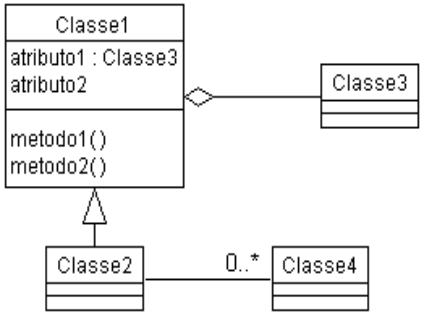
Gerenciamento de configuração

Leitura

- Livro: Engenharia de Software
- Autor: Summerville
- Capítulo Gerenciamento de configuração

Fases de um projeto de software

Elicitação/ Especificação de requisitos



Projeto

Codificação

Testes

Implantação

Manutenção

Desafios 1
Gerenciamento de configuração

- **Representar e gerenciar a relação entre os artefatos**
  - Qualquer coisa que vamos desenvolver em computação envolve um conjunto relacionado de artefatos: software ou web page
  - Desafio: como gerenciar (administrar) esta conjunto de artefatos e suas relações

Imagem.jpg

topo.php

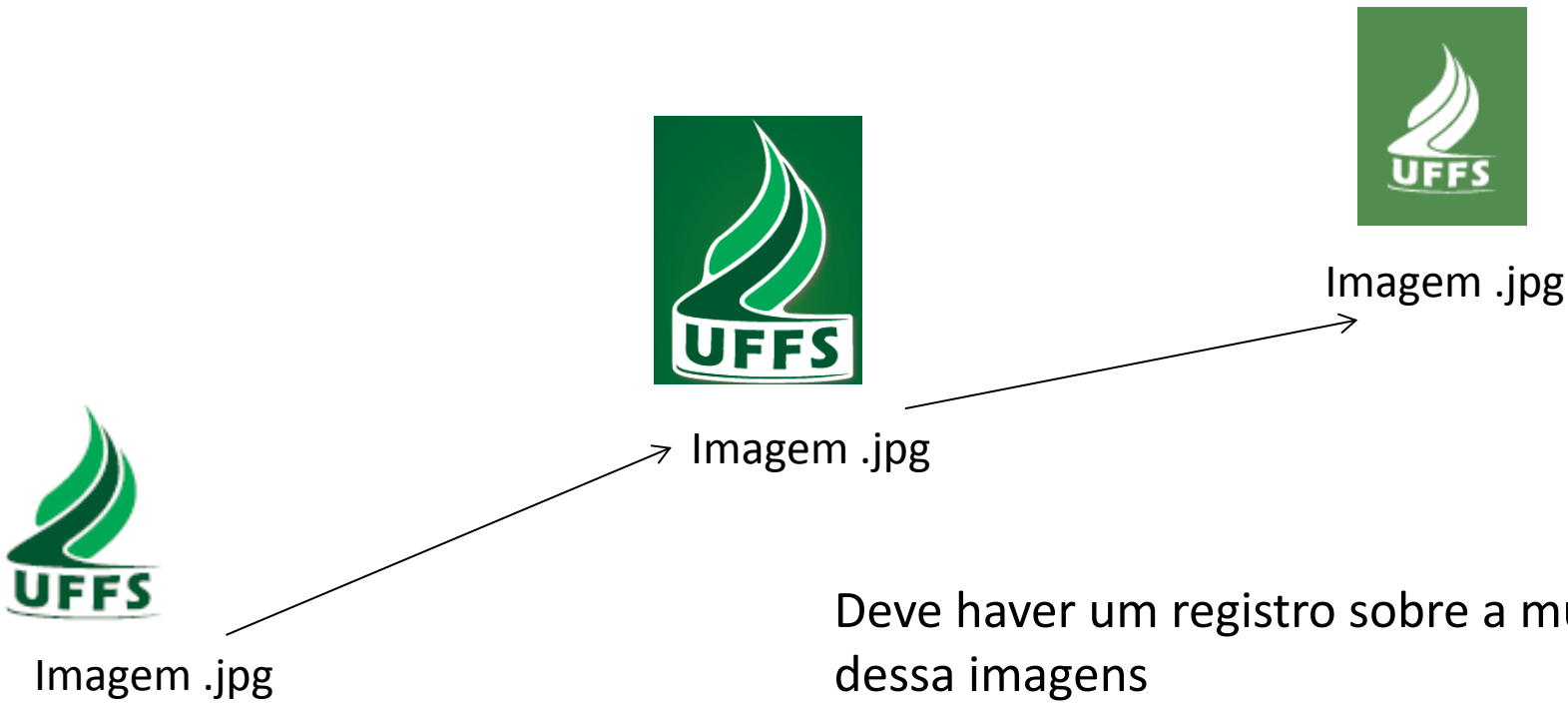
```
.slide-desc-bg {
        background: #334;
        opacity: 0.8;
        filter: alpha(opacity = 80);
}
.slide-desc-text {
        color: #fff;
        padding: 10px;
        text-align: left;
}
.slide-desc-text .slide-title {
        font-size: 1.5em;
        color: #eeee88;
        margin-bottom: 5px;
}
.slide-desc-text .slide-title a {
        color: #eeee88;
}
```

style.css

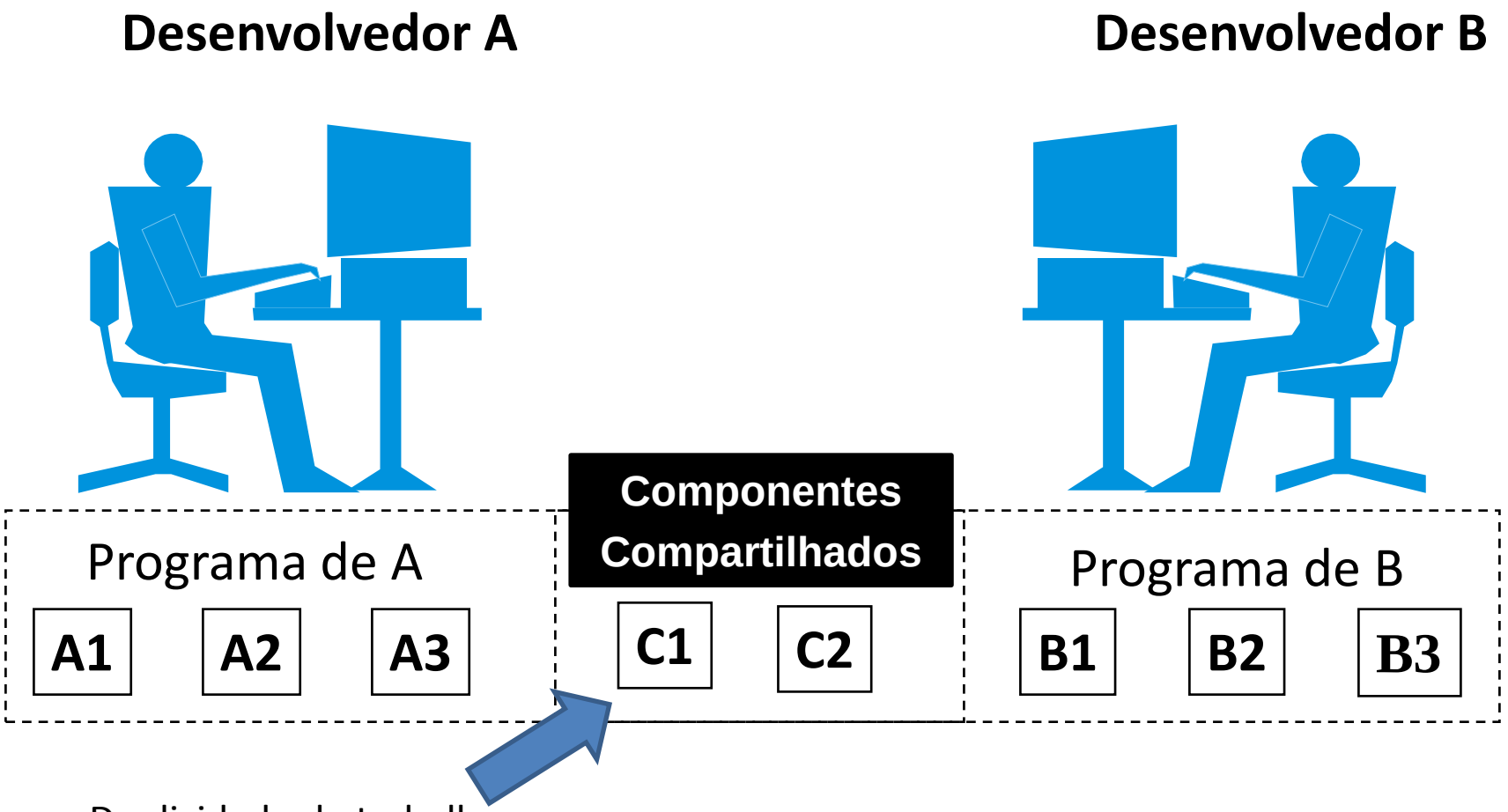
Pagina index.php

Desafio 2
Gerenciamento de versões

- **Acompanhar e gerenciar a evolução dos artefatos ao longo do tempo**
  - Os artefatos evoluem como o passar do tempo
  - Desafio é como acompanhar e gerenciar as alterações que vão acontecendo



Problemas

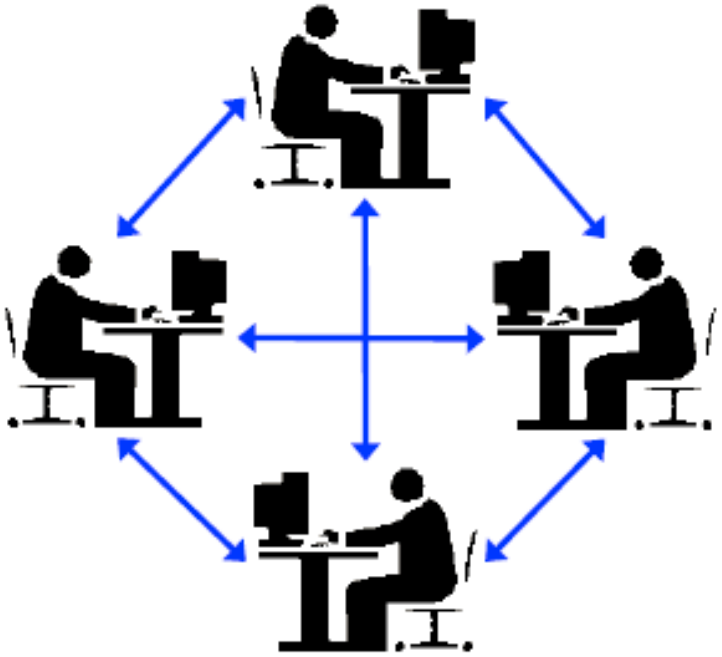


Duplicidade de trabalho
Arquivos sobrepostos
Trabalhos perdidos
Não tem como saber quem fez o que (histórico de mudanças)
Não é possível voltar numa versão anterior para corrigir um erro após a homologação

Problema dos dados compartilhados – Cenário comum

- O desenvolvedor A modifica o componente compartilhado
- Mais tarde, o desenvolvedor B realiza algumas alterações no mesmo componente
- Ao tentar compilar o componente, erros são apontados pelo compilador, mas nenhum deles ocorre na parte que B alterou
- O desenvolvedor B não tem a menor ideia sobre a causa do problema

Como surgiu a necessidade da Gerência de Configuração



- Normalmente se desenvolve software com várias pessoas trabalhando num mesmo projeto (equipe), em máquinas diferentes, mas que compartilham os mesmos códigos.
- Quanto maior o projeto maiores serão os problemas: maior será a complexidade do problema, mais código a ser gerado, mais pessoas gravando e compartilhando arquivos.

Problemas pela falta de GC

- Perda de código fonte
- Programas inesperadamente param de funcionar
  - Alguém sobre escreveu numa versão estável
- Impossível saber qual foi a evolução do desenvolvimento de um programa
- Impossível saber quem, porque e quando foram realizadas as alterações no sistema
- Uma classe simplesmente sumiu
- Bugs corrigidos aparecem inesperadamente

Como resolver esses problemas?

Como resolver esses problemas?

➔ Com uma boas práticas de Gerência de Configuração

Gerenciamento de configuração

- Gerência de configuração (GC) é o processo de **identificar**, **organizar** e **controlar** modificações ao software sendo construído (MPS.BR)
- É responsável por gerenciar como o software é modificado e construído através de técnicas que incluem **controle de mudanças**, **controle de versão**, **rastreabilidade na construção dos objetos** e **geração de versões de software**.

Por que GC?

Evolução do software → MUDANÇAS

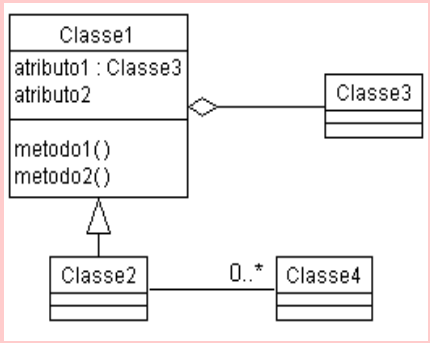
- 75% do custo total do ciclo de vida do software é com manutenção.
- 20% do tempo para consertar erros.
- 80% do tempo para modificações nos requisitos:
  - Requisitos funcionais
  - Regras de negócio
  - Reengenharia da aplicação.

- Em qualquer equipe é inevitável um certo grau de **confusão**

- O objetivo é maximizar a produtividade minimizando os erros (minimizar a confusão)

Em todas as fases do software possuem artefatos que evoluem

Elicitação/ Especificação de requisitos



Projeto

Codificação

Testes

Implantação

Manutenção

Configuração de software

- Um projeto de software produz vários artefatos:
  - Código fonte
  - Programas executáveis
  - Biblioteca de componentes
  - Dados dos testes
  - Documentação de requisitos
  - Scripts do banco de dados
  - Imagens, ícones ...
  - Documentação do projeto do software: casos de uso, diagramas de classes, diagrama entidade-relacionamento, etc

➔ Esse conjunto de itens são chamados de **itens de configuração de software**

➔ Quando se criar um **processo** de gerenciamento desses itens se chama de **gerenciamento de configuração**

Responsabilidades da Gerência de configuração

- Identificação dos itens de configuração
- Obter controle das alterações e das versões de cada um desses itens de configuração, de modo efetivo e contínuo
- Definir o ambiente de desenvolvimento (ambiente de desenvolvimento e de homologação)
- Política para controle de versões (como o software será versionado através de builds, releases, versões no decorrer do tempo)
- Garantir a integridade e rastreabilidade dessas versões desses itens
- Garantir a consistência dos artefatos – determina onde as coisas serão guardadas e de que forma
- Facilitar a integração entre as partes dos sistema
- Definir procedimentos para solicitação de mudança
- Todo o histórico de mudanças deve ser recuperável e auditável

Benefícios

- Aumento da disciplina no processo de desenvolvimento e da organização (eficiência do processo)
- Aumento de produtividade e de eficiência no desenvolvimento (evitando problemas de duplicidade de trabalho, sobreposição de arquivos, visualizar as mudanças)
- Redução de defeitos (consequência do ambiente organizado)
- Menores custos de manutenção (pois eu não se perde código fonte e não ocorre retrabalho)
- Maior rapidez na identificação e correção dos problemas (consequência dos outros itens)
- Uma vez que se tem um ambiente de desenvolvimento organizado possibilitando o crescimento do software de maneira organizada, ocorre o aumento da produtividade do desenvolvimento, em consequência irá reduzir os defeitos, etc.

Gerência de configuração

- Não é unicamente controle de versão
- Não é configuração de conteúdos/dados
- Não é backup

- Não é simples
- Não é impossível
- Não é opcional

O que é a gerencia de configuração

➔ Gerenciamento de configuração é muito mais que controlar as versões, é controlar as mudanças do projeto

- Gerencia de configuração é o processo de identificar, organizar e controlar as modificações de um software em construção
- É um fluxo de apoio para todo o projeto (não apenas de apoio para a etapa de desenvolvimento)

➔ É um dos tópicos mais importantes da Engenharia de software

Itens de configuração

É o que se deseja controlar ao longo de ciclo de vida de um projeto de software. É o menor item dentro da GC:

- Código fonte
- Programas executáveis
- Biblioteca de componentes
- Dados dos testes
- Documentação de requisitos
- Scripts do banco de dados
- Imagens, ícones ...
- Documentação do projeto do software: casos de uso, diagramas de classes, diagrama entidade-relacionamento, etc
- etc

➔ **É um engano pensar que a gerencia de configuração deve se preocupar apenas com os códigos fontes do sistema**

Responsabilidades da Gerência de configuração

- Definir o ambiente de desenvolvimento (ambiente de desenvolvimento, de homologação)
- Política para controle de versões (como o software será versionado através de builds, releases, versões no decorrer do tempo)
- Garantir a consistência dos artefatos – determina onde as coisas serão guardadas e de que forma
- Definir procedimentos para solicitação de mudança
- Administrar o ambiente e auditar mudanças
- Facilitar a integração entre as partes dos sistema

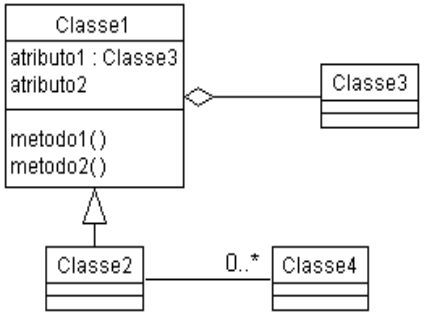
Porque o sistema mudou?

Quais foram as mudanças?

Gerenciamento de mudanças

Fases de um projeto de software

Elicitação/ Especificação de requisitos



Projeto

Codificação

Testes

Implantação

Manutenção

Gerenciamento de mudanças

- Durante a vida útil de um sistema as necessidades organizacionais e os requisitos desse mudam, *bugs* precisam ser reparados e os sistemas têm de se adaptar às mudanças em seu ambiente.

- O gerenciamento de mudanças **visa garantir que a evolução do sistema seja um processo gerenciado**

Gerenciamento de mudanças

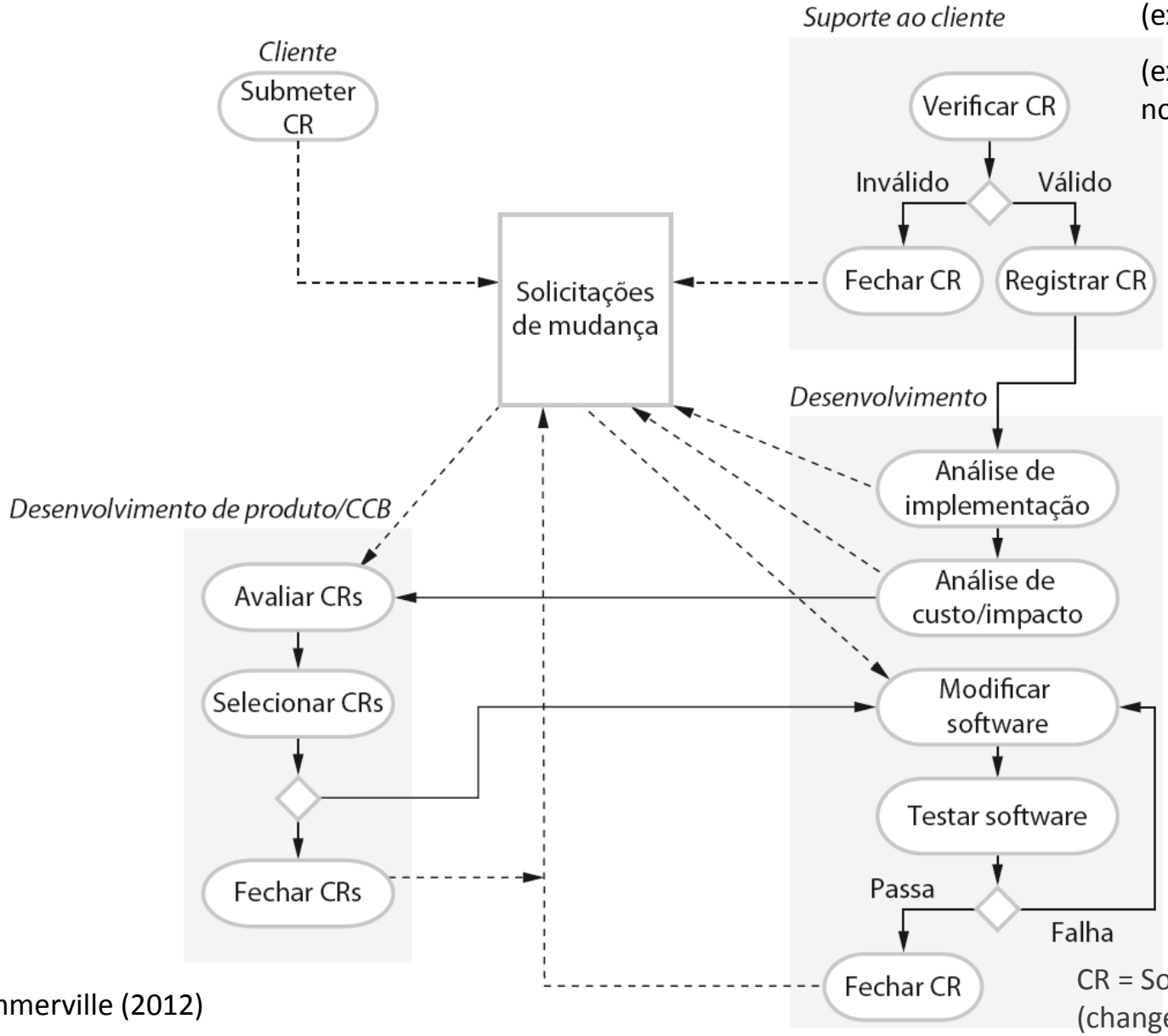
- O gerenciamento de mudanças **visa garantir que a evolução do sistema seja um processo gerenciado** e que **seja dada prioridade às mudanças mais urgentes e de custo-benefício.**

- O processo de gerenciamento de mudanças **está relacionado com a análise dos custos** e **benefícios das mudanças** propostas, a aprovação das **mudanças que valem a pena** e o **acompanhamento das alterações** nos componentes do sistema.

O processo de gerenciamento de mudanças segundo Sommerville

(ex. correção de um bug)

(ex. Solicitação de nova funcionalidade)



CR = Solicitação de mudança (change request)

Fonte: Sommerville (2012)

Fatores na análise de mudança

- Quais as consequências de não realizar a mudança
- Os benefícios da mudança
- O número de usuários afetados pela mudança (impacto nos clientes)
- Os custos de se fazer a mudança
- O ciclo de release de produto

Um formulário de solicitação de mudança parcialmente concluído (a)

Projeto: SICSA/AppProcessing
Número: 23/02
Solicitante de mudança: I. Sommerville
Data: 20/jan./2009
Mudança solicitada: O *status* dos requerentes (rejeitados, aceitos etc.) deve ser mostrado visualmente na lista de candidatos exibida.

Analista de mudança: R. Looek
Data da análise: 25/jan./2009
Componentes afetados: ApplicantListDisplay, StatusUpdater

Componentes associados: StudentDatabase

Avaliação de mudança: Relativamente simples de implementar, alterando a cor de exibição de acordo com ***status***. Uma tabela deve ser adicionada para relacionar ***status*** a cores. Não é requerida alteração nos componentes associados.

Um formulário de solicitação de mudança parcialmente concluído (b)

Prioridade de mudança: Média
Implementação de mudança:
Esforço estimado: 2 horas
Data para equipe de aplicação de SGA: 28/jan./2009
Data de decisão do CCB: 30/jan./2009
Decisão: Aceitar alterar. Mudança deve ser implementada no *Release* 1.2
Implementador de mudança:
Data de mudança:
Data de submissão ao QA:
Decisão de QA:
Data de submissão ao CM:
Comentários:

Ferramentas de controle de mudanças

Open source:

- Redmine
- Mantis
- Bugzilla
- Trac

Comerciais:

- Jira
- IBM Rational ClearQuest

Porque o sistema mudou?

Quais foram as mudanças?

Controle de versões

Gerenciamento de versões

- O **gerenciamento de versões** (VM – *Version Management*) **é o processo de manter o controle das diferentes versões dos componentes do software ou itens de configuração** e os sistemas em que esses componentes são usados.
- Também envolve **assegurar que as mudanças sejam feitas por desenvolvedores diferentes** para que **essas versões não interfiram umas com as outras**.

Histórico de derivação

```
/ / SICSA project (XEP 6087)
//
/ / APP-SYSTEM/AUTH/RBAC/USER_ROLE
//
/ / Objeto: currentRole
/ / Autor: R. Looek
/ / Data de criação: 13/11/2009
//
/ / © Universidade ST. Andrews 2009
//
/ / Histórico de modificações
```

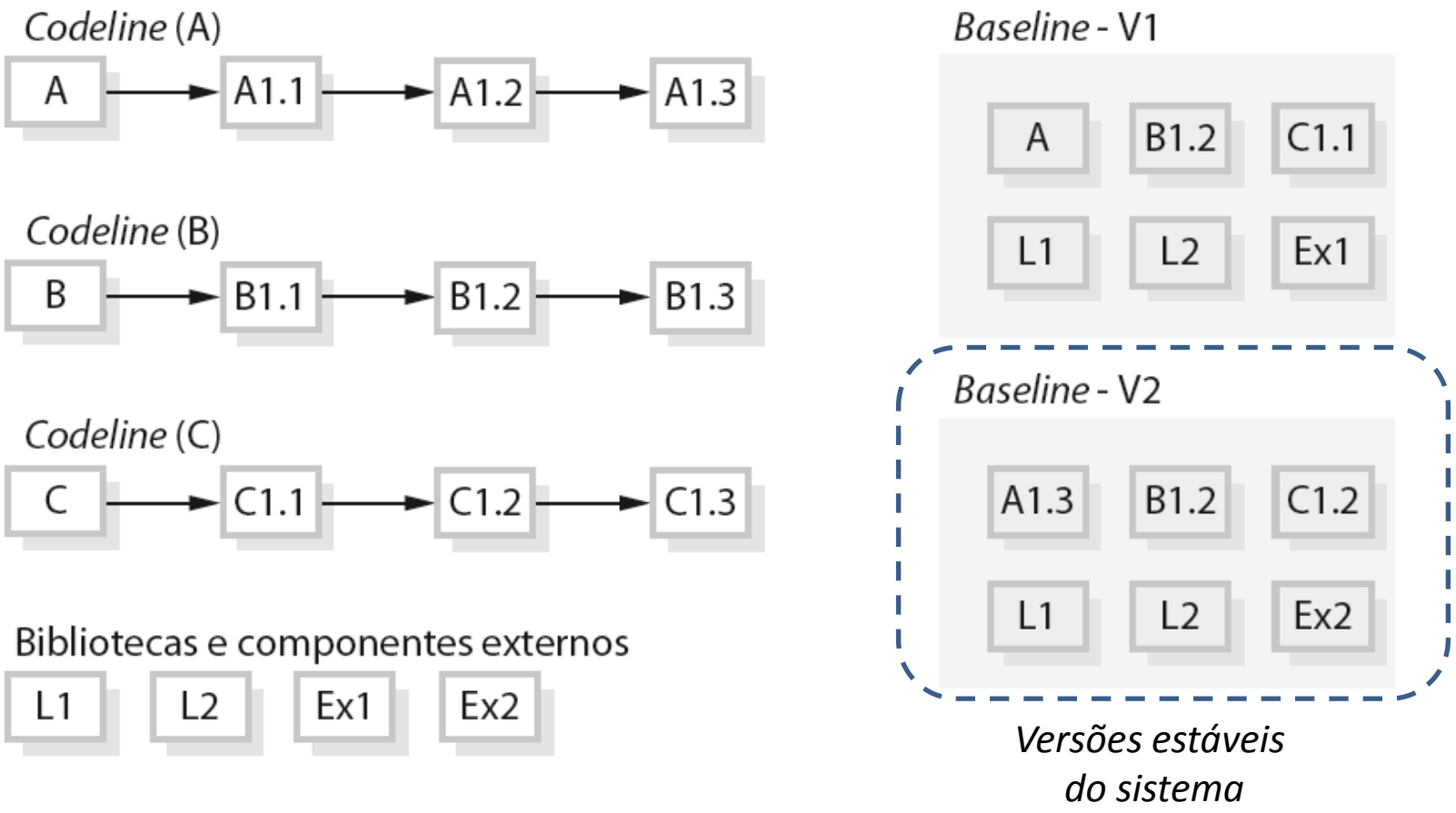
/ / Versão	Modificador	Data	Mudança	Razão
/ / 1.0	J. Jones	11/11/2009	Adicionar cabeçalho	Submetido ao CM
/ / 1.1	R. Looek	13/11/2009	Novo campo	Solicitação de mudança R07/02

Fonte: Summerville (2011)

Codelines e Baselines

- O gerenciamento de versões pode ser pensado como o processo de gerenciamento de ***codelines*** e ***baselines***.
- Um **codeline** é uma sequência de versões de código-fonte com as versões posteriores na sequência derivadas de versões anteriores.
- **Codelines** normalmente se aplicam a componentes de sistemas de modo que existem diferentes versões de cada componente.
- Um **baseline** é uma definição de um sistema específico.
- Um **baseline**, portanto, especifica as versões dos componentes que estão incluídos no sistema além de uma especificação das bibliotecas usadas, arquivos de configuração, etc.

Codelines e Baselines



Versões estáveis do sistema

Baseline

- **Baseline (linha de base)**:
  - É a configuração do software em um ponto do tempo (faz uma marco como uma bandeira em determinados momento do tempo do projeto, ou como tirar uma foto para registrar a situação num determinado momento)
  - As baselines representam conjuntos de itens de configuração formalmente aprovados que servem de base para as etapas seguintes de desenvolvimento
  - Serve como base para os passos posteriores de desenvolvimento
  - Baselines são considerados marcos no processo de desenvolvimento
  - **Definição: é uma coleção de versões de componentes que compõe um sistema**.

- **Razões para criar uma baseline**:
  - Reproducibilidade: A habilidade de reproduzir uma versão anterior do sistema (analogia de voltar o backup)
  - Rastreabilidade: estabelece uma relação predecessor-sucessor entre os artefatos do projeto (analogia de percorrer o caminho até chegar a situação atual)
  - Controle de mudanças: referencial para comparações, discussões e negociações (entender as decisões tomadas ao autorizar a mudança)

Build

- Representa alguma **versão incompleta** do sistema em desenvolvimento, **mas com certa estabilidade** (não é uma entrega, mas uma versão menor até chegar na entrega final), conforme evoluímos (cada passo) no desenvolvimento vamos criando biulds, quando concluir tudo então vira uma release.
- Costuma representar limitações conhecidas (não precisa estar perfeito pois não é uma entrega para o cliente, mas sim um passo do avanço do desenvolvimento)
- Espaço para integração de funcionalidades
- Incluem não só código fonte, mas documentação, arquivos de configuração, bases de dados, etc;
- A política de geração de builds deve ser definida (ex. gerar build uma ou duas vezes por semana);

➔ A geração de builds deve ser automatizada e realizada com a frequência adequada

Releases

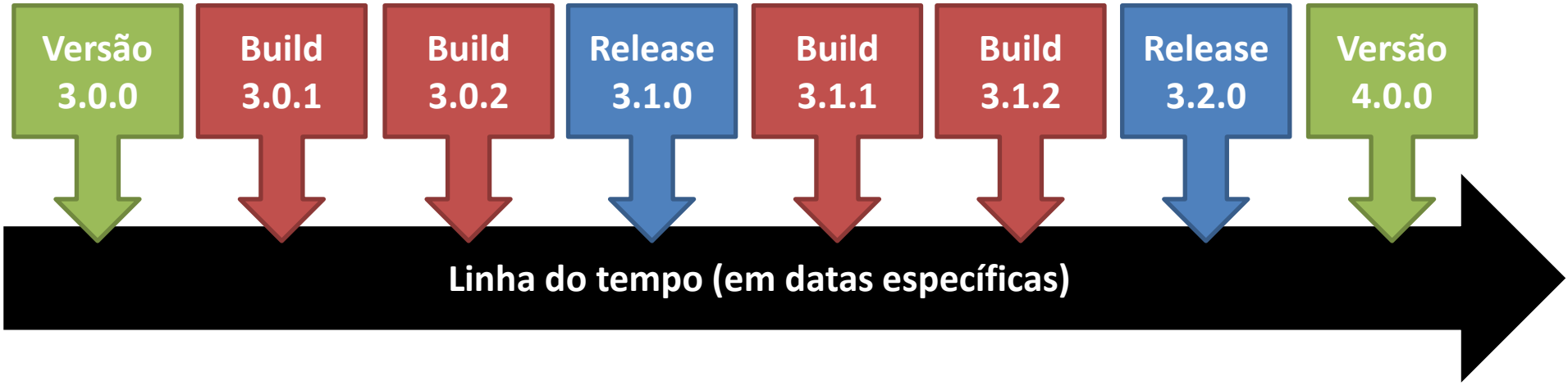
- Versão de um sistema validada após os diversos tipos de teste
- Produto de software
- Supostamente sem erros (está pronto e foi testado)
- Entre ao cliente ou ao mercado (é uma entrega)
- Implantado no cliente

➔Diferença: release deve funcionar corretamente, builds não está completo e pode haver problemas

Versão

- Uma política de evolução do produto
- Evoluiu tanto que se tornou uma nova versão

Builds, Releases e Versões



Três níveis de gerenciamento de configuração

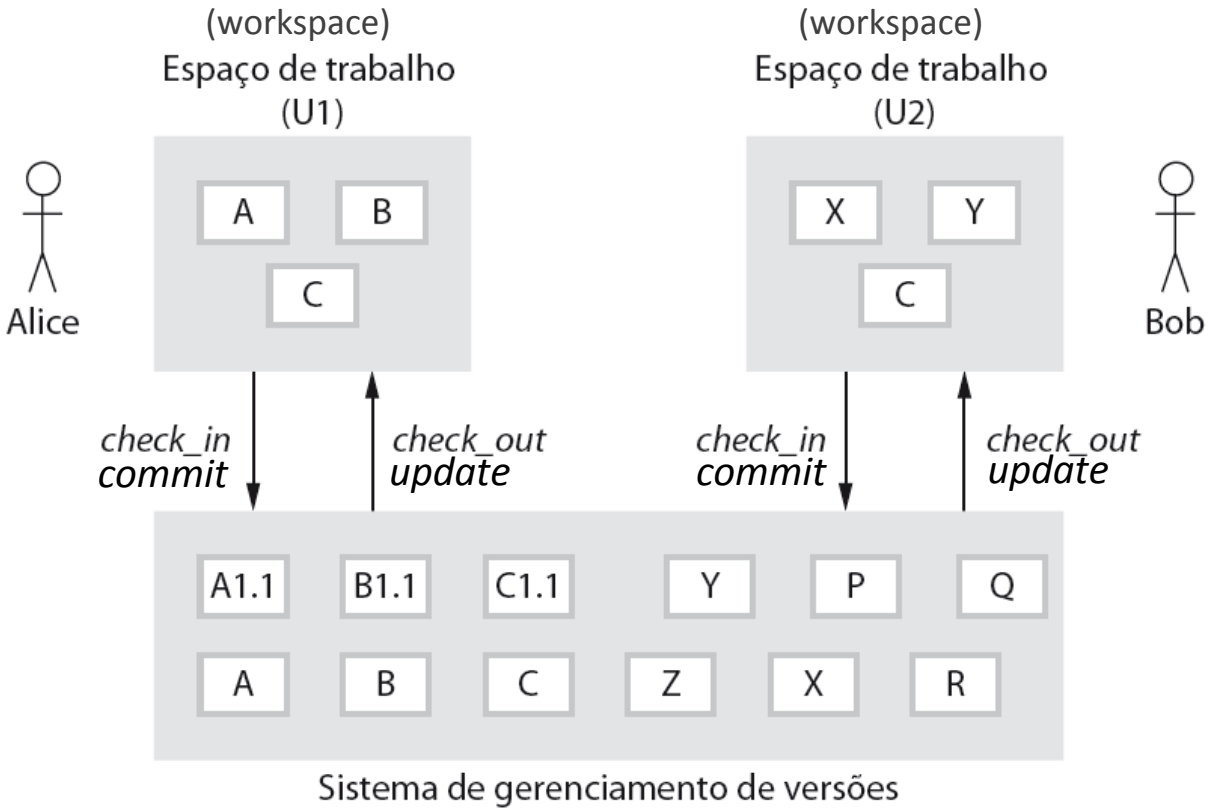
3.1.2 → Versão 3

3.**1**.2 → Release 1

3.1.**2** → Build 2

Check-in (commit) e *check-out (update)* a partir de um repositório versões

- Para apoiar no desenvolvimento independente sem influencia os sistemas de controle de versão utilizam o conceito de repositório público e espaço de trabalho privado

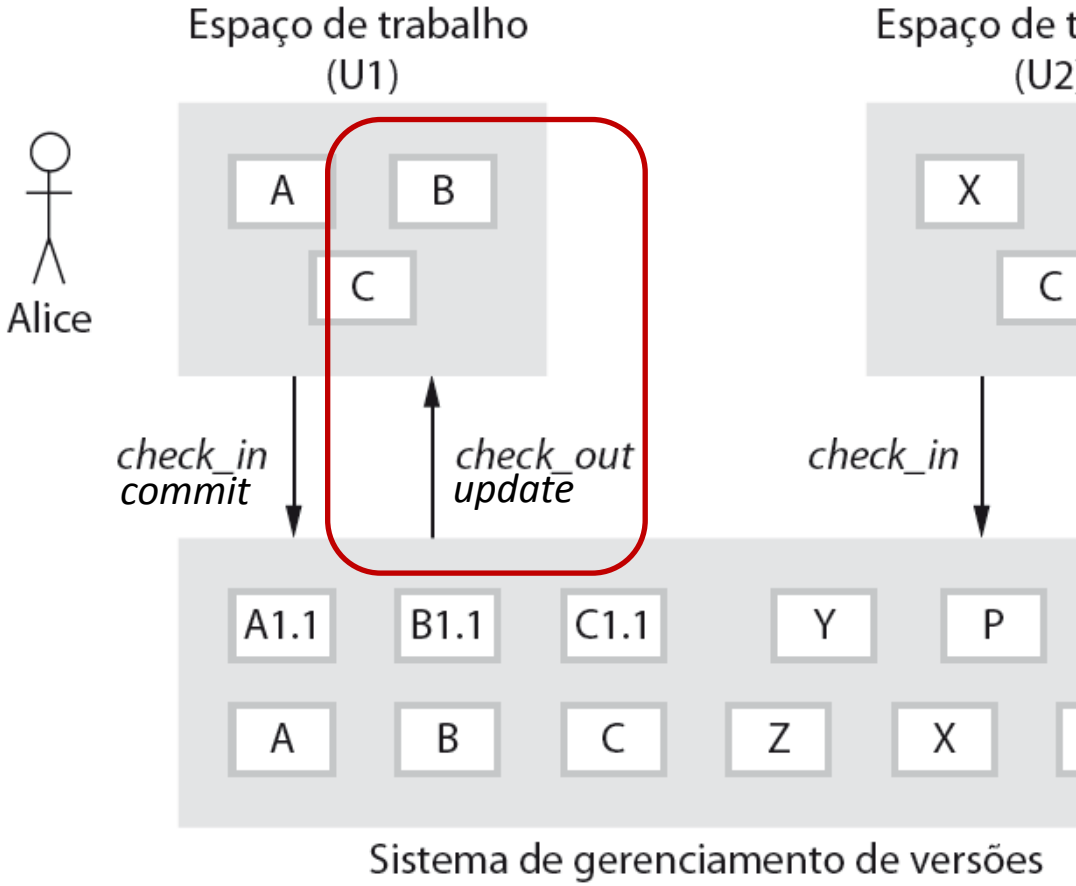


Espaço de Trabalho (Workspace)

- Lugar onde o desenvolvedor pode trabalhar isoladamente nos seus artefatos enquanto ele finaliza uma tarefa sem sofrer interferências externas.

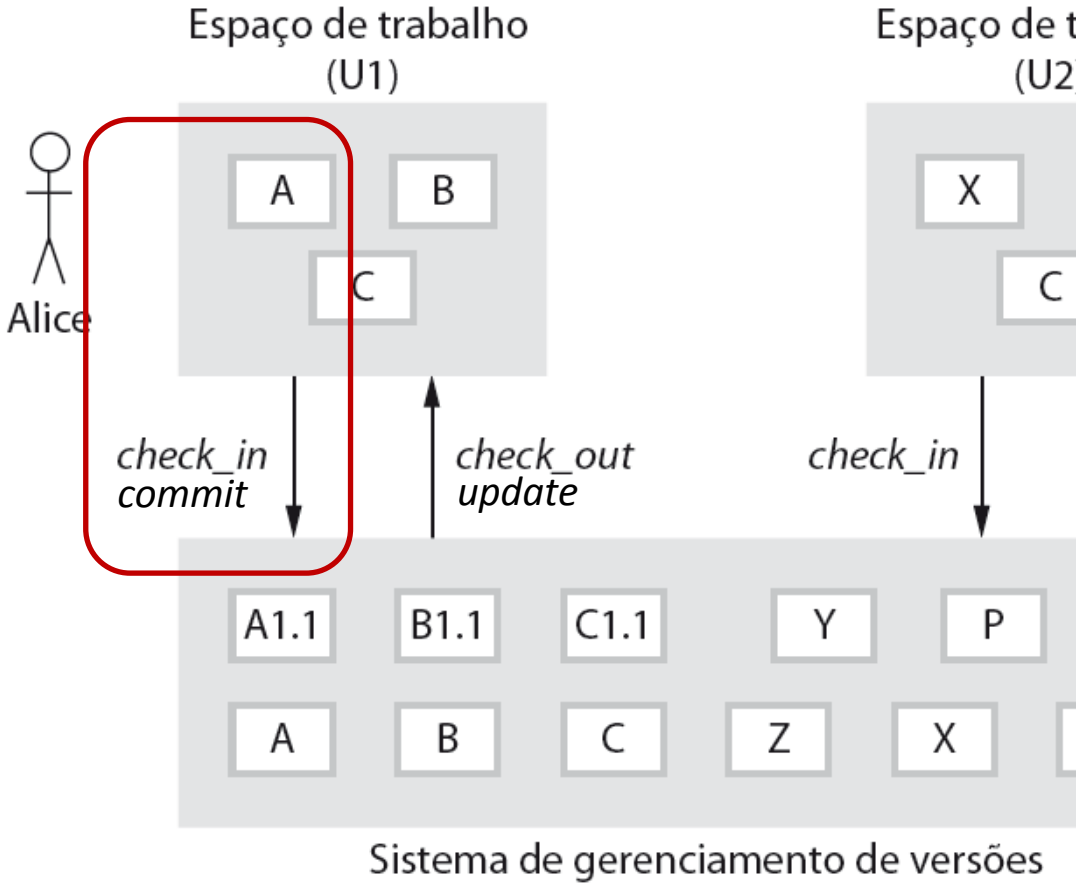
Check-out / update

- Cópia dos componentes de um repositório público para o espaço de trabalho privado
- Clone do diretório de trabalho

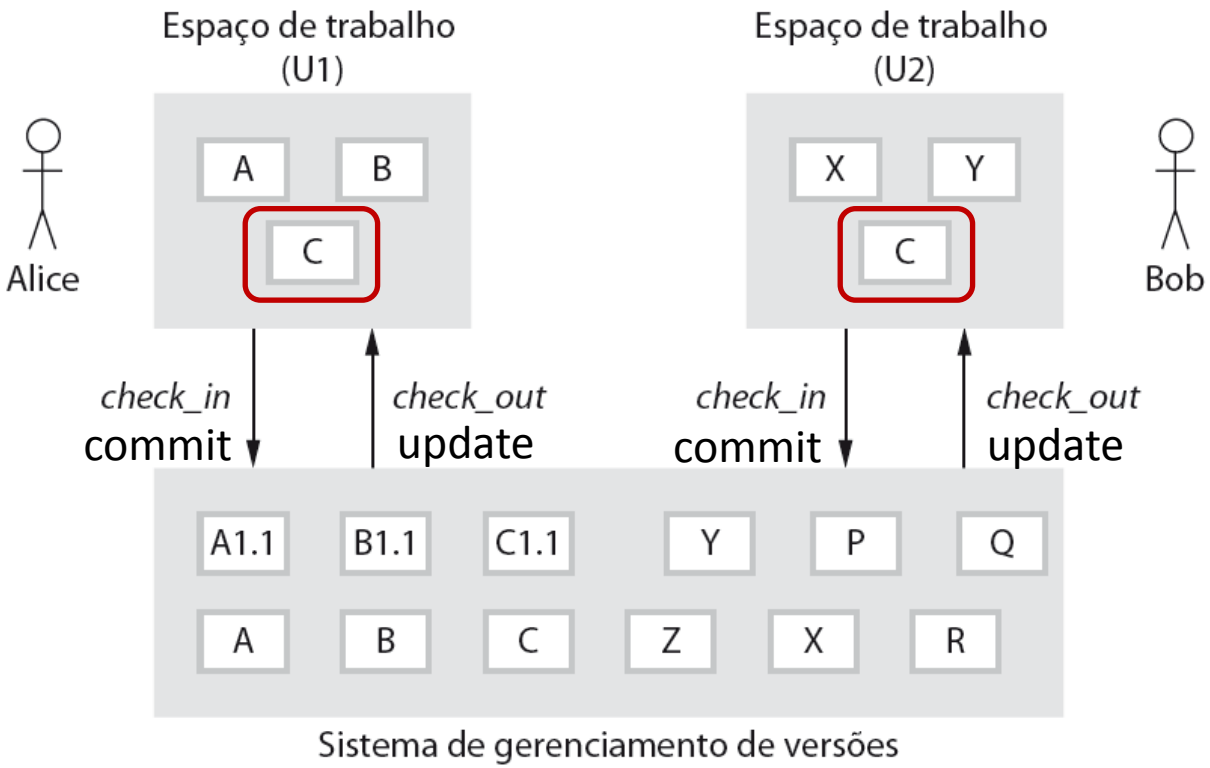


Check-in / commit

- Quando as mudanças forem concluídas eles realizam check-in dos componentes para o repositório

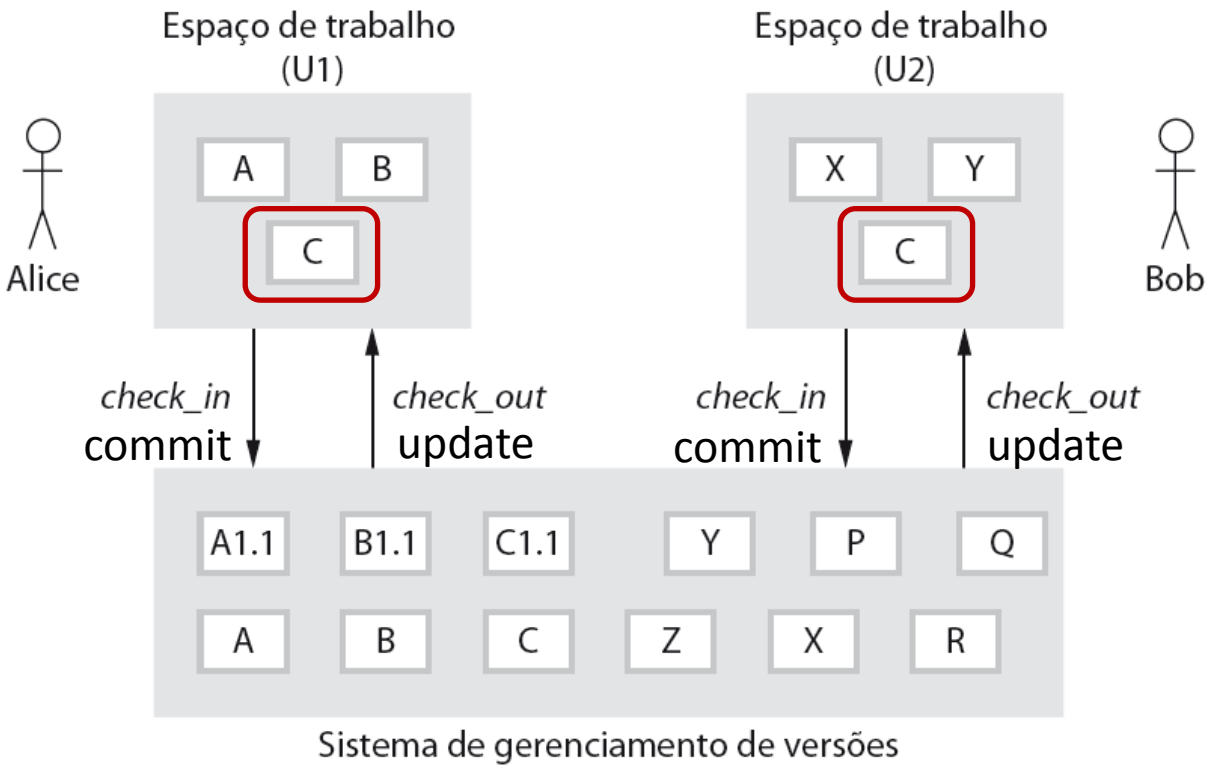


Como fazer quando há a necessidade de atualização simultânea de algum item de configuração ?



➔ Quando não se deseja que 2 desenvolvedores alterem simultaneamente o mesmo arquivo

Como fazer quando há a necessidade de atualização simultânea de algum item de configuração ?



➔ Quando 2 desenvolvedores podem alterar simultaneamente o mesmo arquivo

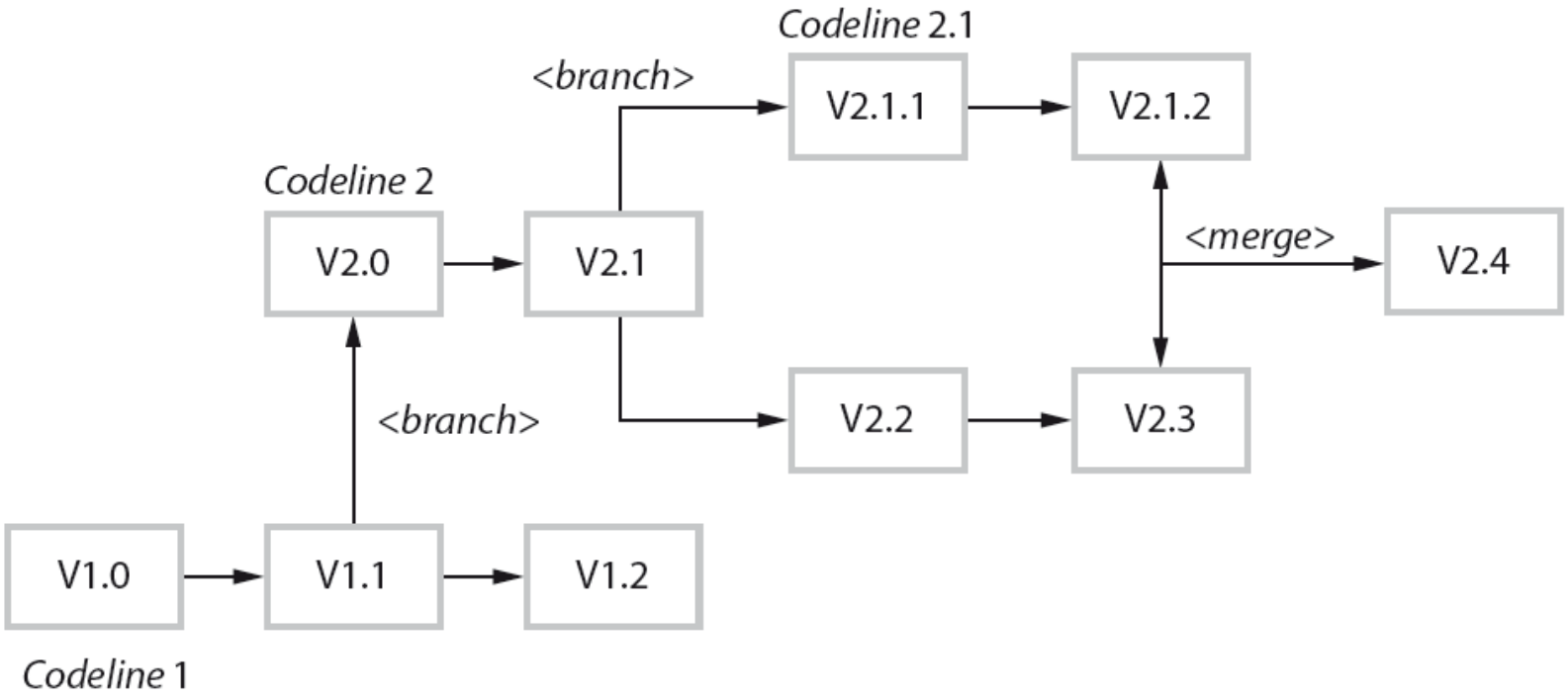
Branch (ramificação)

- Consiste em criar ramos paralelos para o desenvolvimento de um mesmo item de configuração
- Criação de um fluxo alternativo para atualização de versões de itens de configuração
- Devem existir regras bem definidas para criação de *branches*

Merge (junção)

- Quando não for optado pelo bloqueio o sistema de versões irá unificar as diferentes versões de um mesmo item de configuração
- Consistem em juntar as alterações feitas por vários usuários
- Operação de integração de alterações de um determinado *branch* com outro *branch*

Branching e merging



Pontos importantes

- Gerenciamento de configuração é o gerenciamento de um sistema de software em evolução.
- Durante a manutenção de um sistema, uma equipe GC é responsável para garantir que as mudanças são incorporadas ao sistema de uma forma controlada e que os registros são mantidos com os detalhes das mudanças que foram implementadas.
- Os principais processos de gerenciamento de configuração são gerenciamento de mudanças e gerenciamento de versões

Pontos importantes

- Gerenciamento de mudanças envolve avaliar propostas de mudanças do sistema de clientes e outros *stakeholders* e decidir se é efetivo implementá-las em um novo release de um sistema.
- Gerenciamento de versões envolve manter o acompanhamento das diferentes versões de componentes de software como as mudanças são feitas.