**ABHISHEK SHARMA**

**R171218005**

**CSE -DEVOPS 2nd YEAR B1**

**Build and Release Management**

**ASSIGNMENT 2**

**Q1. What is the difference between Build Management and Versioning Control?**

|  |  |
| --- | --- |
| **Versioning Control** | **Build Management** |
| A system that records changes to a file or set of files over time so that you can recall specific versions later is called Version control. | Build management is concerned with timing,  content, scheduling, and consistency of the many types of builds possible. |
| Software tools that help a software team manage  changes to source code over time. | Managing the process of converting raw source  code into a distributable package |
| Improves visibility.  Helps teams collaborate around the world. Accelerates product delivery | Reduction of operating cost through energy optimization across sub-systems.  Reduction of operating cost through minimizing maintenance spend.  Reduced service cost through increased staff  productivity. |

**Q2. Write a short note on following**

1. **Smoke Testing: -**

Smoke testing is the initial testing process exercised to check whether the software under test is ready for further testing or not. The term ‘Smoke Testing ‘comes from the hardware testing, in the hardware testing initial pass is done to check if it did not catch the fire or smoke in the initial switch on. These test cases are executed in starting to start actual testing to check critical functionalities of the program is working fine. This set of test cases written such a way that all functionality is verified but not in deep. The Smoke testing is typically executed by testers after every build is received for checking the build is in testable condition. This type of testing is applicable in the Integration Testing, System Testing and Acceptance Testing levels.

Advantages of Smoke testing: -

* It helps to find issues introduced in integration of modules.
* It helps to find issues in the early phase of testing.
* It helps to get confidence to tester that fixes in the previous builds not breaking major features (off course, only features exercised by smoke testing).

1. **Regression Testing: -**

Regression testing is a type of software testing that intends to ensure that changes to the software have not adversely affected it. The probability of any code changes impacting functionalities that are not directly associated with the code is always there and it is essential that regression testing is conducted to make sure that fixing one thing has not broken another thing. During regression testing, new test cases are not created but previously created test cases are re-executed. Regression literally means the act of going back to a previous state, return or reversion.

Types of Regression Tests: -

* **Final Regression Tests: -** A "final regression testing" is performed to validate the build that hasn't changed for a period of time. This build is deployed or shipped to customers.
* **Regression Tests: -** A normal regression testing is performed to verify if the build has NOT broken any other parts of the application by the recent code changes for defect fixing or for enhancement.

1. **Acceptance Testing: -**

Once the System Testing process is completed by the testing team and is signed-off, the entire Product is handed over to the customer/few users of customers/both, to test for its acceptability i.e., Product/application should be flawless in meeting both the critical and major Business requirements. Moreover, end-to-end business flows are verified similar as in real-time scenario. The production-like environment will be the testing environment for Accepting Testing (Usually termed as Staging, Pre-Prod, Fail-Over, UAT environment). Though System testing has been completed successfully, the Acceptance test is demanded by the customer. Tests conducted here are repetitive, as they would have been covered in System testing.

**Q3. Write any Three majorly used open source tools for Build and Release Management. Design a comparison Table between them.**

|  |  |  |
| --- | --- | --- |
| **Ant** | **Maven** | **Gradle** |
| Ant was the first build tool released in 2000 and was easy to learn. | Maven was released in 2004. It has improved few of the problem of ANT. | Gradle was released in 2012. Google adopted Gradle as the default build tool for the  Android OS. |
| Scripts format was XML | Maven continues using XML as the format to write build script, however, structure is  diametrically different. | Gradle does not use XML. Instead, it had its own DSL based on Groovy (one of JVM  languages). |
| **-** | The Most important addition, was the ability to download dependencies over the network (later on adopted by  Ant through Ivy) | **-** |
| **-** | Dependencies management does not handle conflicts well between different versions of the same library. | The Most important addition, was the ability to download dependencies over the network (later on adopted by  Ant through Ivy) |
| XML tends to become  unmanageably big when used with all but very small projects | Customization of targets (goals) is hard | Gradle build scripts tend to be  much shorter and clearer than those written for Ant or Maven. |

Prepared by: - Dr. Hitesh Kumar Sharma Page 3