Assignment 1: Configuration Management

Aaron Kephart

Successful software engineering requires some level of governance and principles while progressing through design and short or long-term maintenance phases. Configuration management satisfies this requirement and is defined as “establishing processes for ensuring the consistency and performance of products, analyzing physical and functional requirements, and monitoring the design and operation of the product through its life cycle” (Project Schedule Configuration Management Policies, 2018). The main goal of configuration management is to eradicate negative impacts specifically when software is being designed or changes are requested. A lack of configuration management can cause significant company setbacks or application downtime.

Configuration component activities are a part of the configuration management process. Knowing software systems will at some point require updates and there will potentially be change requests or updates made in the future, configuration component activities must be followed. These activities include scope, which “establishes boundaries and guidelines applicable to the configuration change;” references, which list “publications, specifications, and requirements;” configuration item, which “identifies the component or components being changed within the system;” baseline, which is “a benchmark established for future reference enabling the comparison of deltas between new and old hardware and software configurations;” and finally, verification, which “determines the accuracy and completeness of the configuration data while exercising zero error tolerance to protect the integrity of the CM database” (What Is Configuration Management? - Definition, Process & Tools). The components of configuration management are well-defined and when put into practice, fulfill the configuration management process.

Because of the sheer importance of prioritizing configuration management, a lack thereof may result in setbacks. For example, after extensive development of a software solution, it is ready to be implemented but suddenly experiences an unexpected critical error due solely to the application being deployed on unforeseen outdated or misunderstood hardware; this impacts performance significantly (What Is Configuration Management? - Definition, Process & Tools). Also, it may be deduced that the development team and most importantly the customer are suffering through this process as well because the application itself is not working while the development team hastily tries to figure out the issue; application downtime is apparent and the reputation of the development team becomes diminished.

Secondly, "system compliance with established specifications and guidelines" resonates with the above example (What Is Configuration Management? - Definition, Process & Tools). If configuration management is lacking, a possible setback would be that the system does not comply with specifications or guidelines. This paves the way for catastrophe because without guidelines or principles, we can assume each developer would follow their own rendition of what is correct and it may not be in the best interest of the application or organization.

Thirdly, application development or changes become considerably more error-prone when a lack of configuration management exists (What Is Configuration Management? - Definition, Process & Tools). When configuration management is followed, a baseline is set and so are scope, references, configuration items, and verification. Particularly, when a hardware or software deployment or update is scheduled, an entire process initiates where changes are monitored, controlled, and documented upon; a project baseline is referenced if there is any issue with this deployment, and prior documentation can be accessed to revert or fix any broken components (What Is Configuration Management? - Definition, Process & Tools). If a solid plan is in place with guidelines, issues that arise can be more easily addressed and have less of an impact.

Fourth, enhanced online availability, and availability in general may be impacted when a lack of configuration management exists (What Is Configuration Management? - Definition, Process & Tools). Downtime is a major event where an application is no longer operational, and is assumed to be a fear of any developer or organization. The ideology of proper configuration management planning resonates with application uptime. Thinking of any commonly used application, system reliability is vital to efficiency and reputation. A solid configuration management plan lowers risk of downtime.

Finally, system performance suffers when a lack of configuration management exists (Project Schedule Configuration Management Policies, 2018). As previously stated, in a configuration management plan, a baseline is set. This baseline can be utilized to measure performance and reliability of a new version of the software compared to the baseline (Project Schedule Configuration Management Policies, 2018). System performance directly impacts usability and reliability of a system. If a system is down, it cannot be utilized and this has wide-ranging business impacts, including financial, efficiency, and reputation.

In summary, configuration management is vital to software engineering. It offers considerable benefit by structuring software version upgrades and change requests. A lack of configuration management results in many avoidable setbacks that have wide-ranging impacts. When weighing the benefits and consequences, implementing a configuration management process has significant benefits in software engineering and should be applied and practiced.

References

Project Schedule Configuration Management Policies. (2018, February 13). Retrieved from https://study.com/academy/lesson/project-schedule-configuration-management-policies.html.

What Is Configuration Management? - Definition, Process & Tools. (2017, June 22). Retrieved from https://study.com/academy/lesson/what-is-configuration-management-definition-process-tools.html.