Gerência de Configuração de Software

Jônatas Gabriel da Silva Santos

Definição – O que é GCS?

“A gerência de configuração de software (GCS) é uma atividade do tipo "guarda-chuva", aplicada através de toda a gestão de qualidade. Como as mudanças podem ocorrer em qualquer instante, as atividades de CGS são desenvolvidas para (1) identificar a alteração, (2) controlar a alteração, (3) assegurar que a alteração esteja sendo implementada corretamente e (4) relatar as alterações a outros interessados”.

PRESSMAN (2011)

Definição – O que é GCS?

"A GCS é direcionada para identificar e documentar as características funcionais e físicas de um item de configuração, controle de mudanças para essas características, registro e relatório de alteração, assim como o estado de implementação e verificar a conformidade com os requisitos especificados".

"A GCS é um processo do ciclo de vida de software que apoia e beneficia as atividades de gerenciamento de projetos, desenvolvimento e manutenção, atividades de garantia de qualidade, bem como os clientes e usuários do produto final".

SWEBOK (2014)

Definição – O que é GCS?

"A GCS é fundamental para prover controle sobre os produtos de trabalho produzidos e modificados por diferentes engenheiros de software. Além disso, possibilita um acompanhamento minucioso do andamento das tarefas de desenvolvimento".

SOFTEX (2016)

Definição – O que é GCS?

Na Engenharia de Software, as principais atribuições da Gerência de Configuração de Software são:

Definição – O que é GCS?

A GCS permite identificar as unidades que compõe o sistema de acordo com as funcionalidades que deverão desempenhar, e as interfaces entre essas unidades, documentando a interação entre elas. O controle contínuo da evolução das funcionalidades e interfaces permite que a integração entre as unidades tenha sucesso continuado, tendo assim as mudanças gerenciadas e documentadas. A auditoria das funcionalidades identificadas, documentadas e controladas, garante a confiabilidade do sistema.

Definição – O que é GCS?

Mudanças aparecem durante todo o desenvolvimento e devem ser registradas, avaliadas e agrupadas de acordo com sua prioridade. Com isso é possível planejar melhor o escopo, prazo e o custo de cada iteração. Em seguida, à medida que o desenvolvimento acontece, pode-se acompanhar o estado da solicitação da mudança até sua implementação e até o lançamento de uma versão em produção.

Controle de Mudanças

- Mudanças aparecem durante todo o desenvolvimento e devem ser registradas, avaliadas e agrupadas de acordo com sua prioridade. Com base nessas informações, é possível planejar melhor o escopo, prazo e o custo de cada iteração. Em seguida, à medida que o desenvolvimento acontece, pode-se acompanhar o estado da solicitação da mudança até sua implementação e até o lançamento de uma versão em produção.

Controle de Mudanças

Existem várias ferramentas para suportar o controle de mudanças na GCS. Algumas delas são:

- BitBucket
- Bugzilla
- GitHub
- Jira
- Phabricator
- Redmine
- Trac

Controle de Versão

- Cada vez que uma solicitação de mudança é implementada, acontece um incremento na evolução do projeto que deve ser registrado no histórico. Este incremento na evolução corresponde a uma configuração*.

  - ***Configuração** é o estado do conjunto de itens que formam o sistema em um determinado momento.

Controle de Versão

As funcionalidades oferecidas pelo controle de versão vão além do simples registro do histórico das configurações. Além do registro das configurações, o controle de versão tem outras responsabilidades importantes: possibilitar a edição concorrente sobre os arquivos e a criação de variações no projeto.

O controle de versão é a parte principal da GCS. É o elo comum entre o controle de mudança e a integração do projeto.

Controle de Versão

As principais ferramentas que suportam controle de versão são o Git e o Subversion.

A seguir, algumas diferenças sobre o Git e o Subversion.

Git X Subversion

Git

- Distribuido
- Servidor e repositório centralizados
- Conteúdo armazenado como metadata
- Não tem número de revisão global
- Desenvolvido para o Kernel do Linus por Linus Torvalds
- Licença GNU
- Melhor proteção de conteúdo
- Branchs mais fáceis de se trabalhar

Subversion

- Não distribuido
- Servidor e repositórios não centralizados
- Conteúdo armazenado em arquivos de conteúdo
- Possui número de revisão global
- Desenvolvido pela CollabNet, Inc.
- Licença de Código Aberto

Integração Contínua

O objetivo da integração é verificar se a construção do sistema a partir dos itens registrados em uma configuração é bem sucedida. Integrar o sistema consiste em construir o sistema a partir dos itens registrados em uma configuração.

Em termos práticos, a integração é feita através de scripts que automatizam a construção, testes e também a coleta de métricas de qualidade. As ferramentas de integração contínua acompanham o controle de versão e disparam os scripts cada vez que uma nova configuração é registrada.

Integração Contínua

E eis algumas ferramentas de suporte à Integração Contínua:

- BuildBot
- CircleCI
- CodeClimate
- CodeShip
- Concourse
- Drone.io
- Jenkins
- Travis CI

G. C. S.

A Gerência de Configuração de Software é a base para as demais áreas de Engenharia de Software: Por isso, aparece como requisito de implementação já no nível inicial de diversos modelos de maturidade de processo de desenvolvimento tais como o CMMI-DEV, SPICE e o MPS-Br.

Maven

Maven é uma ferramenta de automação de compilação utilizada primariamente em projetos Java. É utilizada para construir e gerenciar projetos escritos em C#, Ruby, Scala e outras linguagens. O projeto Maven é hospedado pela Apache Software Foundation, que fazia parte do antigo Projeto Jakarta.

Maven – O arquivo pom.xml

O Maven utiliza um arquivo XML (POM) para descrever o projeto de software sendo construído, suas dependências sobre módulos e componentes externos, a ordem de compilação, diretórios e plug-ins necessários.

Ele vem com objetivos pré-definidos para realizar certas tarefas bem definidas como compilação de código e seu empacotamento.

Maven – O arquivo pom.xml

```
        </plugins>
        <finalName>MavenEnterpriseApp-ear</finalName>
    </build>
        <dependencies>
            <dependency>
                <groupId>com.mycompany</groupId>
                <artifactId>MavenEnterpriseApp-ejb</artifactId>
                <version>1.0-SNAPSHOT</version>
                <type>ejb</type>
            </dependency>
            <dependency>
                <groupId>com.mycompany</groupId>
                <artifactId>MavenEnterpriseApp-web</artifactId>
                <version>1.0-SNAPSHOT</version>
                <type>war</type>
            </dependency>
        </dependencies>
</project>
```

Exemplo de pom.xml com declaração de dependências.

Maven – O arquivo pom.xml

O Maven automaticamente baixa as dependências declaradas e as dependências das dependências (chamadas dependências transitivas) e as armazena em um repositório local do usuário.

O Repositório Central do Maven 2 é usado por padrão para procurar bibliotecas, mas pode-se configurar os repositórios usados (e.g., repositórios privados de uma empresa) no POM.

Maven - Benefícios

Dentre os principais benefícios do uso do Maven para construção e integração de projetos, pode-se citar:

- Existência de plug-ins para as principais IDEs: Eclipse e NetBeans;
- Configuração inicial do projeto com o uso de boas práticas e sem a necessidade de codificação;
- Ao criar um projeto Maven utilizando archetypes (templates), automaticamente são criadas pastas com arquivos, classes com exemplos de código e o pom.xml já previamente configurado;
- Uso do repositório Maven Central para gerenciamento das bibliotecas open source;

Maven - Benefícios

Dentre os principais benefícios do uso do Maven para construção e integração de projetos, pode-se citar:

- Uso do repositório local, permitindo que a mesma biblioteca seja utilizada por qualquer projeto configurado na máquina;
- Possibilidade de criação de repositórios próprios para o gerenciamento de bibliotecas internas, facilitando a reutilização destas nos projetos;
- Possibilidade de usar tarefas Ant para o deployment, facilitando a reutilização do legado;
- Gerenciamento de bibliotecas e versionamento com pouca ou nenhuma configuração adicional.

Maven – Ciclo de Vida

Um projeto Maven pode é dividido nas seguintes fazes:

- VALIDAR - Validar se o projeto está correto e todas as informações necessárias estão disponíveis.
- COMPILAR - Compilar o código fonte do projeto.
- VALIDAR - Validar se o projeto está correto e todas as informações necessárias estão disponíveis.
- COMPILAR - Compilar o código fonte do projeto.
- PACOTE - Pegar o código compilado e empacotá-lo em seu formato de distribuição, como um JAR.

Maven – Ciclo de Vida

Um projeto Maven pode é dividido nas seguintes fazes:

- VERIFICAR - Executar todas as verificações sobre os resultados de testes de integração para garantir se critérios de qualidade são cumpridos.
- INSTALAR - Instalar o pacote no repositório local, para uso como uma dependência em outros projetos localmente.
- IMPLANTAR - Feito no ambiente de compilação, copia-se o pacote final para o repositório remoto para compartilhar com outros desenvolvedores e projetos.

Auditoria de Configuração

Auditoria de Configuração Física

Uma auditoria de configuração física (PCA) identifica os componentes de um produto que serão implantados do Repositório do Projeto. Os passos são:

- Identificar a baseline a ser implantada (geralmente é apenas um nome e/ou número, mas também pode ser uma lista completa de todos os componentes e suas respectivas versões).
- Confirmar que todos os artefatos necessários, conforme especificado pelo Caso de Desenvolvimento, estão presentes na baseline. Liste os artefatos ausentes em Descobertas da Auditoria de Configuração.

Auditoria de Configuração

Outros níveis da auditoria de configuração física

Algumas organizações usam uma Auditoria de Configuração Física para confirmar a consistência do design e/ou documentação do usuário com o código. O Rational Unified Process recomenda que a verificação dessa consistência seja executada como parte da atividade de revisão no decorrer do processo de desenvolvimento. No último estágio, as auditorias devem se limitar a verificar os produtos liberados necessários que estão presentes, sem se preocupar em revisar o conteúdo.

Referência

https://blog.pronus.io/posts/o-que-eh-gerencia-de-configuracao-de-software/

https://pt.wikipedia.org/wiki/Ger%C3%AAncia_de_configura%C3%A7%C3%A3o_de_software

https://www.devmedia.com.br/as-diferencas-entre-git-e-svn-revista-java-magazine-116/28079

http://www.differencebetween.net/technology/software-technology/difference-between-git-and-svn/#ixzz51AZExabt

https://netbeans.org/kb/docs/javaee/maven-entapp_pt_BR.html

https://pt.wikipedia.org/wiki/Apache_Maven

https://www.devmedia.com.br/fundamentos-do-maven/29315

http://web.unipar.br/~seinpar/2016/publicacao/Julio_Fernandes_Rocha.pdf