UNIPÊ
Pós-graduação

Especialização em Desenvolvimento Web

Gerência de Configuração de Software
Conceitos

André Figueiredo

andre.unipe@gmail.com

- What about me!
- What about you!

Agenda

- Gerência de Configuração (14/04)
  - Conceitos, termos, etc.
- Gerenciamento de Builds (14/04 e 15/04)
  - Maven
  - Gradle
- Sistemas de Controle de Versão (16/04)
  - Git
- Ferramentas para Gestão de Mudanças (16/04)
  - Redmine
- Gestão de Build e Deploy (16/04)
  - Jenkins

Dinâmica

- 3 voluntários com conhecimento mínimo de HTML;
- Alterar “As 5 Melhores Praias da Paraíba.html” com o que o “cliente” pede;
- Regras:
  - Os participantes podem conversar sobre organização, mas, não podem se falar sobre o que deve ser feito;
  - Cada um tem que fazer a sua parte;
  - Todos devem trabalhar em paralelo. Um não pode esperar pelo outro.

UNIPÊ
Pós-graduação

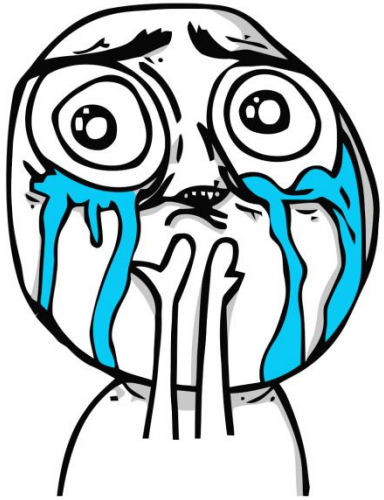
Ferramentas

git

CVS

SVN

SUBVERSION



Jenkins

TC

Buildbot

Travis CI

APACHE ANT

ivy

REDMINE

Gradle

Bugzilla

JIRA

Maven

Bower

GRUNT

npm

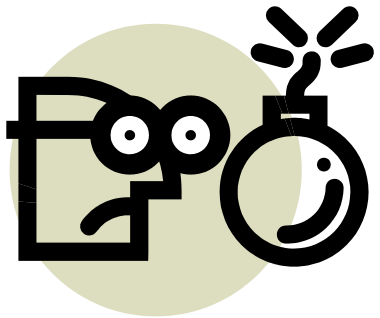
Trello

Motivações para GCS

- Desenvolvimento iterativo e incremental impõe a necessidade do controle de versões de artefatos do software em desenvolvimento e versões em produção.
- Desenvolvimento de software concorrente, em equipe.
- Desenvolvimento distribuído, com equipes geograficamente dispersas.
- Controle sobre diferentes distribuições do software (i.e. *standard, professional, enterprise* etc.).
- Melhor controle e acompanhamento das manutenções.

Problemas pela Falta de GCS

- Perda de código fonte ou de outros artefatos do desenvolvimento.
- Impossibilidade de rastrear as evoluções, correções e adaptações no software.
- Impossibilidade de se controlar Quem, Por que e Quando uma modificação foi efetuada.
- Programa em execução e código fonte em diferentes versões.
- Dificuldade de sincronizar o trabalho em equipe.
- Dificuldade em se retornar o produto a um estado anterior.



etc....

Conceitos

Definição

Gerência de Configuração: *"Disciplina aplicando direcionamento técnico e administrativo para identificar e documentar as características físicas e funcionais de um item de configuração, controlar as modificações sobre estas características, gravar e relatar o processamento da mudança e o status da sua implementação, e verificar a sua adequação aos requisitos especificados." [IEEE]*

Conceitos

- **Plano de gerência de configuração:** documento que descreve como será aplicada a GC em um projeto;
- **Item de configuração:** agregação de hardware e/ou software que será passível de GC e tratado como um elemento único (e.g.: plano de GC, requisitos, modelos, código-fonte, casos de testes, etc);
- **Versões:** instâncias de um mesmo item de configuração que diferem entre si em algo (sinônimo: revisões);

Plano de Gerência de Configuração

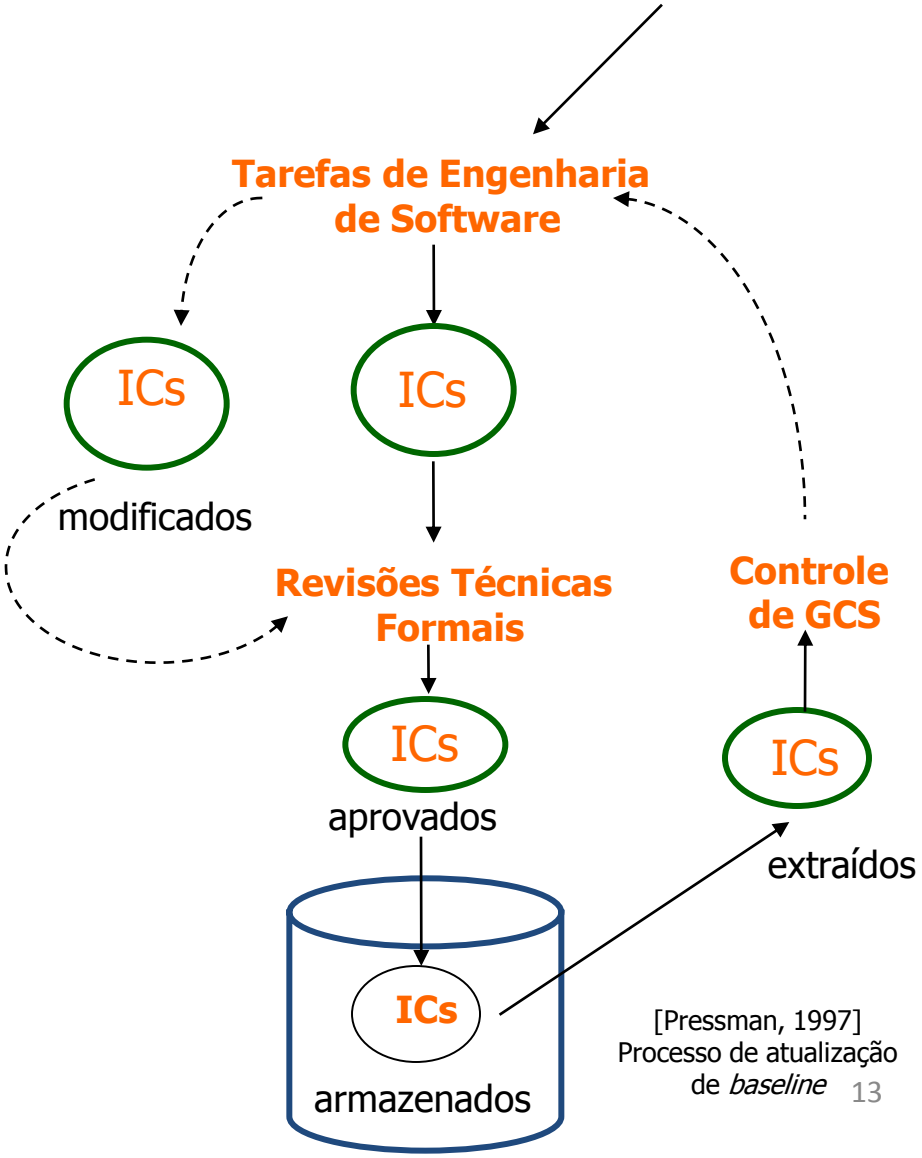
- A gerência de configuração pode utilizar ferramentas, convenções de nomes, políticas e procedimentos.
- O grau de utilização desses elementos deve variar em função das características do projeto (e.g.: tamanho, volatilidade, processo adotado).
- O **plano de gerência de configuração de software** é o documento utilizado para descrever como será utilizada a disciplina de gerencia de configuração no projeto em questão.

Plano de Gerência de Configuração

- Pode ser definido um plano padrão para a empresa como um todo ou para determinados departamentos.
- Esse plano padrão deverá ser adaptado para cada novo projeto, levando em consideração as suas peculiaridades.
- Apesar de existirem várias normas para GC (e.g.: FAA, NASA, DoD, MIL, IEEE, ISSO, etc.), escolha a mais adequada!
- Coloque aquilo que é importante!

Repositórios e *Baselines*

- **Repositórios**: local de acesso controlado, onde as versões dos itens de configuração são armazenadas.
- ***Baselines***:em momentos específicos no desenvolvimento de software (denominados *milestones*), o conjunto de Itens de Configuração resultante é revisado, aprovado e formalmente designado como uma *Baseline.*



[Pressman, 1997]
Processo de atualização de *baseline*

Baseline (Configuração de Referência)

- Baseline é um conceito de GCS que ajuda a controlar as mudanças, pois os Itens de Configuração em uma baseline somente podem ser modificados através de procedimentos formais de controle de mudança.
- Modificar um Item de Configuração em uma Baseline requer todo um controle sobre a sua saída do Repositório (Checkout) e atualização no repositório (Checkin).
- Como as modificações são controladas sobre os ICs em uma Baseline, a decisão sobre quando ICs devem compor Baselines deve ser ponderada.

Quando estabelecer Baselines?

- Normalmente, baselines são estabelecidas após o término de cada fase do Ciclo de Vida do software: Requisitos, Projeto, Codificação + Testes, Release.
- Uma baseline representa a configuração do software, através das configurações dos seus ICs, em determinados momentos do tempo.
- Cada baseline serve como um ponto de partida para a próxima fase do ciclo de vida.

Quando estabelecer Baselines?

- Baselines devem ser estabelecidas de maneira ponderada, levando em conta uma análise de custo-benefício ("trade-offs"):
  - Baselines determinam marcos no ciclo de vida.
  - ICs em uma baseline têm a sua modificação controlada.
  - Controle de mudança apóia o trabalho em equipe, o trabalho gerencial etc.
  - Mudanças sobre os itens em uma baseline seguem um procedimento "burocrático".

Check-in e Checkout

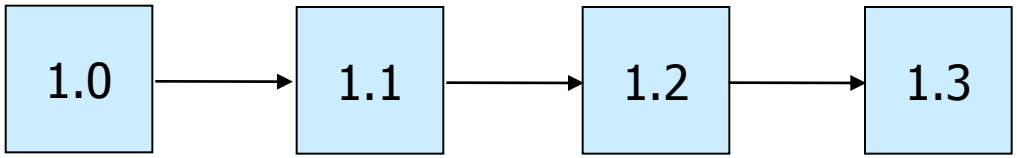
- Check-in: formalmente, representa o processo de revisão, aprovação e inclusão/atualização de um item em um repositório controlado. O processo de mudança gera uma nova versão ou revisão do IC.
- Checkout: uma vez que o pedido de mudança seja aprovado, o item pode ser copiado do repositório para uma base local para que as modificações possam ser realizadas.

Versões, Variantes e *Releases*

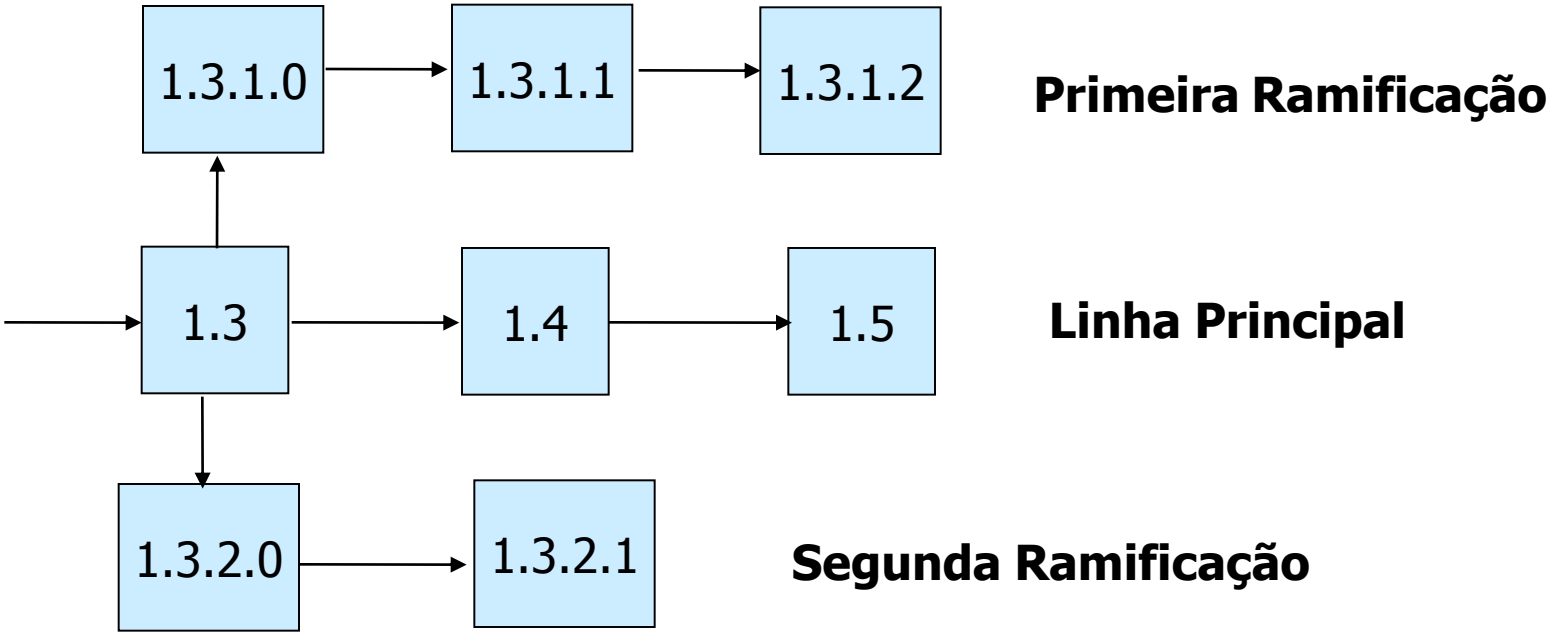
- Versões (ou Revisões): instância controlada de um item de configuração, criada em determinado momento no tempo, que difere de outras instâncias através de alguma característica.
- Variante: versões funcionalmente equivalentes, mas projetadas para ambientes de hardware ou software distintos.
- Release (ou Liberação): conjunto de produtos que deve ser entregue ao cliente. Liberações diferem de versões pois versões podem ser somente para uso interno ao projeto.

Desenvolvimento Paralelo e Ramificação (Branching)

Versões Lineares de um Item de Configuração



Versões ramificadas de um Item de Configuração (Branches)



Desenvolvimento Paralelo e Ramificação (Branching)

- **Ramos (*Branches*)**: versões que não seguem a linha principal de desenvolvimento. Os ramos são **temporários** e terão que sofrer junção (*merge*) com a linha principal de desenvolvimento.
- *Branches* são adequados para permitir que 2 ou mais desenvolvedores trabalhem sobre um mesmo item ao mesmo tempo (desenvolvimento paralelo, concorrente).

Desenvolvimento Paralelo e Ramificação (Branching)

- **Números de Versões em *Branches*:**
  - Números de Versões da Linha de desenvolvimento principal contêm 2 partes (*major* e *minor*).
  - Números de versões em *Branches* contêm 4 partes: os dois primeiros números correspondem ao ponto em que o *branch* (ramo) se desvia da linha principal; o terceiro número corresponde ao número do *branch;* o quarto número indica a versão do artefato naquele *branch* específico.

Outros Conceitos Importantes

- **Item Derivado**: item de configuração que pode ser obtido a partir de outro item de configuração. Ex: arquivos binários, executáveis etc.
- **Item Fonte**: item de configuração que serve como origem para um item derivado.
- Exemplo: os itens de configuração que compõem o código-fonte são itens fonte para o programa executável, que é item derivado.
- É necessário documentar as ferramentas e o ambiente utilizado, para que a geração possa ser repetível. Por exemplo, para programas em Java, deve ser registrada a versão do compilador utilizada na geração da release, parâmetros de compilação, bibliotecas utilizadas, recursos, classpath, buildfile etc.

Outros Conceitos Importantes

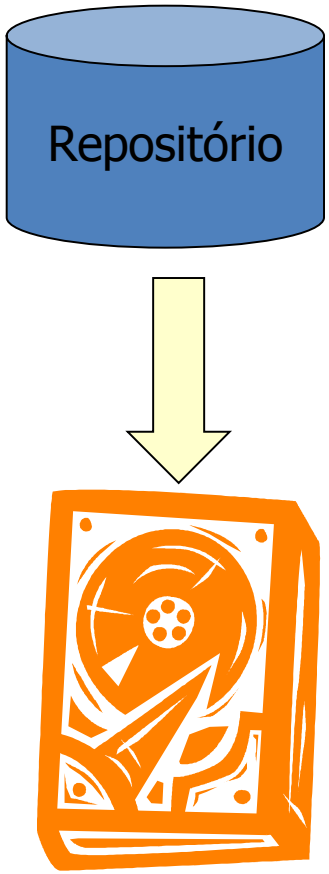
- Building (construção): processo de compilação do sistema a partir dos itens fonte para uma configuração alvo.
- Utiliza arquivos de comando que descreve como deve ocorrer a construção.
- Exemplo: *makefile, pom.xml e build.gradle*.
- Os arquivos de comando também são considerados itens de configuração.

Outros Conceitos Importantes: *Deltas*

- É importante que sistemas de versionamento mantenham registro da última versão e das versões anteriores de um item de configuração.
- Usuários (releases) podem estar usando versões mais antigas dos itens.
- Versões mais antigas nos ajudam a detectar problemas em versões mais recentes.

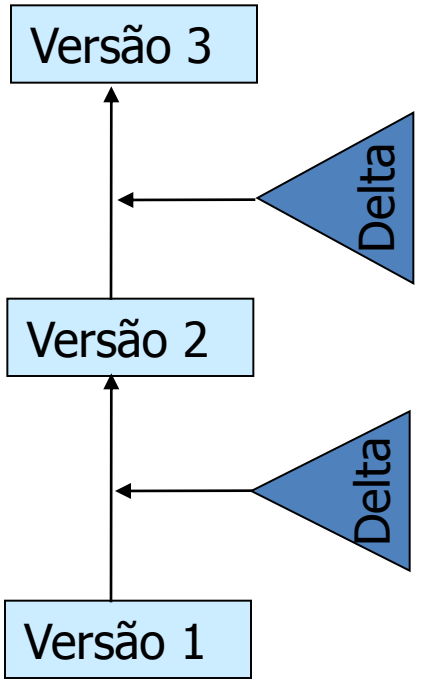
Outros Conceitos Importantes: *Deltas*

- Idealmente, cópias de todas as versões dos itens poderiam ser mantidas no repositório. Mas, isso requer considerável espaço de armazenamento em disco...
- **Deltas** visam poupar este espaço de armazenamento! *Delta* representa a diferença entre a versão atual e a última versão de um item de configuração.

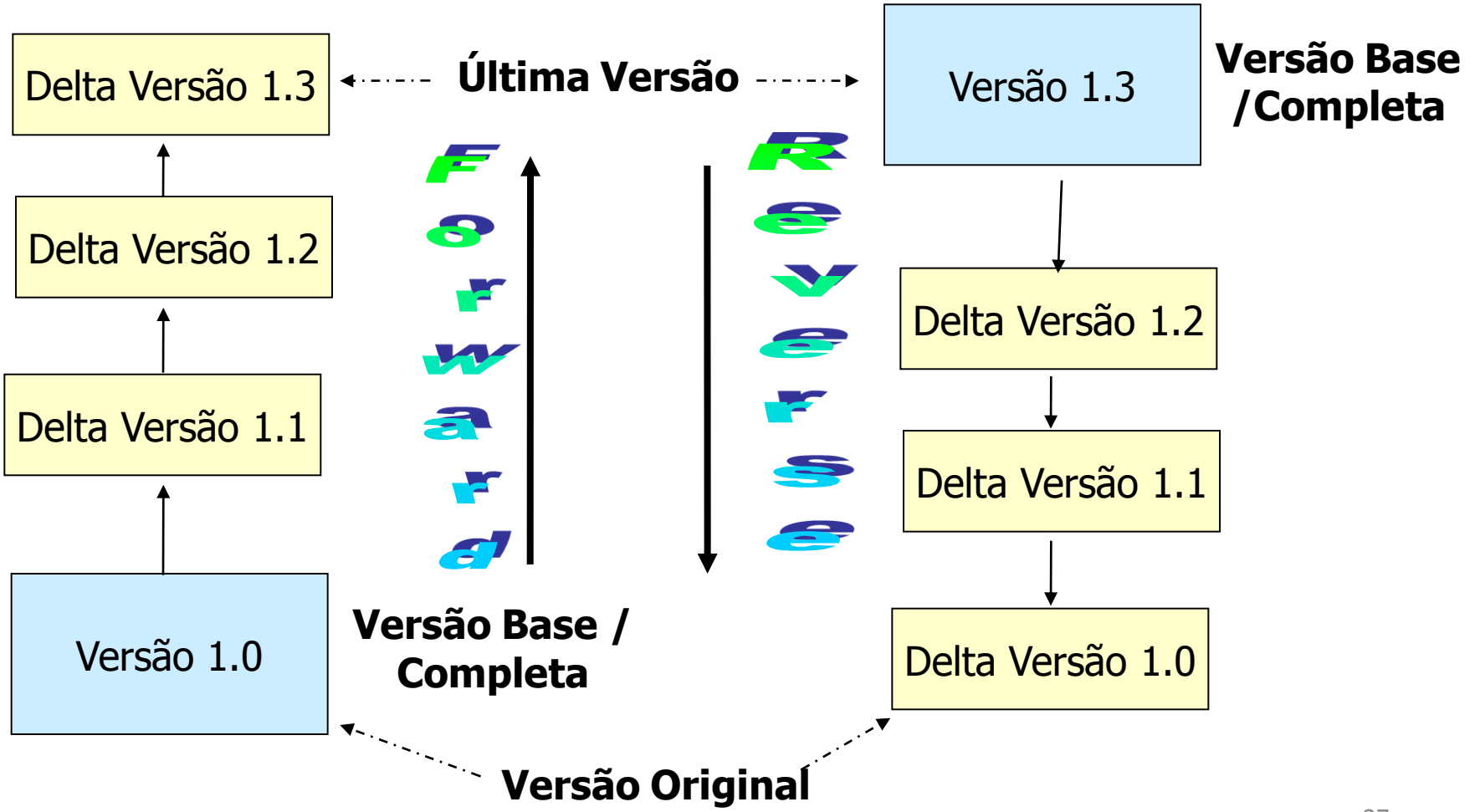


Deltas

- Ao invés de armazenar cópias completas de todas as versões de um item, uma única versão e seus *Deltas* são armazenados.
- Assim, a qualquer momento, versões podem ser derivadas a partir de uma versão base.
- *Deltas* podem ser: *forward* e *reverse*.



Deltas: Forward e Reverse



Delta: Considerações

- A decisão pela adoção do Delta em ferramentas de Controle de Versões envolve uma análise de custo-benefício de espaço de armazenamento x tempo de recuperação.
- Algumas ferramentas oferecem a facilidade de se optar pelo uso de *Delta* ou armazenamento das cópias completas das versões dos itens.

Atividades

- IEEE divide as funções da SCM nas quatro atividades seguintes:
  - **1. Identificação da Configuração**
  - 2. Controle da Configuração
  - 3. Relato de Status
  - 4. Auditorias e Revisões

Identificação da Configuração

- **Atividades**: (1) estabelece os itens a serem controlados, (2) estabelece esquemas de identificação para os itens e suas versões, e (3) estabelece as ferramentas e técnicas a serem utilizadas na aquisição e gerência dos itens controlados.
- **Atividades 1 e 2** envolvem:
  - **Seleção de itens de configuração**: agrupamento dos artefatos de software em itens de configuração, que serão sujeitos ao controle da SCM.
  - **Designação**: desenvolvimento de um esquema de numeração ou nomenclatura para relacionar os artefatos de software e a sua documentação associada.
  - **Descrição**: documentação das características funcionais e físicas para cada um dos itens de configuração. Determinar que características devem ser capturadas, de forma que as propriedades e os requisitos do produto sejam refletidos corretamente, é uma decisão importante.

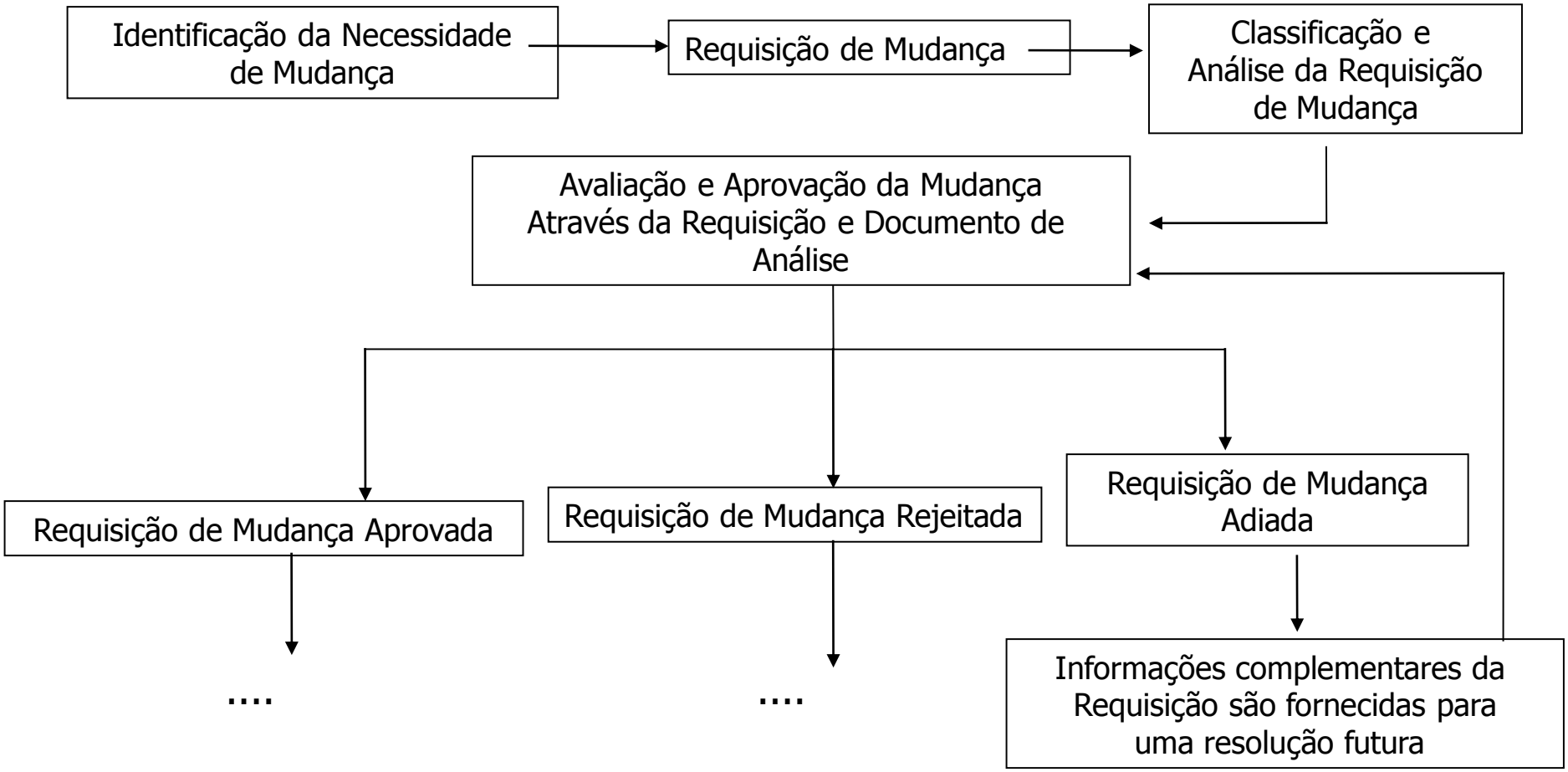
Item de Configuração

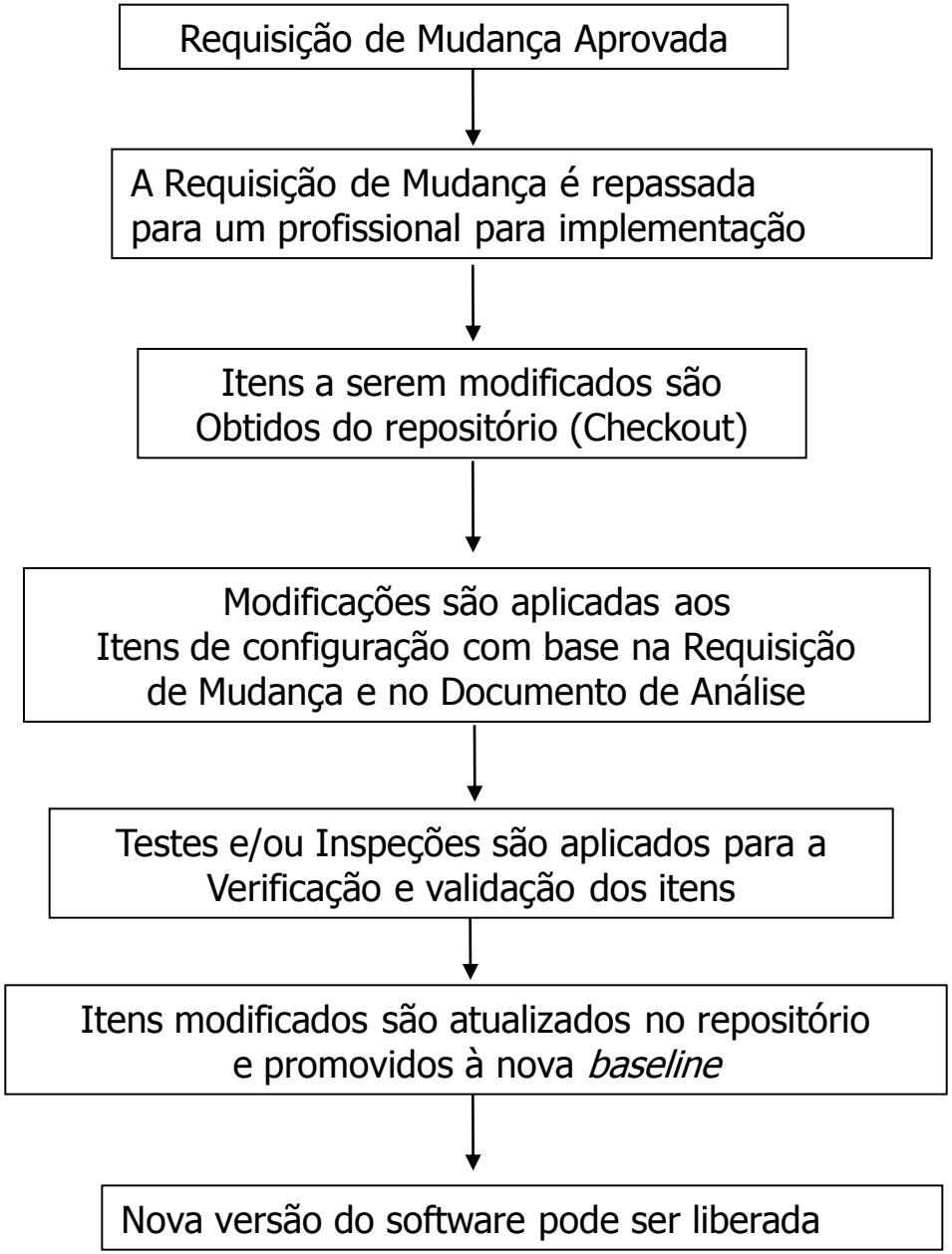
- **Item de Configuração (IC)**: agregação de hardware e/ou software que será passível de gerência de configuração e tratado como um elemento único.
- **Exemplos**: planos, especificação de análise e projeto, modelos, material de teste, código fonte, código executável, bibliotecas, documentação de instalação, documentação de manutenção, manual de usuário, dentre outros.
- **Características de um Item de Configuração**: é versionado, tem a sua evolução/modificação controlada (quem, por que, quando), fonte de informação sobre o software, pode ser composto.

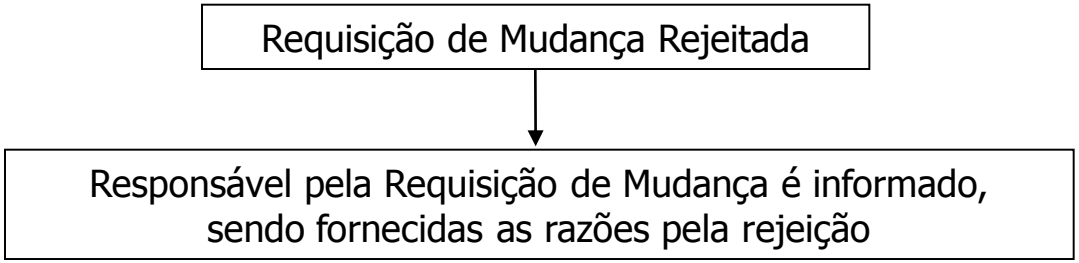
Atividades

- IEEE divide as funções da SCM nas quatro atividades seguintes:
  - 1. Identificação da Configuração
  - **2. Controle da Configuração**
  - 3. Relato de Status
  - 4. Auditorias e Revisões

Processo de Controle da Configuração




Requisição de Mudança Aprovada
A Requisição de Mudança é repassada para um profissional para implementação
Itens a serem modificados são Obtidos do repositório (Checkout)
Modificações são aplicadas aos Itens de configuração com base na Requisição de Mudança e no Documento de Análise
Testes e/ou Inspeções são aplicados para a Verificação e validação dos itens
Itens modificados são atualizados no repositório e promovidos à nova *baseline*
Nova versão do software pode ser liberada


Requisição de Mudança Rejeitada
Responsável pela Requisição de Mudança é informado,
sendo fornecidas as razões pela rejeição

Formulário de Requisição de Mudança

- Informações: Nome do Projeto/Sistema; Componente/Item a ser Modificado; Classificação da Mudança; Prioridade; Descrição; Status; Observações; Número de Requisição da Mudança (Modificação).
- Status: Iniciada (submetida); Recebida; Classificada: Aprovada/Rejeitada/Adiada; Analisada; Atribuída a um Desenvolvedor; Checked-out; Modificações realizadas e testadas; Revisada; Aprovada; Checked-in; Incluída em uma Nova Baseline.
- Requisições de Mudança devem ter um identificador único para fins de rastreamento.

Classificação e Análise da Mudança

- Os critérios de classificação das mudanças devem estar detalhados no plano de gerência de configuração.
- A **classificação** visa priorizar modificações mais importantes (críticas, fatais, não fatais, cosméticas).
- A **análise** visa relatar os impactos em custo, cronograma, funcionalidades, etc. da implementação da modificação.

Informações do Relatório de Análise da Mudança

- Número da Requisição de Modificação;
- Sistema/Projeto;
- Item a ser analisado;
- Analisada por;
- Itens afetados;
- Alternativas de implementação;
- Esforço estimado;
- Impacto no cronograma e custo do projeto.

Análise de Impacto de Modificação em Software

- A análise de impacto pode ser definida como o processo de identificação de possíveis conseqüências de uma modificação no software (BOHNER, 2002).
- A partir da análise do impacto da modificação no software, estima-se o esforço e o custo da implementação.
- Técnicas para apoiar a Análise de Impacto:
  - Análise de documentos de análise e projeto;
  - Engenharia reversa;
  - Busca de referências a variáveis e funções (ex: Eclipse);
  - Análise de artefatos modificados em conjunto em repositórios de gerência de configuração (ex:Odyssey-WI).
  - Etc.

Classificação e Análise da Mudança

- Caso a **análise** conclua que não existe chance de aprovar a modificação (casos extremos), pode ocorrer rejeição antes da avaliação para poupar custos no processo.
- A **avaliação** utilizará a requisição de modificação e o laudo da análise para tomar a decisão.
    - A requisição de modificação pode ser aceita, rejeitada ou adiada.
    - No caso de rejeição, deve ser informada a razão, e no caso de adiamento deve ser informado o novo prazo, a razão e as novas informações desejadas (caso se aplique).

Implementação da Modificação

- Checkout dos itens de configuração do repositório;
- Implementação das mudanças com base na requisição e documento de análise da modificação:
  - Para manutenção em código, Testes de Unidade devem ser realizados.
  - Durante a verificação, devem ser aplicados Testes de Sistema também. Isso pode requerer a re-execução de testes especificados anteriormente (testes de regressão).
  - A documentação de projeto deve ser revisada para refletir a mudança.

Implementação da Modificação

- Itens modificados são atualizados no repositório.
- Uma nova configuração de referência ou *baseline* é gerada.
- Uma nova *release* pode ser gerada. Mas, geralmente mudanças são agrupadas para a geração de uma nova *release*.
- No caso de correções emergenciais, podem ser criados ramos sem a necessidade do processo formal.

Atividades

- IEEE divide as funções da SCM nas quatro atividades seguintes:
  - 1. Identificação da Configuração
  - 2. Controle da Configuração
  - **3. Relato de Status**
  - 4. Auditorias e Revisões

Relato de *Status* (*Status Accouting*)

- Representa o acompanhamento da configuração.
- Envolve o registro e relato de informações necessárias para efetivamente se gerenciar um software e os seus itens de configuração.
- Inclui, entre outros itens de informação:
  - Registro de documentação das configurações aprovadas: modelo, documentos, código;
  - *Status* das requisições de mudança;
  - Informações sobre *builds* e *releases*;
  - etc.

Atividades

- IEEE divide as funções da SCM nas quatro atividades seguintes:
  - 1. Identificação da Configuração
  - 2. Controle da Configuração
  - 3. Relato de Status
  - **4. Auditorias e Revisões**

Auditoria

- Tarefas:
  - Verificação funcional, assegurando que a configuração de referência cumpre o que foi especificado.
  - Verificação física, assegurando que a configuração de referência é completa (todos os itens de configuração especificados).
- Auditorias servem para garantir que os procedimentos e padrões foram aplicados.

Auditoria

- A auditoria ocorre antes de cada liberação, para verificar a configuração de referência de produto.
- Preferencialmente, auditoria deve ser efetuada por auditor externo e isento.
- Caso deseje efetuar internamente, a equipe de auditoria deve ser composta por representantes da gerência, garantia de qualidade e do cliente.

Auditoria

- A auditoria funcional ocorre através da revisão dos planos, dados, metodologia e resultado dos teste, para verificar se são satisfatórios.
- A auditoria física examina a estrutura de todos os itens de configuração que compõem a configuração de referência.
- A auditoria física é efetuada após a auditoria funcional.
- Pode ocorrer auditorias no próprio sistema de GC pelos mantenedores do plano de GC, para verificar se as políticas e procedimentos estão sendo cumpridos.

UNIPÊ
Pós-graduação

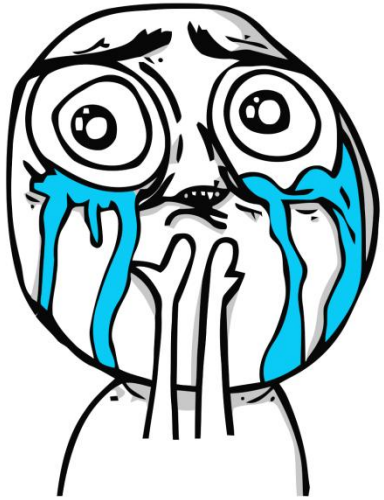
Ferramentas

git

CVS

SVN

SUBVERSION



Jenkins

TC

Buildbot

Travis CI

APACHE ANT

ivy

REDMINE

Gradle

Bugzilla

JIRA

Maven

Bower

GRUNT

npm

Trello