentation-abstraction-control.**

Web Applications: Recommended Directory Contents

Create Project in any popular IDE, like Eclipse, NetBeans, IntelliJ IDEA, to see the typical Java Web application structure.



Directory Name	Directory Contents
build/	Created by the build tasks and used to hold project-wide build items such as compiled source, assembled modules, or files generated by the Javadoc tool. When deploying a Web module, consider building an unzipped view of the component to enable deployment as a directory structure directly to an application server.
conf/	Configuration or other set-up files, such as Web service configuration files needed by a component or module during the build process. Includes files that are placed in a module's META-INF directory. Also includes configuration files that may require processing prior to being placed in the final module.
dist/	Created by the top-level build dist task, structures under this directory represent the unzipped versions of the binary created by the project.
docs/	Contains all the documentation for a project, including HTML files, installation and setup files, etc.
lib/	Holds specific versions of components of external libraries used by an application. There will be multiple dependent libraries to compile and build the application. Consider including targets that download the correct versions of dependent binaries at build time.
setup/	Contains files that are relevant to the environment for a project or application. This directory may contain database SQL files, ant files containing shared tasks, or any other files that are used to configure a container for a project or application.
web/	Contains the static content of the resulting WAR file.
web/WEB-INF/	Contains the web.xml deployment descriptor and static configuration files, such as faces-config.xml. May also include vendor-specific runtime deployment descriptors, such as sun-web.xml. Generally, this directory contains files that are copied rather than changed during a build. Dynamic configuration files should be placed in the conf/ directory.
test/	The top-level test/ directory contains project-wide tests. Each individual component or module should also include a unit test, which should be placed in the src/test directory for each component or module.

Web & enterprise development stack

Development Scope	Various Web Application accessible through browser ERP application, e-commerce solution, SAP and Salesforce			
Cloud Service	Amazon Cloud Web Services	Google Cloud Web Services	Azure Cloud Web Services	
Languages, IDE, Library	Languages PHP, Java EE, Python, Kotlin	IDE Eclipse, PyCharm, IntelliJ IDEA, Atom, VSCode	Library jQuery, Bootstrap, D3.js, three.js	
Frameworks, Platforms & Architecture	Frameworks Laravel, Spring, Play, Struts, Angular JS, Django, Ruby & Rails, Node.js, ReactJS, Vue.js, MEAN, MERN	Architecture MVC, n-tier, SOA, micro service	Platforms WordPress, Wix, Magento	
Infrastructure	Server Cloud / on premise Web Server (Tomcat, Apache, Nginx etc.)	Database AWS, MSSQL, SQLite, MySQL, PostgreSQL, MongoDB.	Container Docker, Kubernetes	

Smartphone and IoT development stack

Development Scope	Various native Apps for Android and iOS Phone/Tablet IoT based Application, Hybrid App both for Android and iOS Mobile based Gaming application			
Cloud Service	Amazon Cloud Web Services	Google Cloud Web Services	Azure Cloud Web Services	
Languages, IDE, Library	Languages Java, Kotlin, Swift, Objective C, ReactNative, Flutter	IDE Android Studio, Xcode	Library Android SDK, iOS and Mac OS	
Frameworks, Platforms & Architecture	Frameworks Java framework, Android framework, Cocoa Touch, AVFoundation, CloudKit, WebKit, HomeKit, Ionic, CoreData	Architecture MVC, MVVM, Linux Karnel	Platforms Playstore, App Store, Firebase	
Infrastructure	Android, iOS, IoT	Database SQLite, Realm, firebase	Testflight, Google Cloud	

AI Development Stack

Development Scope

- Face / Object recognition
- Chatbot
- Human voice detection
- Language translation
- Text classification

AI Service

- Amazon Cloud AI
- Google Cloud AI
- Microsoft Azure AI
- IBM Watson

Languages, IDE, Library and Visualization

- Languages
Python, C++
- IDE
Pycharm, VS Code, Jupyter
- Library
OpenCV, OpenVino. Auto ML
- Visualization
Seaborn, Jupyter etc.

Platform, Architecture and Algorithms

- Platforms
Tensorflow, Keras, Pytorch, Scikit-Learn
- Architecture
MobileNet, ResNet, U-net etc.
- Algorithms
YOLO, SSD, Mask RCNN etc.

Infrastructure

- Server
Cloud / on premise GPU server
- Data
Video, Audio, SQL, NoSQL
- Learning
Supervised, unsupervised, reinforced

Thank You
For Watching