Capítulo 25

Gerenciamento de Configuração

Tópicos abordados

engenharia de SOFTWARE

- Gerenciamento de mudanças
- Gerenciamento de versões
- Construção de sistemas
- Gerenciamento de *releases*

Gerenciamento de configuração

- Porque os softwares mudam frequentemente, os sistemas podem ser pensados como um conjunto de versões, e cada qual precisa ser mantida e gerenciada.

- Versões implementam propostas de mudanças, correções de defeitos, e adaptações de hardware e sistemas operacionais diferentes.

- O gerenciamento de configuração (CM – Configuration Management) se interessa pelas políticas, processos e ferramentas para o gerenciamento de sistemas de software que sofrem mudanças. Você precisa de CM porque é fácil perder a noção de quais mudanças e versões de componentes foram incorporadas em cada versão do sistema.

Atividades de gerenciamento de configuração

engenharia de SOFTWARE

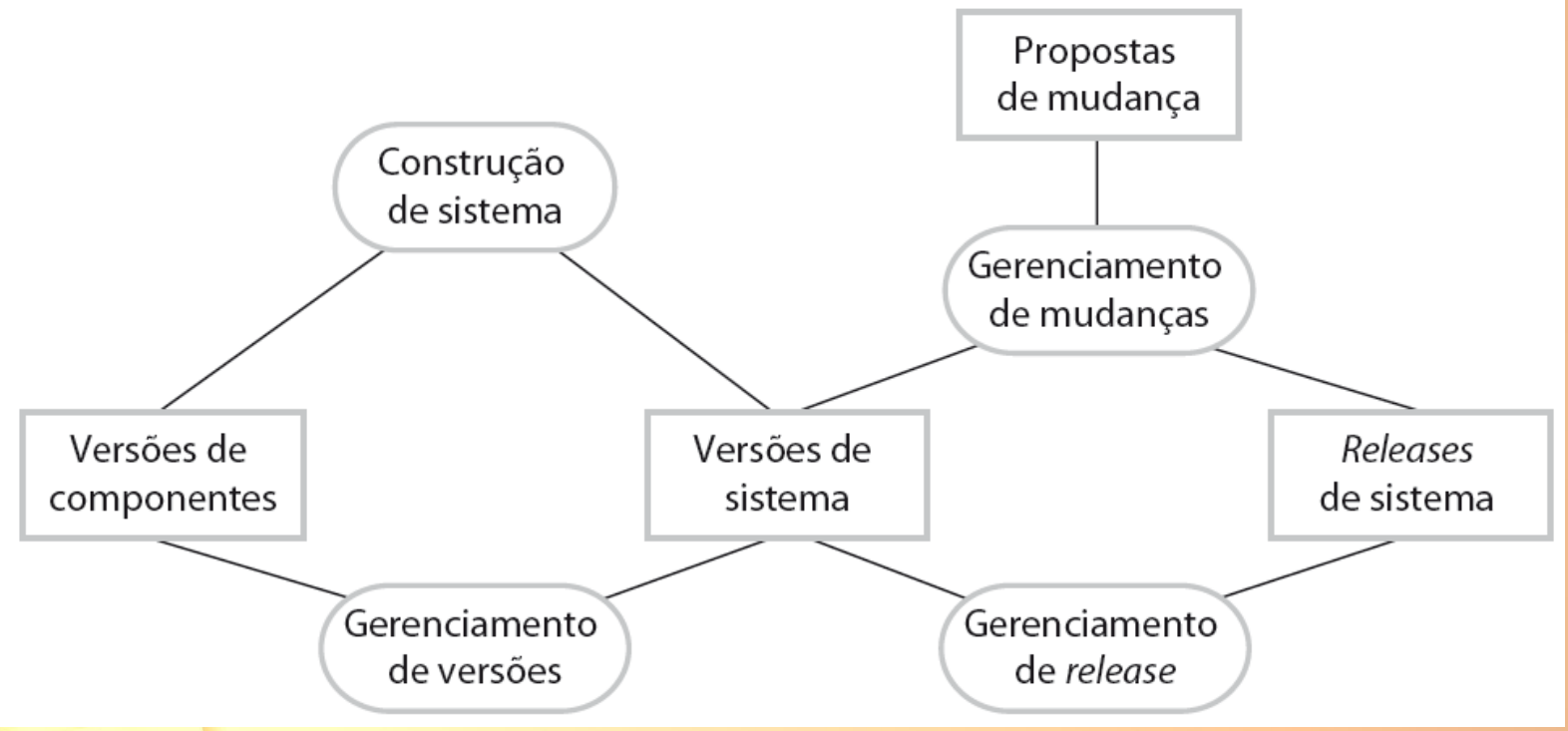
- Gerenciamento de mudanças
  - ✓ Manter o acompanhamento das solicitações de mudanças no software dos clientes e desenvolvedores, definir os custos e o impacto das mudanças, e decidir quais mudanças devem ser implementadas.
- Gerenciamento de versões
  - ✓ Manter o controle das múltiplas versões de componentes do sistema e assegurar que as alterações feitas aos componentes por diferentes desenvolvedores não interfiram umas com as outras.

Atividades de gerenciamento de configuração

engenharia de SOFTWARE

- Construção do sistema
  - ✓ O processo de montagem dos componentes de programa, dados e bibliotecas, e em seguida, a compilação desses para criar um sistema executável.
- Gerenciamento de releases
  - ✓ Preparar o software para release externo e manter o acompanhamento das versões do sistema que foram liberadas para uso pelo cliente.

Atividades de gerenciamento de configuração



Terminologia de CM

engenharia de SOFTWARE

Termo	Explicação
Item de configuração ou item de configuração de software (SCI, do inglês *software configuration item*)	Qualquer coisa associada a um projeto de software (projeto, código, dados de teste, documentos etc.) que tenha sido colocado sob controle de configuração. Muitas vezes, existem diferentes versões de um item de configuração. Itens de configuração têm um nome único.
Controle de configuração	O processo de garantia de que versões de sistemas e componentes sejam registradas e mantidas para que as mudanças sejam gerenciadas e todas as versões de componentes sejam identificadas e armazenadas por todo o tempo de vida do sistema.
Versão	Uma instância de um item de configuração que difere, de alguma forma, de outras instâncias desse item. As versões sempre têm um identificador único, o qual é geralmente composto pelo nome do item de configuração mais um número de versão.
Baseline	Uma *baseline* é uma coleção de versões de componentes que compõem um sistema. As *baselines* são controladas, o que significa que as versões dos componentes que constituem o sistema não podem ser alteradas. Isso significa que deveria sempre ser possível recriar uma *baseline* a partir de seus componentes.
Codeline	Uma *codeline* é um conjunto de versões de um componente de software e outros itens de configuração dos quais esse componente depende.

Terminologia de CM

Termo	Explicação
Mainline	Trata-se de uma sequência de *baselines* que representam diferentes versões de um sistema.
Release	Uma versão de um sistema que foi liberada para os clientes (ou outros usuários em uma organização) para uso.
Espaço de trabalho	É uma área de trabalho privada em que o software pode ser modificado sem afetar outros desenvolvedores que possam estar usando ou modificando o software.
Branching	Trata-se da criação de uma nova *codeline* de uma versão em uma *codeline* existente. A nova *codeline* e uma *codeline* existente podem, então, ser desenvolvidas independentemente.
Merging	Trata-se da criação de uma nova versão de um componente de software, fundindo versões separadas em diferentes *codelines*. Essas *codelines* podem ter sido criadas por um *branch* anterior de uma das *codelines* envolvidas.
Construção de sistema	É a criação de uma versão de sistema executável pela compilação e ligação de versões adequadas dos componentes e bibliotecas que compõem o sistema.

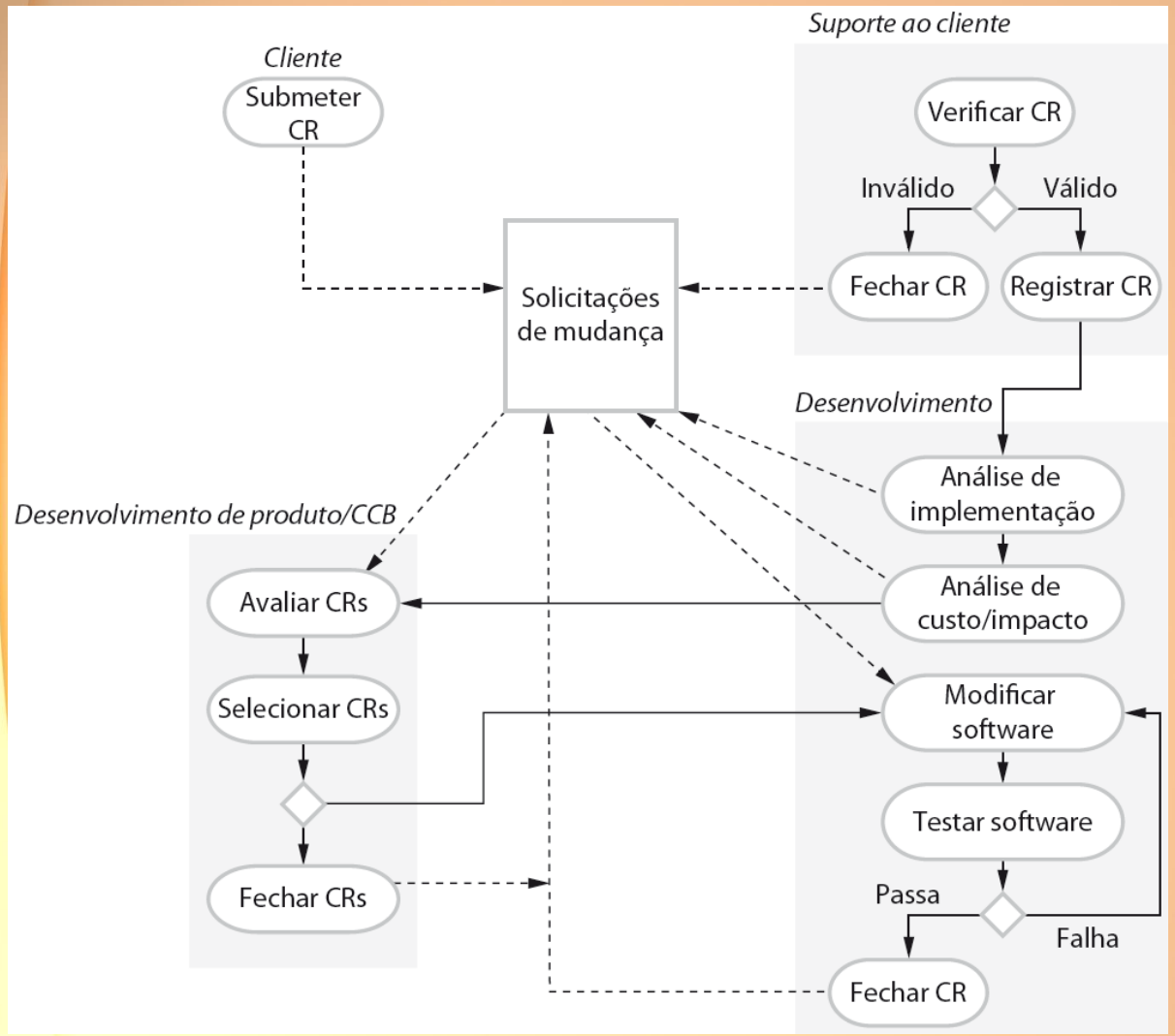
Gerenciamento de mudanças

- 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 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



Um formulário de solicitação de mudança parcialmente concluído (a)

engenharia de SOFTWARE

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)

engenharia de SOFTWARE

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:

Fatores na análise de mudança

engenharia de SOFTWARE

- As consequências de não fazer a mudança
- Os benefícios da mudança
- O número de usuários afetados pela mudança
- Os custos de se fazer a mudança
- O ciclo de release de produto

Gerenciamento de mudanças e métodos ágeis

- Em alguns métodos ágeis, os clientes estão diretamente envolvidos no gerenciamento de mudanças.
- Propor uma mudança nos requisitos e trabalhar com a equipe para avaliar o seu impacto e decidir se a mudança deve ter prioridade sobre os recursos planejados para o próximo incremento do sistema.
- As mudanças para melhorar o software são decididas pelos programadores que trabalham no sistema.
- A refatoração, em que o software é melhorado continuamente, não é vista como uma sobrecarga, mas como uma parte necessária do processo de desenvolvimento.

Histórico de derivação

engenharia de SOFTWARE

```
/ / 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
```

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.

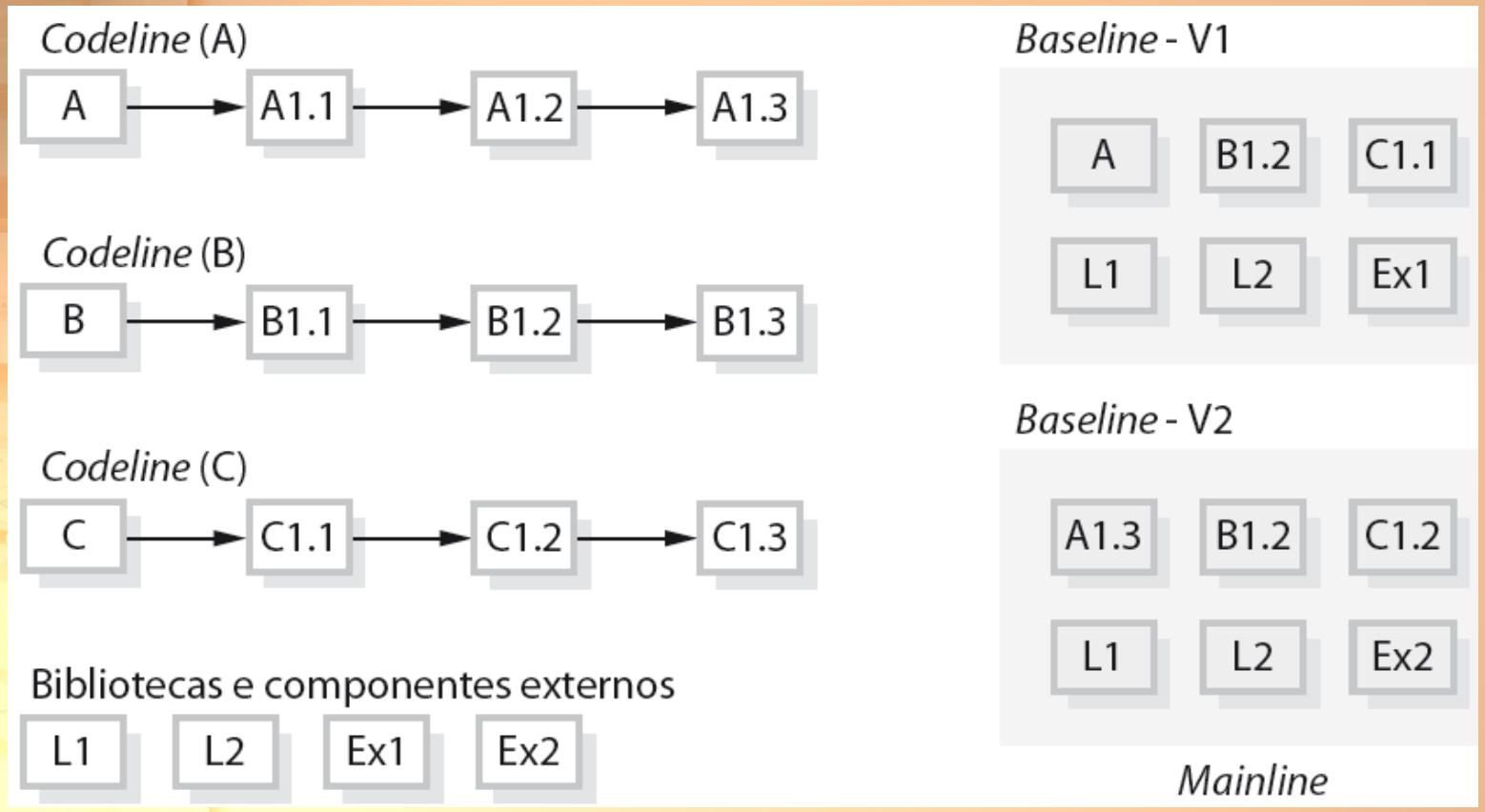
- Também envolve assegurar que as mudanças sejam feitas por desenvolvedores diferentes para que essas versões não interfiram umas com as outras.

- Portanto, o gerenciamento de versões pode ser pensado como o processo de gerenciamento de *codelines* e *baselines*.

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



Baselines

- *Baselines* podem ser especificados usando uma linguagem de configuração, que permite definir quais componentes estão incluídos em uma versão de um sistema particular.

- *Baselines* são importantes porque muitas vezes você tem de recriar uma versão específica de um sistema completo.

  - ✓ Por exemplo, uma linha de produtos pode ser instanciada para que haja versões de sistemas individuais para diferentes clientes. Você pode ter que recriar a versão entregue a um cliente específico, se, por exemplo, que o cliente relata defeitos em seu sistema que tem de ser reparados.

Sistemas de gerenciamento de versões

- Identificação de versão e release
  - ✓ Versões gerenciadas recebem identificadores quando são submetidos ao sistema.

- Gerenciamento de armazenamento
  - ✓ Para reduzir o espaço de armazenamento exigido por múltiplas versões de componentes que diferem apenas ligeiramente, sistemas de gerenciamento de versões geralmente oferecem recursos de gerenciamento de armazenamento.

- Registro de histórico de mudanças
  - ✓ Todas as mudanças feitas no código de um sistema ou componente são registradas e listadas.

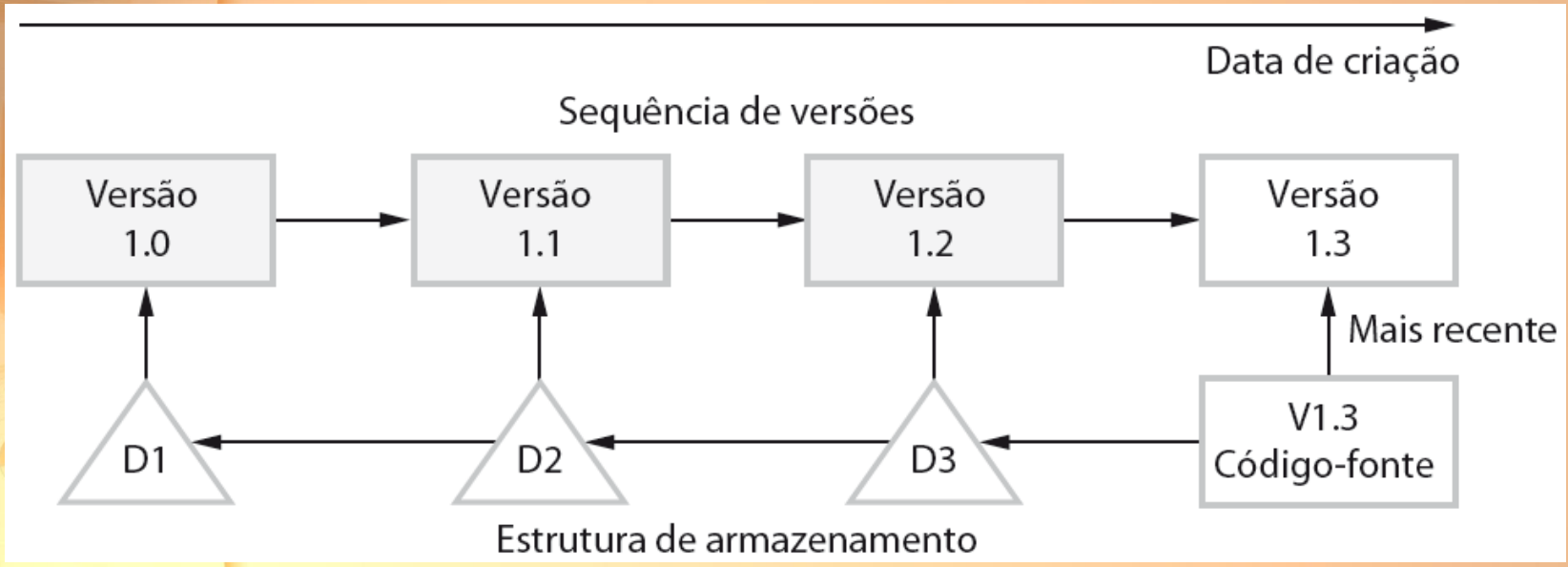
Sistemas de gerenciamento de versões

engenharia de SOFTWARE

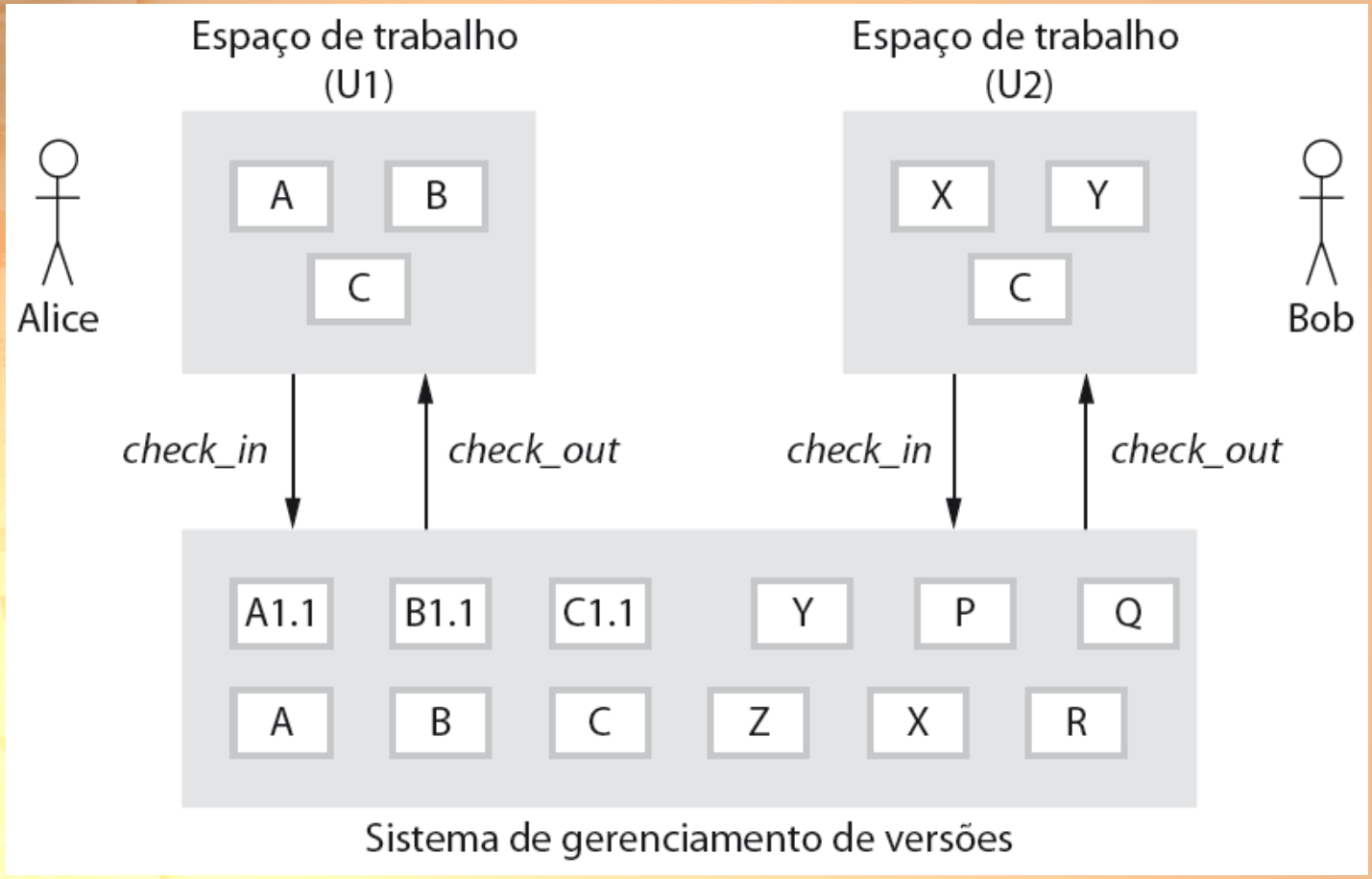
- Desenvolvimento independente
  - ✓ O sistema de gerenciamento de versões mantém o acompanhamento de componentes que foram retirados para edição e garante que as mudanças feitas em um componente por diferentes desenvolvedores não interfiram.
- Suporte a projetos
  - ✓ Um sistema de gerenciamento de versões pode apoiar o desenvolvimento de vários projetos que compartilham componentes.

Gerenciamento de armazenamento usando deltas

engenharia de SOFTWARE



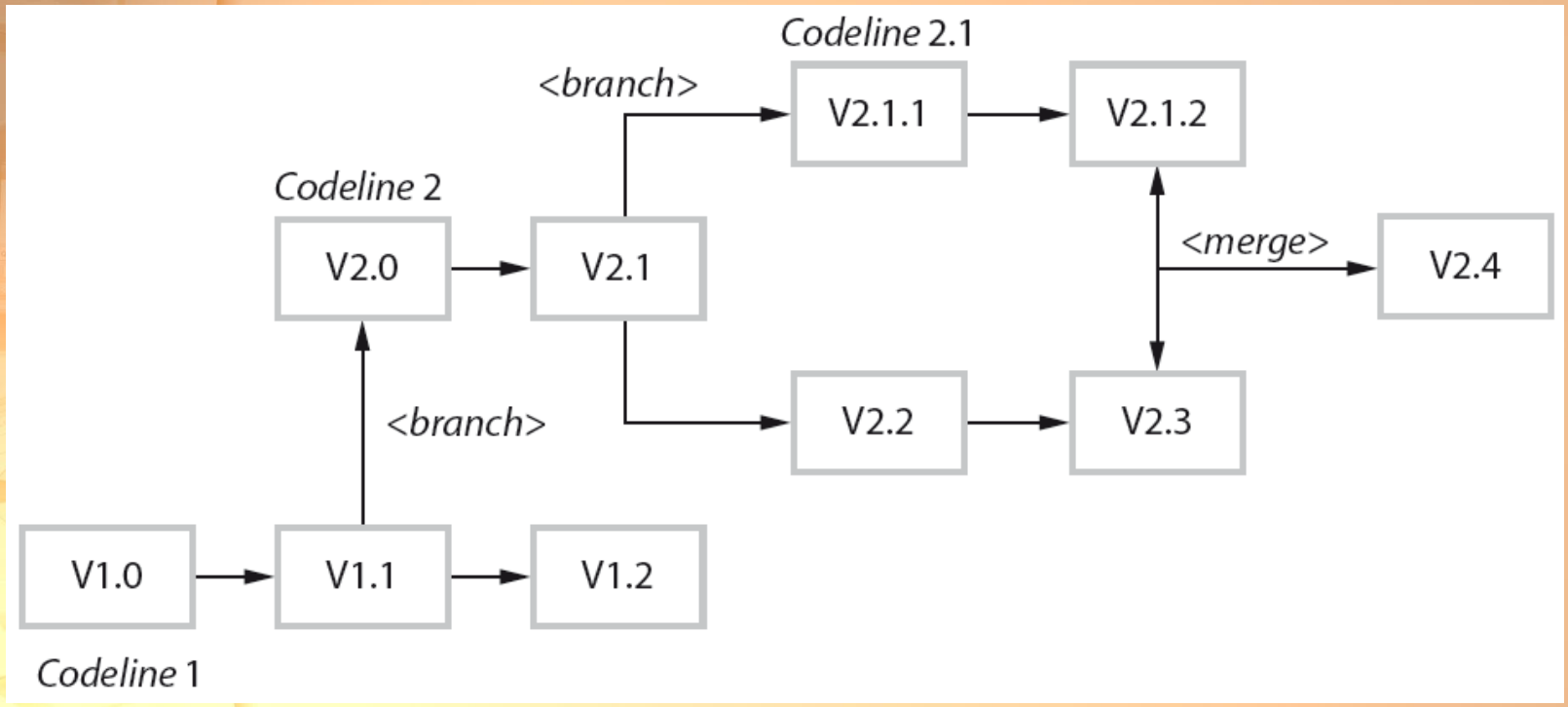
Check-in e *check-out* a partir de um repositório versões



Ramificações de *codelines*

- Em vez de uma sequência linear de versões que refletir as mudanças para o componente ao longo do tempo, pode haver várias sequências independentes.
  - ✓ Isso é normal no sistema em desenvolvimento, onde os desenvolvedores diferentes trabalham de forma independente em diferentes versões do código-fonte e assim, alterá-las em diferentes maneiras.
- Em algum momento, pode ser necessário fundir ramificações de *codelines* para criar uma nova versão de um componente que inclui todas as mudanças que foram feitas.
  - ✓ Se as mudanças feitas envolvem diferentes partes do código, as versões do componente podem ser fundidas automaticamente, combinando os deltas que se aplicam ao código.

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 CM 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, gerenciamento de versões, a construção de sistemas e o gerenciamento de releases.

- 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.

Construção de sistemas

- Construção de sistemas é o processo de criação de um sistema completo e executável por compilar e ligar os componentes do sistema, bibliotecas externas, arquivos de configuração, etc.

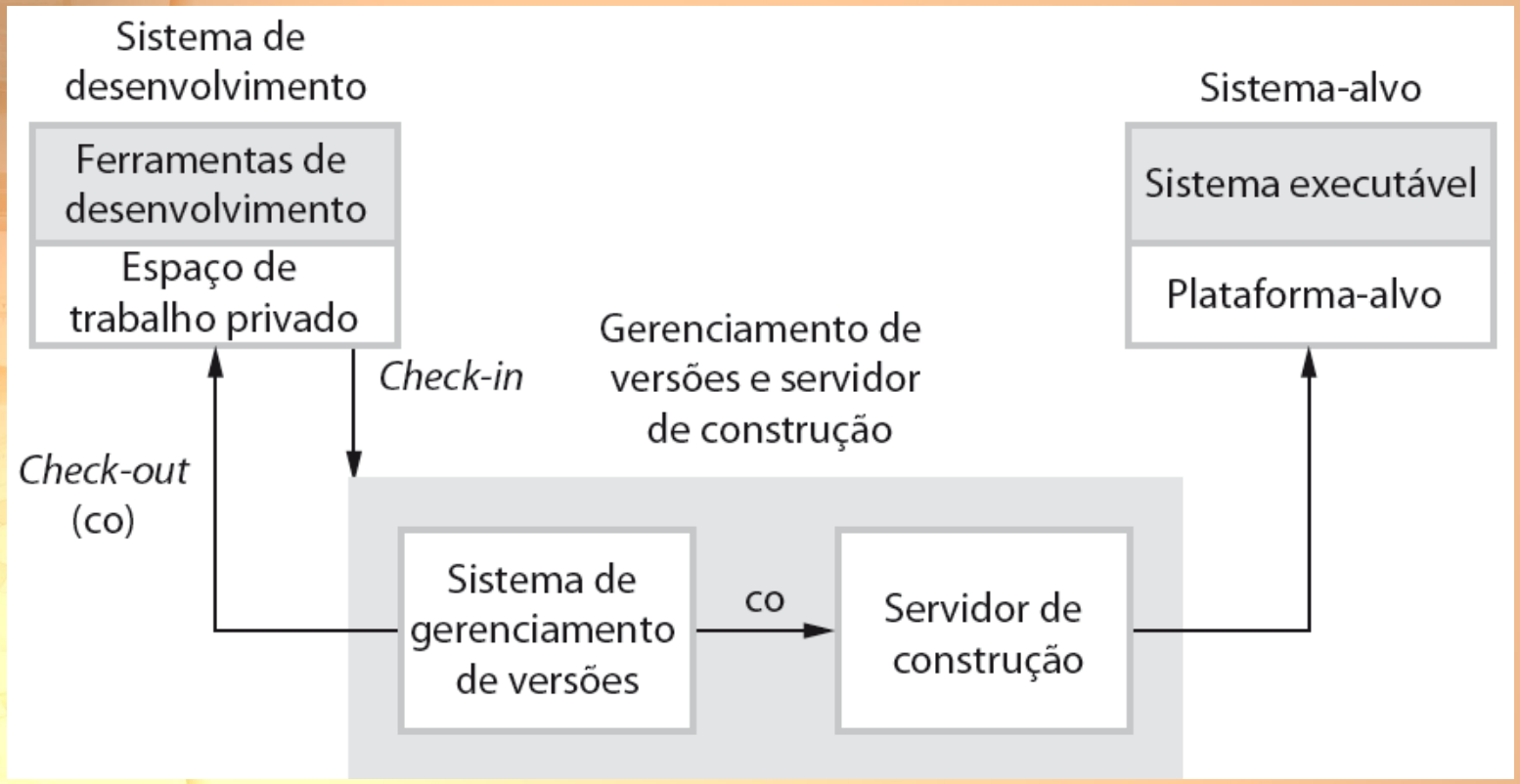
- Ferramentas de construção de sistema e ferramentas de gerenciamento de versões devem se comunicar com o processo de construção já que envolve a realização de *check-out* de versões de componentes do repositório gerenciado pelo sistema de gerenciamento de versões.

- A descrição de configuração usada para identificar uma *baseline* também é usada pela ferramenta de construção do sistemas.

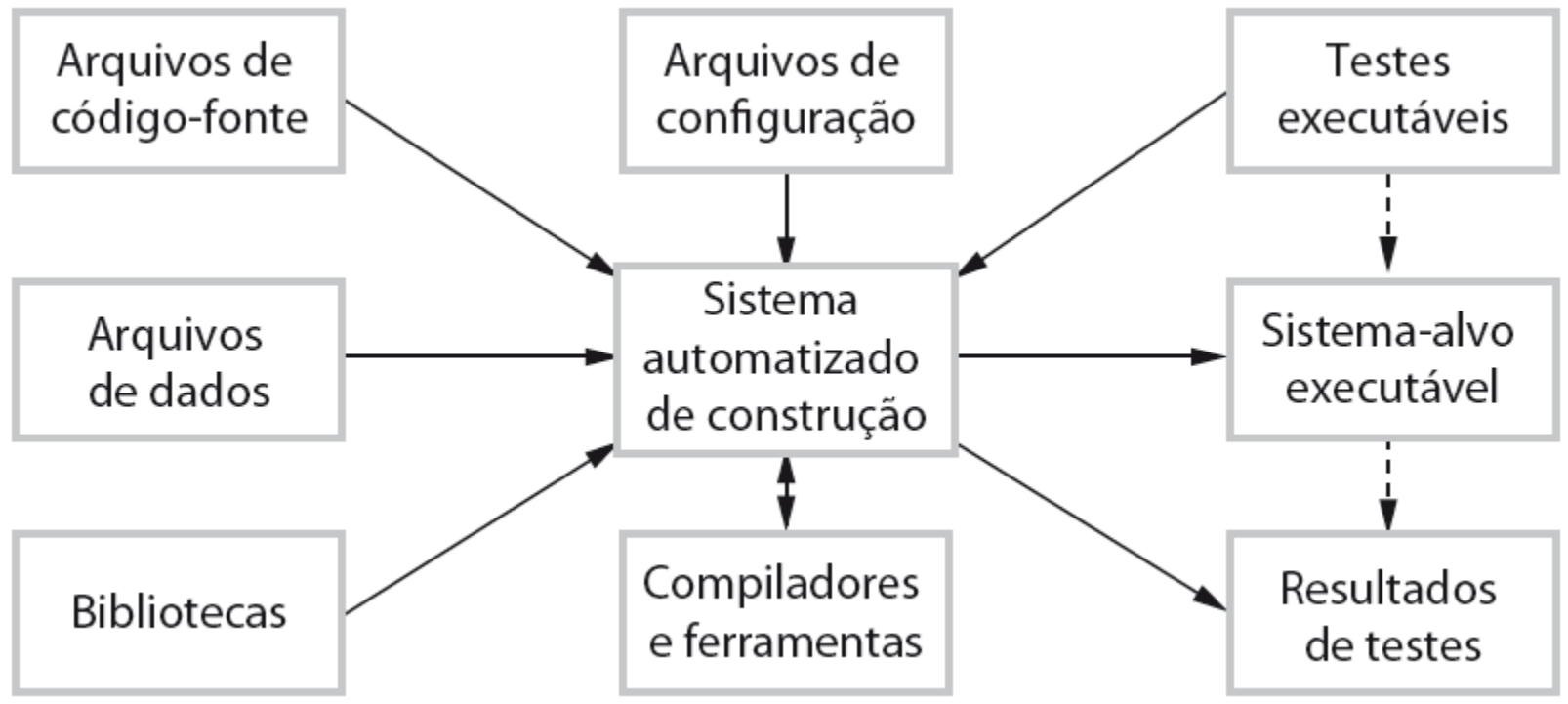
Plataformas de construção

- O sistema de desenvolvimento, que inclui ferramentas de desenvolvimento como compiladores, editores de código-fonte, etc.
  - ✓ Desenvolvedores realizam *check-out* de código do sistema de gerenciamento de versões em um espaço de trabalho privado antes de fazer mudanças ao sistema
- O servidor de construção, que é usado para construir versões definitivas e executáveis do sistema.
  - ✓ Desenvolvedores realizam check-in de código para o sistema de gerenciamento de versões antes de ser construído. A construção do sistema pode contar com bibliotecas externas que não estão incluídas no sistema de gerenciamento de versões.
- O ambiente-alvo, que é a plataforma sobre a qual o sistema é executado.

Plataformas de desenvolvimento, construção e alvo



Construção de sistemas



Construir a funcionalidade do sistema

engenharia de SOFTWARE

- Geração de script de construção
- Integração de sistema de gerenciamento de versões
- Recompilação mínima
- Criação de sistemas executáveis
- Automação de testes
- Emissão de relatórios
- Geração de documentação

Recompilação mínima

- As ferramentas de apoio à construção do sistema geralmente são projetadas para minimizar a quantidade de compilação necessária.
- O que é feito por meio da verificação da disponibilidade de uma versão compilada de um componente. Se assim for, não existe a necessidade de recompilar esse componente.
- Uma assinatura única identifica cada arquivo e cada versão do código-objeto que é alterado quando o código-fonte é editado.
- Ao comparar as assinaturas nos arquivos de código-fonte e código-objeto, é possível decidir se o código-fonte foi usado para gerar o código-objeto do componente.

Identificação de arquivos

- *Timestamps* de modificação
  - ✓ A assinatura no arquivo do código-fonte é a hora e a data em que o arquivo foi modificado. Se o arquivo do código-fonte de um componente foi modificado após o arquivo do código-objeto relacionado, em seguida, o sistema assume que é necessária a recompilação para criar um novo arquivo de código-objeto.

- *Checksums* de código-fonte
  - ✓ A assinatura no arquivo do código-fonte é uma soma calculada a partir dos dados no arquivo. A função soma calcula um número único usando o texto-fonte como input. Se você alterar o código fonte (mesmo que por um único caractere), vai gerar uma somatória diferente. Portanto, você pode ter certeza de que os arquivos de código fonte com diferentes somatórias são realmente diferentes.

Timestamps vs. *checksums*

- *Timestamps*
  - ✓ Como os arquivos de código-fonte e código-objeto são ligados por nome em vez de por meio de uma assinatura explícita de arquivo fonte, geralmente não é possível construir diferentes versões de um componente de código-fonte no mesmo diretório, ao mesmo tempo, já que essas iriam gerar arquivos-objeto com o mesmo nome .

- *Checksums*
  - ✓ Ao recompilar um componente, ele não substitui o código-objeto, como seria normalmente o caso quando o *timestamp* é usado. Em vez disso, ele gera um novo arquivo de código-objeto e *tags* com a assinatura do código-fonte. É possível fazer a compilação em paralelo, e diferentes versões de um componente podem ser compiladas ao mesmo tempo.

Integração contínua

- Realizar o *check-out* do *mainline* do sistema de gerenciamento de versões no espaço de trabalho privado do desenvolvedor.

- Construir o sistema e executar testes automatizados para garantir que o sistema construído passe em todos os testes. Se não, a construção será interrompida e você deverá informar quem fez o último check-in do sistema de *baseline*. Eles são os responsáveis pela reparação do problema.

- Fazer as mudanças para os componentes do sistema.

- Construir o sistema no espaço de trabalho privado e executar novamente os testes do sistema. Se os testes falharem, continuar editando.

Integração contínua

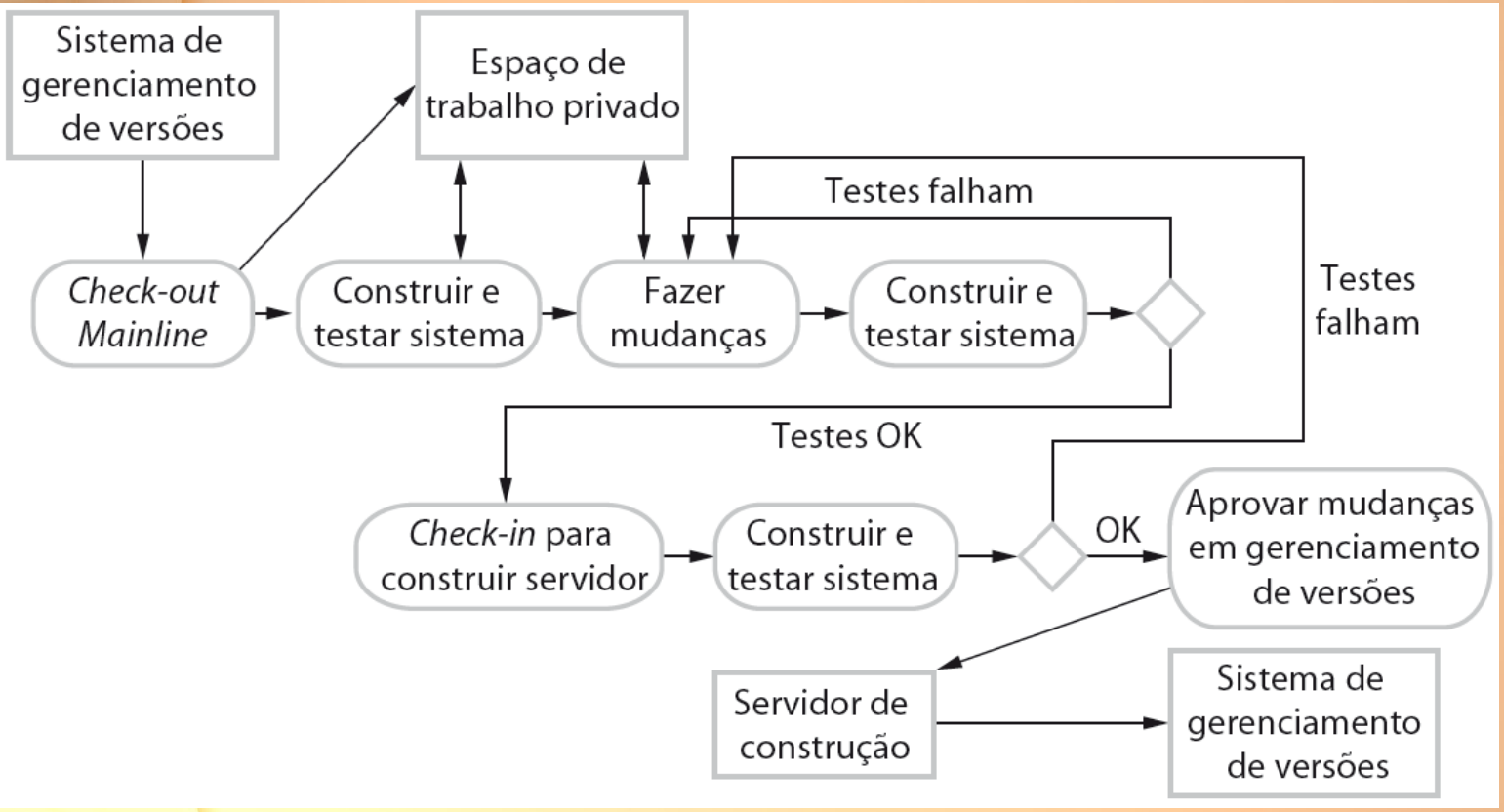
- Uma vez que o sistema passou nos testes, verificar no sistema construído, mas não aprovar como novo *baseline* de sistema.

- Construir o sistema no servidor de construção e executar os testes. Você precisa fazer isso no caso de outros componentes terem sido modificados depois de já ter acontecido o *check-out* do sistema.

- Se este for o caso, fazer o *check-out* dos componentes que falharam e editar esses testes passem em seu espaço de trabalho privado.

- Se o sistema passar nos testes sobre o sistema de construção, e em seguida, aprovar as mudanças feitas como uma nova *baseline* no *mainline* de sistema.

Integração contínua



Construção diária

- A organização responsável pelo desenvolvimento define um tempo de entrega para os componentes do sistema (por exemplo, 14 horas).

  - ✓ Caso os desenvolvedores tenham novas versões dos componentes que estão escrevendo, eles devem entregá-las nesse período.
  - ✓ Uma nova versão do sistema é construída a partir desses componentes, por meio da compilação e ligação desses para formar um sistema completo.
  - ✓ Em seguida esse sistema é entregue à equipe de testes, que realiza um conjunto predefinido de testes de sistema.
  - ✓ Defeitos que são descobertos durante os testes do sistema são documentados e voltam para os desenvolvedores do sistema. Eles reparam esses defeitos em uma versão posterior do componente.

Gerenciamento de *releases*

- Um release de um sistema é uma versão de um sistema de software distribuído aos clientes.
- Para um software de mercado, normalmente é possível identificar dois tipos de release: releases grandes que proporcionam nova funcionalidade importante, e releases menores, que reparam bugs e solucionam os problemas dos clientes.
- Para softwares customizados ou linhas de produto de software, os releases do sistema podem ter que ser produzidos para cada cliente e clientes individuais podem estar executando várias versões diferentes do sistema, ao mesmo tempo.

Acompanhamento de *releases*

- No caso de um problema, pode ser necessário reproduzir exatamente o software que foi entregue para um cliente particular.
- Quando é produzida uma versão do sistema, essa deve ser documentada para assegurar que possa ser recriada no futuro.
- Isso é particularmente importante para sistemas embutidos e customizados, de longa vida útil, tais como os que controlam máquinas complexas.
  - ✓ Os clientes podem usar um único release desses sistemas por muitos anos e podem exigir mudanças específicas para um sistema de software especial muito tempo depois da data do release original.

Documentação de *releases*

- Para um documento de release, você precisa gravar as versões específicas dos componentes do código-fonte que foram usados para criar o código executável.
- Você deve manter cópias dos arquivos de código-fonte, executáveis correspondentes e todos os dados e arquivos de configuração.
- Você também deve gravar as versões do sistema operacional, bibliotecas, compiladores e outras ferramentas usadas para construir o software.

Planejamento de *releases*

- Bem como o trabalho técnico envolvido na preparação e distribuição de um release, deve-se preparar material de publicidade e divulgação além de estratégias de marketing para convencer os clientes a comprarem o novo release do sistema.

- Calendário de release

  - ✓ Se os releases são muito frequentes ou exigem atualizações do hardware, os clientes podem não mudar para o novo release, especialmente se tiverem que pagar por isso.

  - ✓ Se os releases do sistema são muito pouco frequentes, pode se perder parte do mercado pois os clientes mudam para sistemas alternativos.

Componentes de *releases*

- Bem como o código executável do sistema, um release também pode incluir:

  - ✓ Os arquivos de configuração definem como o release deve ser configurado para instalações particulares;
  - ✓ Os arquivos de dados, tais como arquivos de mensagens de erro, são necessários para a operação do sistema ser bem sucedida;
  - ✓ Um programa de instalação que é usado para ajudar a instalar o sistema no hardware alvo;
  - ✓ Documentação eletrônica e em papel que descreve o sistema;
  - ✓ Empacotamento e publicidade associada que foram projetadas para esse release.

Fatores que influenciam o planejamento de *releases* do sistema

Fator	Descrição
Qualidade técnica do sistema	Caso sejam relatados defeitos graves de sistema, que afetem a maneira como muitos clientes o usam, pode ser necessário emitir um *release* de reparação de defeitos. Pequenos defeitos de sistema podem ser reparados mediante a emissão de *patches* (normalmente distribuídos pela Internet) que podem ser aplicados no *release* atual do sistema.
Mudanças de plataforma	Talvez você precise criar um novo *release* de uma aplicação de software quando uma nova versão da plataforma do sistema operacional for lançada.
Quinta lei de Lehman (ver Capítulo 9)	Essa 'lei' sugere que se você adicionar nova funcionalidade a um sistema, você também introduzirá *bugs* que limitarão a quantidade de funcionalidade que pode ser incluída no próximo *release*. Portanto, um *release* de sistema com funcionalidade nova e significativa pode ser seguido por um *release* que se concentra em reparar os problemas e melhorar o desempenho.

Fatores que influenciam o planejamento de *releases* do sistema

Fator	Descrição
Concorrência	Para software de mercado de massa, um novo *release* de sistema pode ser necessário porque um produto concorrente introduziu novos recursos e a fatia de mercado pode ser perdida caso estes não sejam fornecidos aos clientes existentes.
Requisitos de marketing	O departamento de marketing de uma organização pode ter feito um compromisso para *releases* estarem disponíveis em uma determinada data.
Propostas de mudança de cliente	Para sistemas customizados, os clientes podem ter feito e pago por um conjunto específico de propostas de mudanças de sistema e eles esperam um *release* de sistema assim que estas sejam implementadas.

Pontos importantes

- A construção do sistema é o processo de montagem dos componentes de sistema em um programa executável para executar em um sistema de computador-alvo.

- Os software devem ser frequentemente reconstruído e testados imediatamente após uma nova versão ser construída. O que facilita a detecção de bugs e problemas que tenham sido introduzidos desde a última construção.

- Os releases de sistema incluem o código executável, arquivos de dados, arquivos de configuração e documentação.

- O gerenciamento de releases envolve a tomada de decisões sobre as datas de release de sistema, a preparação todas as informações para a distribuição e a documentação de cada release de sistema.