Contexto para Gerência de Configuração

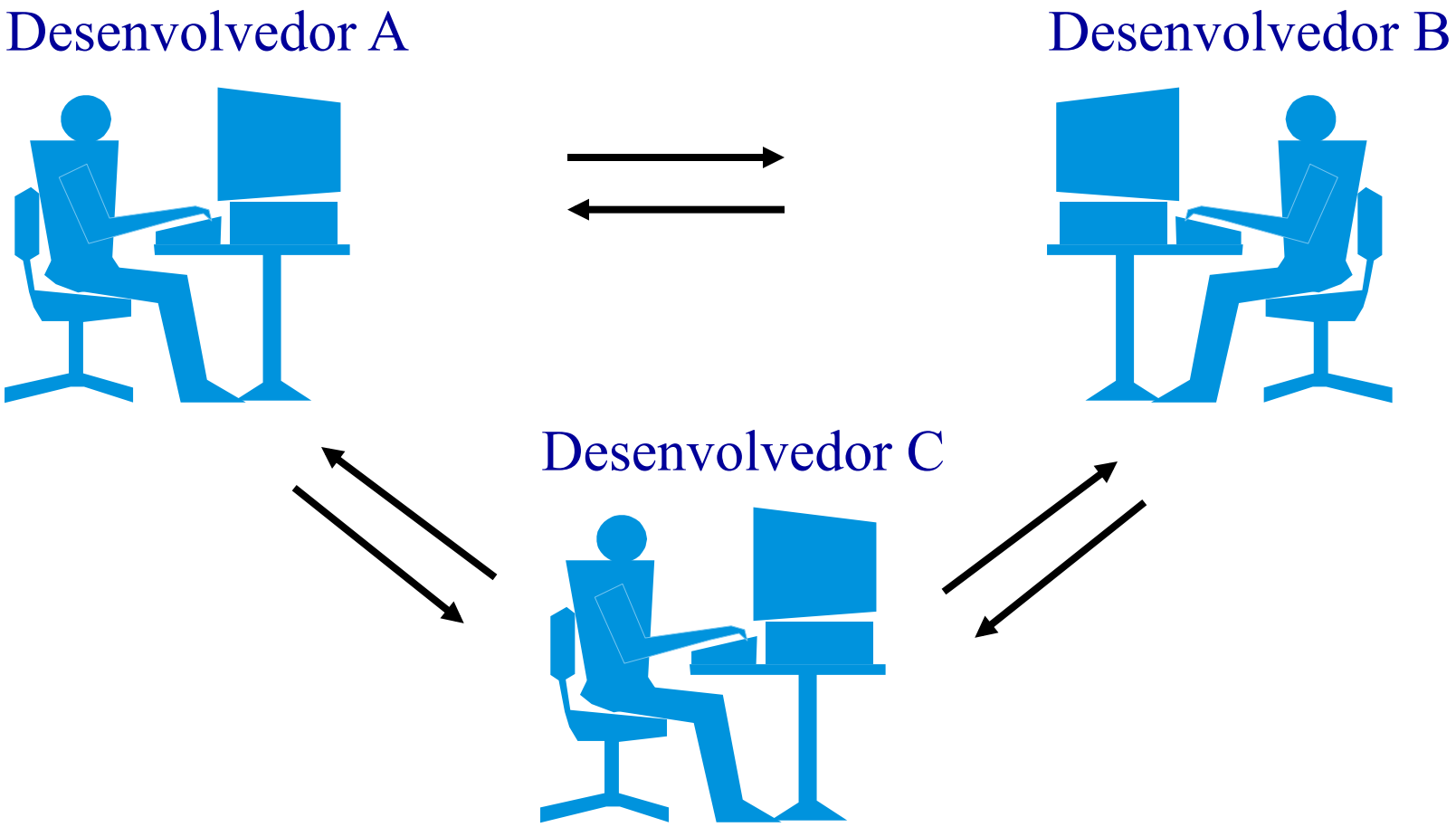
Guilherme Alexandre Monteiro Reinaldo
CIN/UFPE

Gerência de Configuração e Mudança

Objetivo

• Compreender a importância do uso de mecanismos de gerência de configuração e de mudança, seus métodos, processos e ferramentas.
• Fornecer os principais conceitos relacionados a GC.
• Criar uma visão geral de como GC pode ser aplicada a um projeto de software.

Problema da Quebra de Comunicação



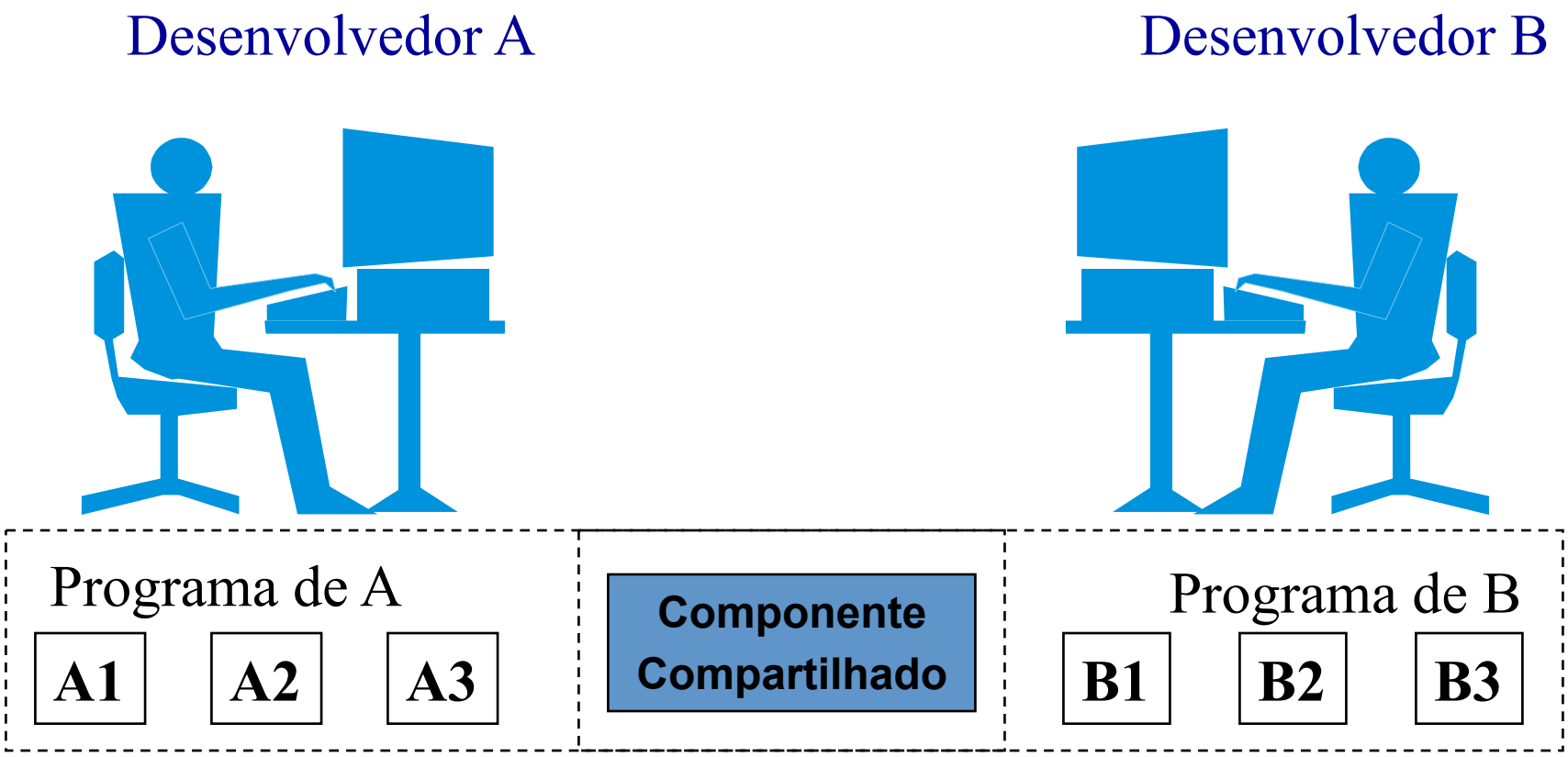
Problema da Quebra de Comunicação (continuação)

- Falhas de comunicação em equipes

- Ocorre pelas mais diversas razões:
  - Vocabulários incompatíveis
  - Culturas de desenvolvimento diferentes
  - Distância geográfica
  - Dificuldade de expressão

- Quando este problema acontece:
  - Os sistemas produzidos não atendem aos requisitos
  - Força de trabalho é desperdiçada

Problema dos Dados Compartilhados



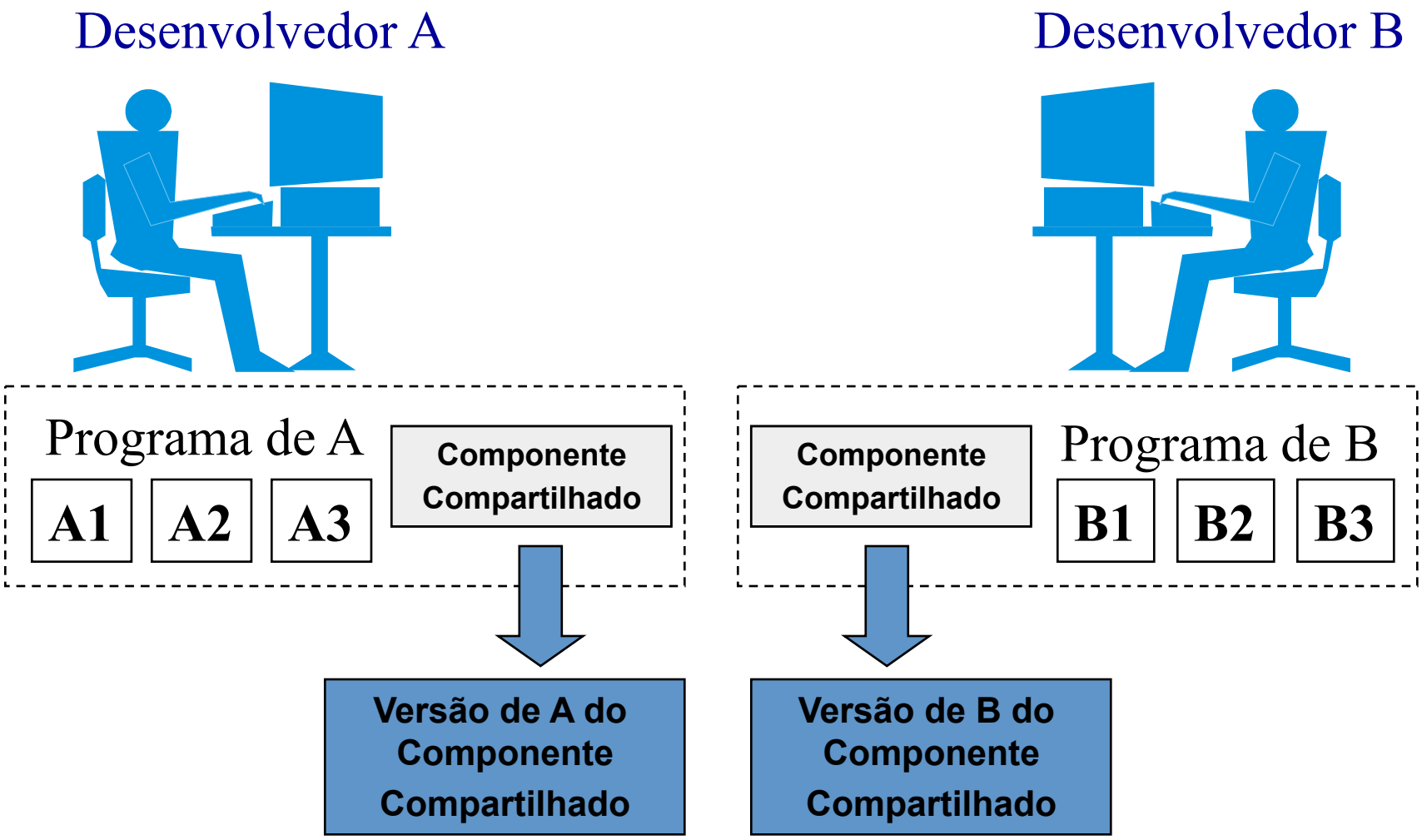
Problema dos Dados Compartilhados - Cenário

- O desenvolvedor A modifica o componente compartilhado
- Mais tarde, o desenvolvedor B realiza algumas alterações no mesmo
- 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 idéia sobre a causa do problema

Problema dos Dados Compartilhados - Solução simplista

- Solução simplista:
  - cada desenvolvedor trabalha em uma cópia "local" do componente
  - resolve o Problema dos Dados Compartilhados, mas cria um novo problema

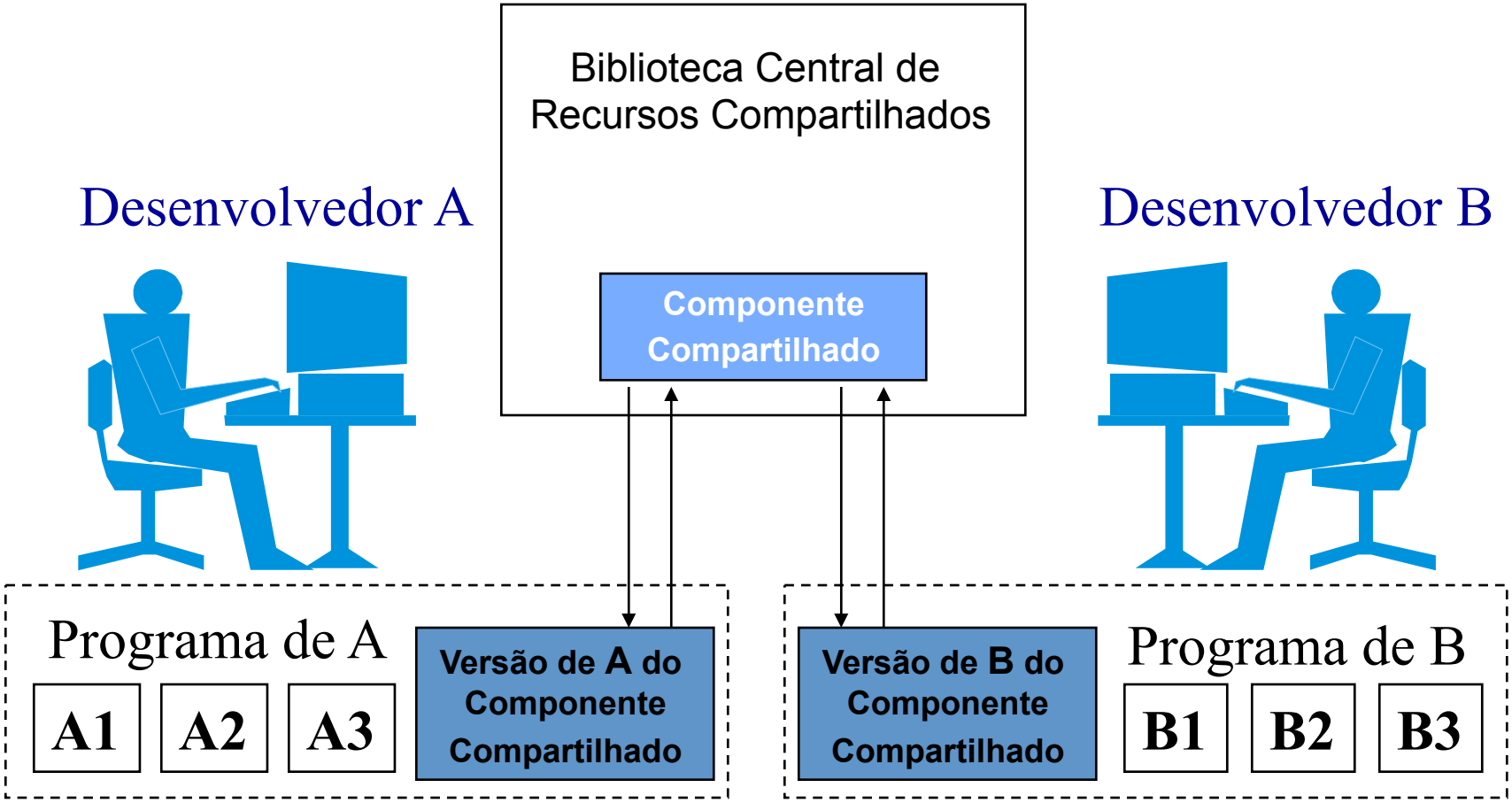
Problema da Manutenção Múltipla



Problema da Manutenção Múltipla (continuação)

- Ocorre quando cada desenvolvedor trabalha com uma cópia "local" do que seria o mesmo componente
- Dificuldade para saber:
  - Que funcionalidades foram implementadas em quais versões do componente
  - Que defeitos foram corrigidos
- Evitado através de uma biblioteca central de componentes compartilhados
  - Nesse esquema, cada componente é copiado para a biblioteca sempre que alterado
  - Resolve o Problema da Manutenção Múltipla, mas...

Problema da Atualização Simultânea



Problema da Atualização Simultânea – Cenário 1

- O desenvolvedor A encontra e corrige um defeito em sua versão do componente compartilhado
- Uma vez corrigido, o componente modificado é copiado para a biblioteca central
- O desenvolvedor B encontra e corrige o mesmo defeito em sua versão do componente por não saber que A já tinha feito isso
- O trabalho de A é desperdiçado

Problema da Atualização Simultânea – Cenário 2

- O desenvolvedor A encontra e corrige um defeito em sua versão do componente compartilhado
- Uma vez corrigido, o componente modificado é copiado para a biblioteca central
- O desenvolvedor B encontra e corrige um outro defeito em sua versão do componente, sem saber do defeito corrigido por A
- O desenvolvedor B copia sua versão do componente para a biblioteca central
- Além de o trabalho de A ser desperdiçado, a versão do componente que se encontra na biblioteca central continua apresentando um defeito
- O desenvolvedor A julga o problema como resolvido

Como Resolver?

- O problema da atualização simultânea não pode ser resolvido simplesmente copiando componentes compartilhados para uma biblioteca central

- Algum mecanismo de controle é necessário para gerenciar a entrada e saída dos componentes

O que é Gerência de Configuração?

- Gerência de configuração (GC) é o processo de identificar, organizar e controlar modificações ao software sendo construído

- A ideia é maximizar a produtividade minimizando os enganos

Objetivos de GC

- Definir o ambiente de desenvolvimento
- Definir políticas para controle de versões, garantindo a consistência dos artefatos produzidos
- Definir procedimentos para solicitações de mudanças
- Administrar o ambiente e auditar mudanças
- Facilitar a integração das partes do sistema

Benefícios

- Aumento de produtividade no desenvolvimento
- Menores Custos de Manutenção
- Redução de defeitos
- Maior rapidez na identificação e correção de problemas

Conceitos Básicos

Configuração

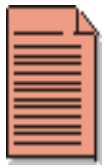
- Um projeto de desenvolvimento de software produz os seguintes itens:
  - Programas (código fonte, programas executáveis, bibliotecas de componentes, etc.)
  - Documentação (manuais do usuário, documento de requisitos, modelo de análise e projeto, etc.)
  - Dados (dados de teste e do projeto)
- Esses conjuntos de itens são chamados, coletivamente, de configuração do software

Item de Configuração

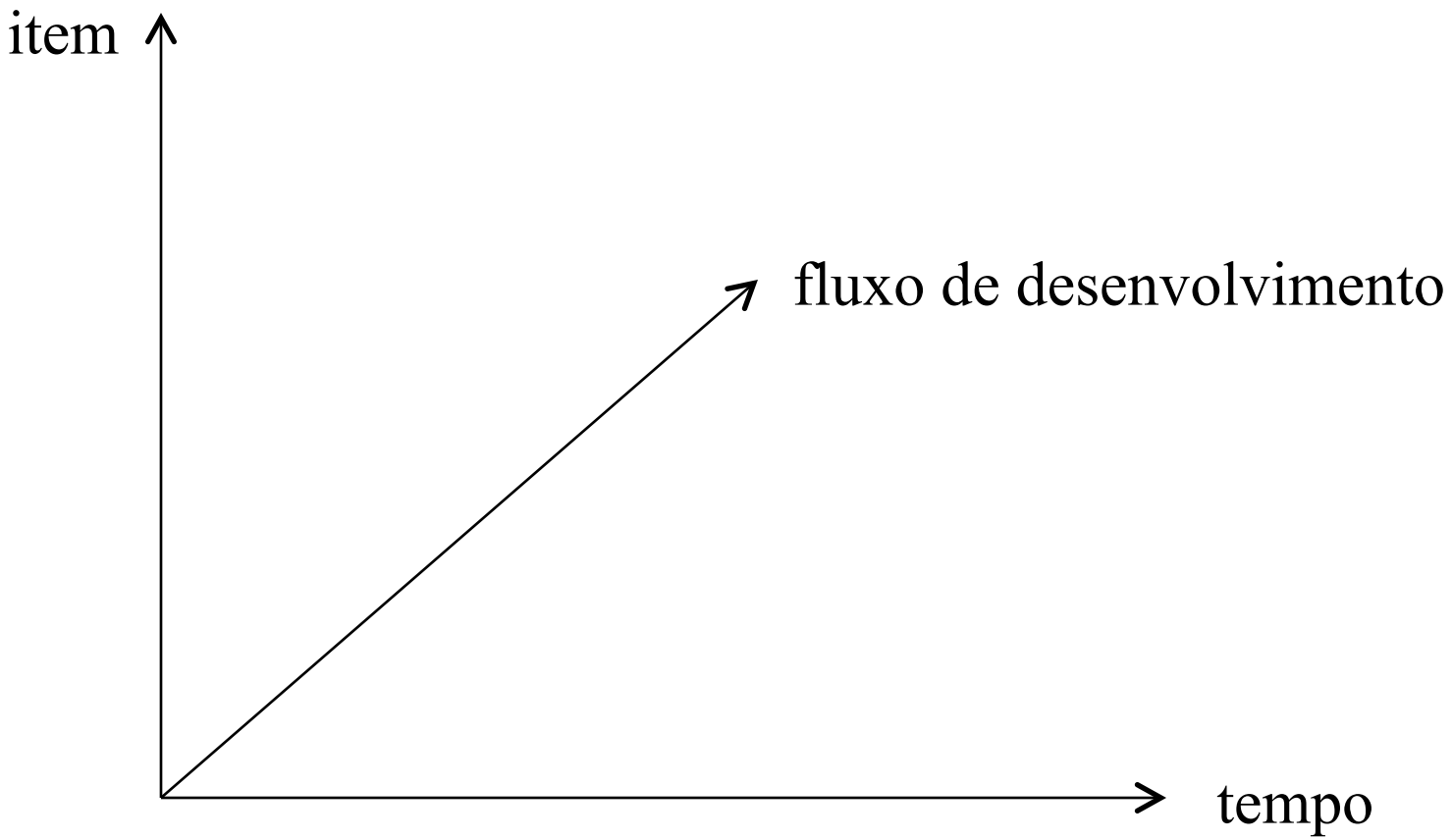
- Um conjunto de itens de hardware e/ou software vistos como uma entidade única para fins de gerência de configuração

- Um item de configuração está sujeito a mudanças e essas devem obedecer às políticas estabelecidas
- Normalmente, um item de configuração é estabelecido para cada pedaço de software que pode ser projetado, implementado e testado de forma independente

- Definição simplificada: um programa e sua documentação associada



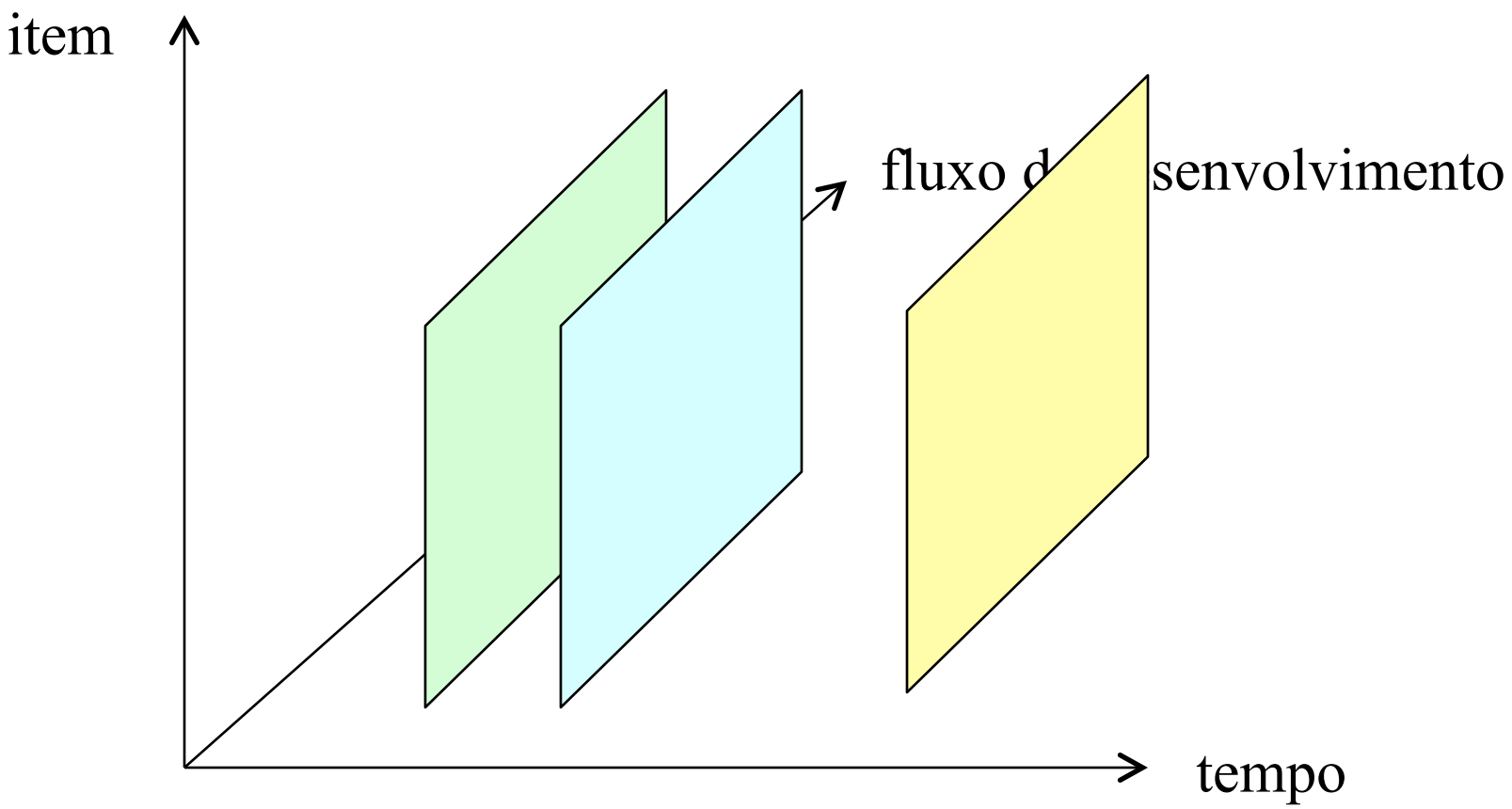
Configuração de Software



Baseline

- Uma especificação ou produto que foi formalmente revisado e aceito
  - Serve como base para os passos posteriores do desenvolvimento
- A configuração do software em um ponto discreto no tempo
- Só pode ser modificado através de procedimentos formais (solicitações de mudança)
- Um artefato ou conjunto de artefatos só se torna um item de configuração depois que um baseline é estabelecido

Baseline



Razões para Criar um Baseline

- **Reproducibilidade** – a habilidade de reproduzir uma versão anterior do sistema
- **Rastreabilidade** – Estabelece uma relação predecessor-sucessor entre artefatos do projeto (projeto satisfaz requisitos, código implementa projeto, etc.)
- **Geração de Relatórios** – A comparação dos conteúdos de dois *baselines* ajuda na depuração e criação de documentação
- **Controle de Mudanças** – referencial para comparações, discussões e negociações

Baselines importantes

- Baselines são considerados marcos no processo de desenvolvimento:
  - funcional: requisitos
  - de produto: releases, iterações

Baselines importantes

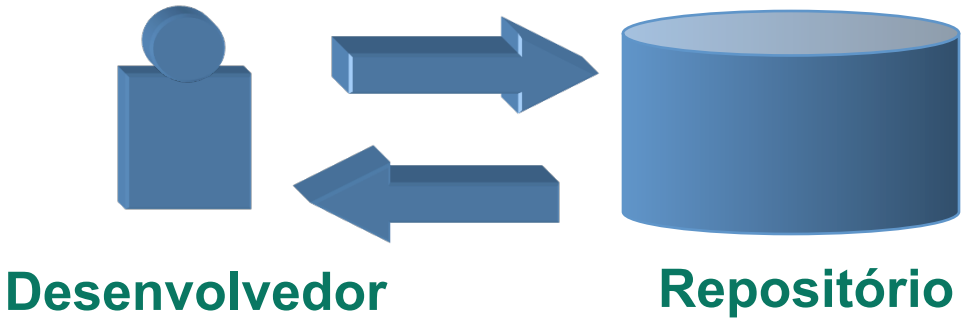
- Funcional
  - É o primeiro *baseline*. Consiste de um documento de requisitos (DR) aprovado, contendo todos os requisitos, funcionais e não-funcionais, associados a um determinado item de configuração. O processo formal de gerência de mudanças é iniciado quando este *baseline* é estabelecido

Baselines importantes

- De produto
  - O conjunto de todos os artefatos produzidos pelo processo de desenvolvimento (código fonte, código objeto, documentação, dados, arquivos de configuração, etc).

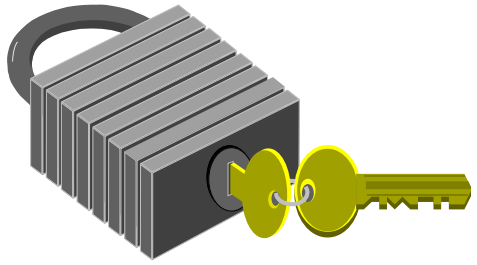
Repositório

- Local (físico e lógico) onde os itens de um sistema são guardados
- Pode conter diversas versões do sistema
- Utiliza mecanismos de controle de acesso

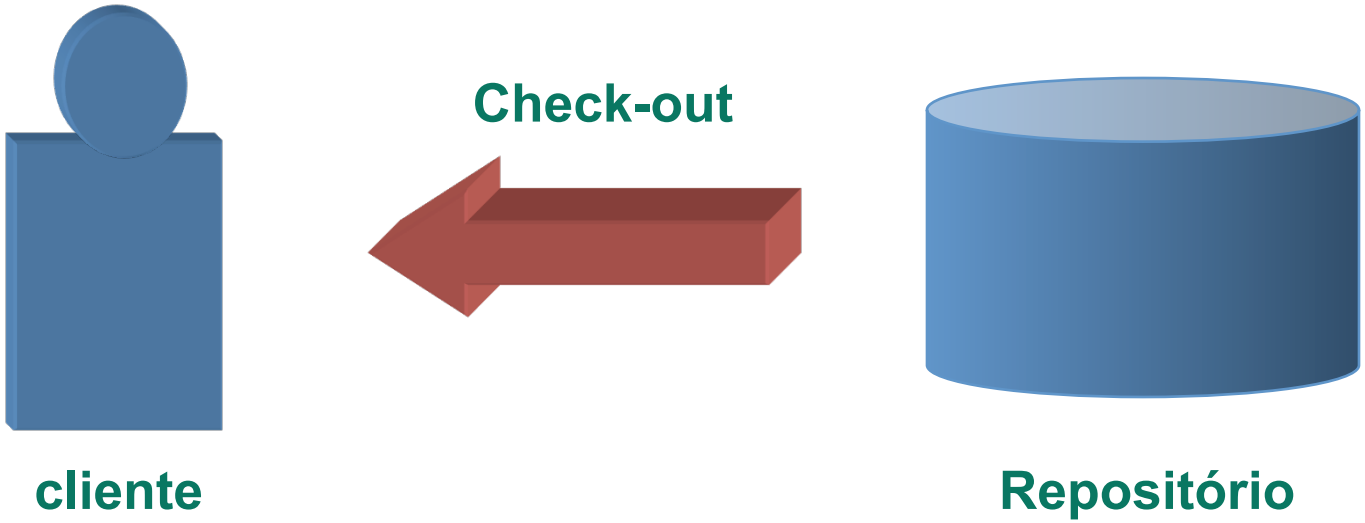


Lock

- Resolve a Atualização Simultânea
- Garante que apenas o usuário que detém o lock pode alterar o arquivo
- Problema: “serializa” o trabalho dos desenvolvedores



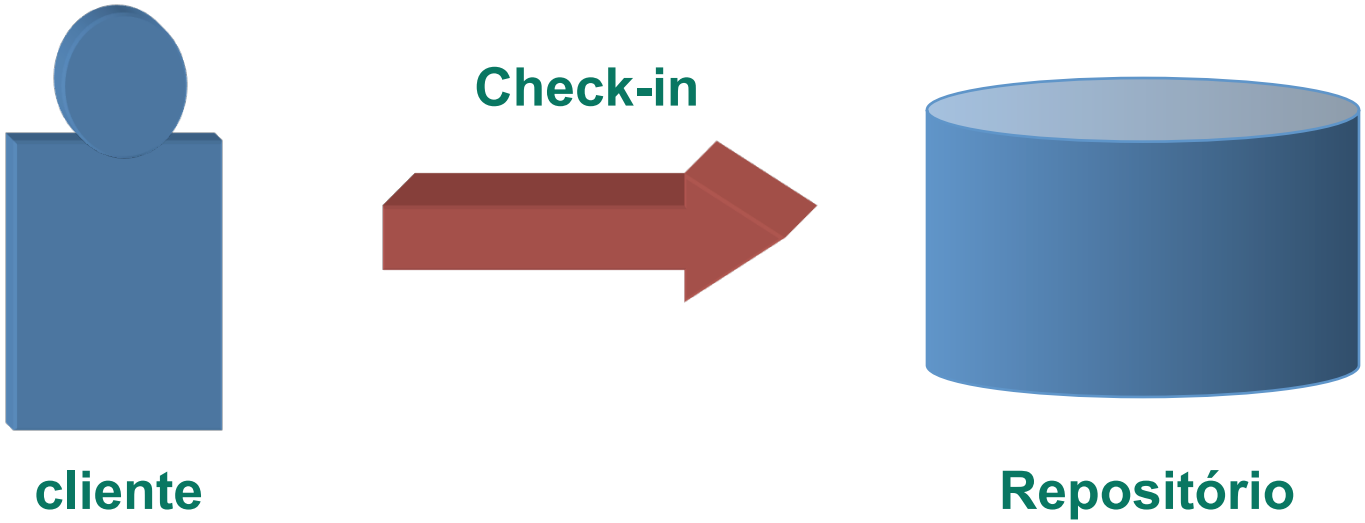
Check-Out



Check-Out (continuação)

- Recupera a (última) versão de um item de configuração guardada no repositório
  - Escrita
    - Verifica que ninguém detém o lock do item de configuração
    - Obtém o lock do item
    - Cria uma cópia, para edição, no cliente
  - Leitura
    - Verifica que alguém já detém o lock
    - Cria uma cópia, apenas para leitura, no cliente

Check-In



Check-In (continuação)

- Ação de inserir/atualizar um item de configuração no repositório
  - Verifica o lock do item de configuração, caso o mesmo já exista
  - Verifica e incrementa a versão do item
  - Registra informações das mudanças (autor, data, hora, comentários)
  - Inclui/atualiza o item

Build

- Representa uma versão ainda incompleta do sistema em desenvolvimento, mas com certa estabilidade
- Costuma apresentar limitações conhecidas
- Espaço para integração de funcionalidades
- Inclue não só código fonte, mas documentação, arquivos de configuração, base de dados, etc.
- A política de geração dos builds deve ser bem definida na estruturação do ambiente

Os Problemas na Geração de Builds

- Fazer os builds do sistema manualmente é muito demorado
- Pode ser difícil saber qual a versão “correta” de um arquivo
- Os pedaços do sistema podem estar em diversos locais diferentes
  - Alguns arquivos podem ser esquecidos

Os Problemas na Geração de Builds

- A integração das partes de um sistema em desenvolvimento normalmente é:
  - Realizada poucas vezes, apenas perto de sua implantação
  - Feita em frequência inversamente proporcional à complexidade do sistema
- Integrar as partes de um sistema é uma tarefa trabalhosa e sujeita a erros
  - Quanto maior o sistema, mais difícil

Os Problemas na Geração de Builds

- Consequência: problemas de integração tornam-se difíceis de detectar cedo no desenvolvimento
  - Costumam ser encontrados muito depois de sua introdução
  - É muito difícil rastrear suas causas

Geração de Buils através da Integração Contínua

- Geração frequente (pelo menos diária) de builds do sistema
  - As partes do sistema são integradas constantemente
  - Problemas de integração passam a ser encontrados logo que introduzidos, na maioria dos casos
- Considerada uma das “melhores práticas” no desenvolvimento de software
- A geração de builds deve ser automatizada e realizada com frequência adequada

Release

- Identificação e empacotamento de artefatos entregues ao cliente (interno ou externo) ou ao mercado
- Um release implica no estabelecimento de um novo baseline, de produto
- Produto de software supostamente sem erros
  - Versão do sistema validada após os diversos tipos de teste
  - Garantia de que todos os itens de configuração foram devidamente testados, avalidos, aceitos e estão disponíveis no novo *baseline*
- Processo iterativo/incremental produz, em geral, mais de um release

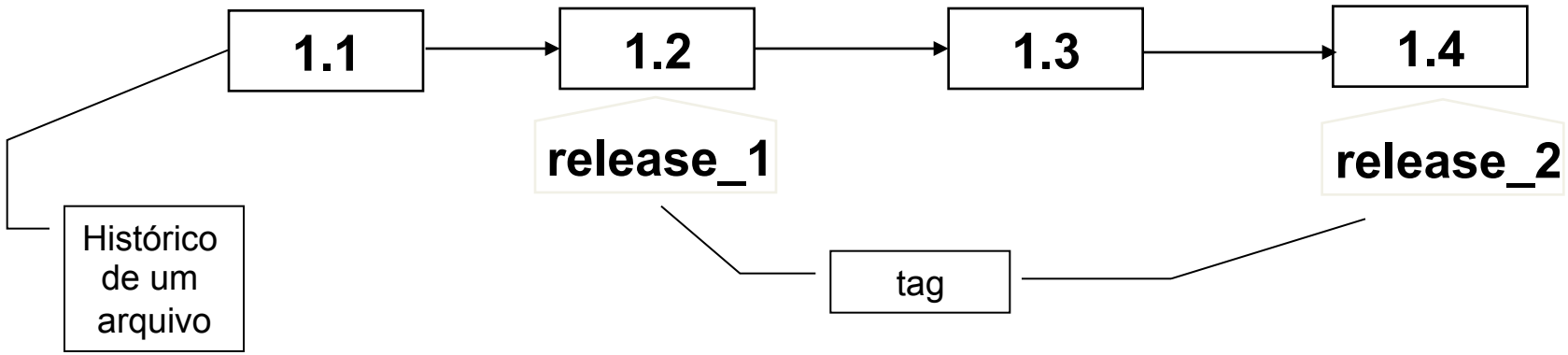
Tipos de release

- Normalmente, releases estão associados aos *milestones* do plano de projeto
- Internos
  - Controle de qualidade, acompanhamento de projeto, controle de riscos, aceitação, aquisição de conhecimento através da coleta de feedbacks, desenho da estratégia de implantação
- Externos
  - Implantado e utilizado pelo cliente

Tags

- Rótulos que são associados a conjuntos de arquivos
- Um tag referencia um ou mais arquivos em um ou mais diretórios
  - Costuma-se usar tags para:
    - Denominar projeto rotulando todos os arquivos associados ao projeto
    - Denominar uma versão do projeto (um build ou release) rotulando todos os arquivos associados ao build ou release

Tags – Cenário



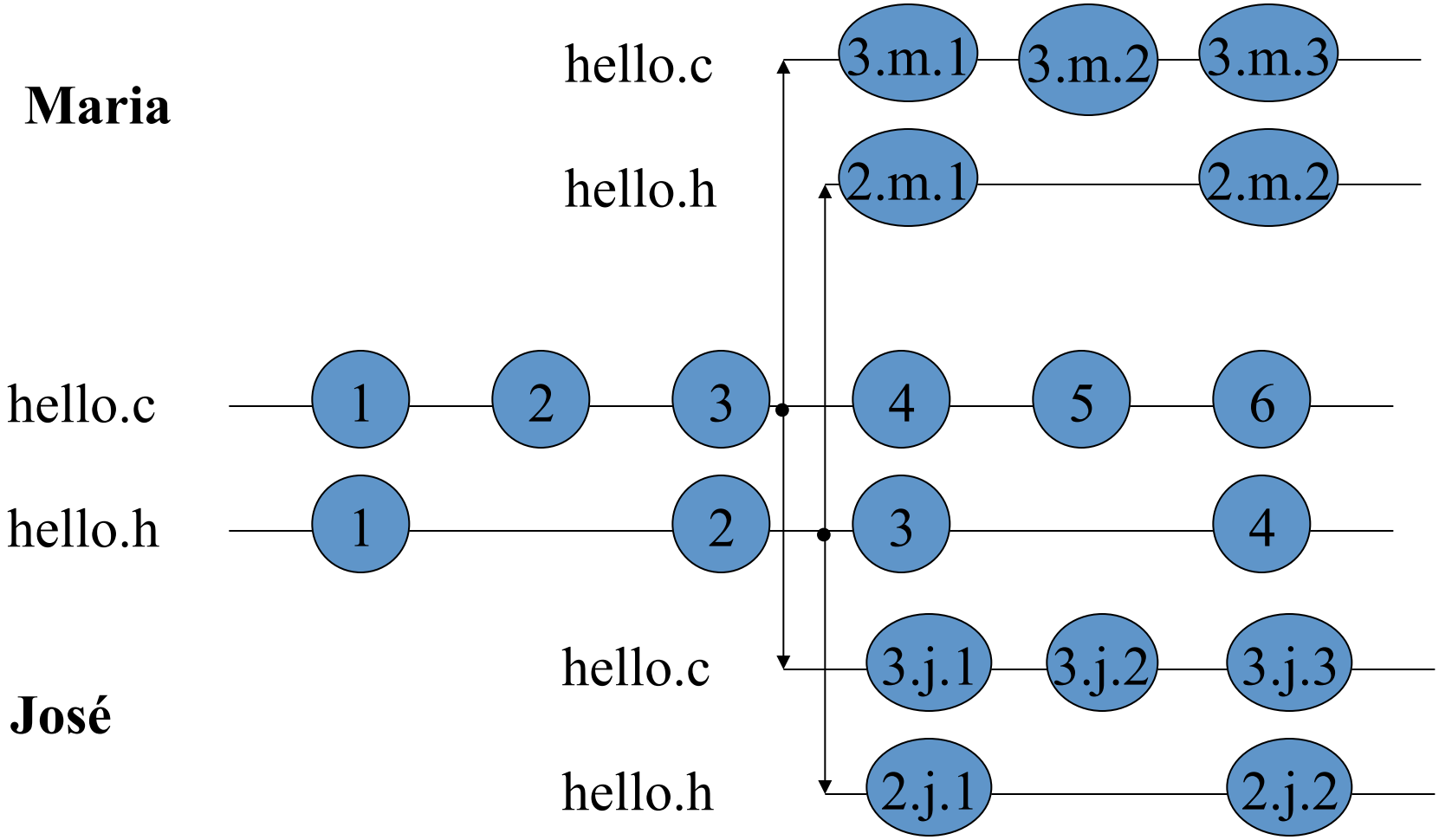
Branch

- Criação de um fluxo alternativo para atualização de versões de itens de configuração
- Recurso muito poderoso
- Devem existir regras bem definidas para criação de branches
  - Por que e quando devem ser criados?
  - Quais os passos?
  - Quando retornar ao fluxo principal?

Branch (continuação)

- Uso de lock inviabiliza a criação de branches
- Branches normalmente se originam de correções em versões anteriores

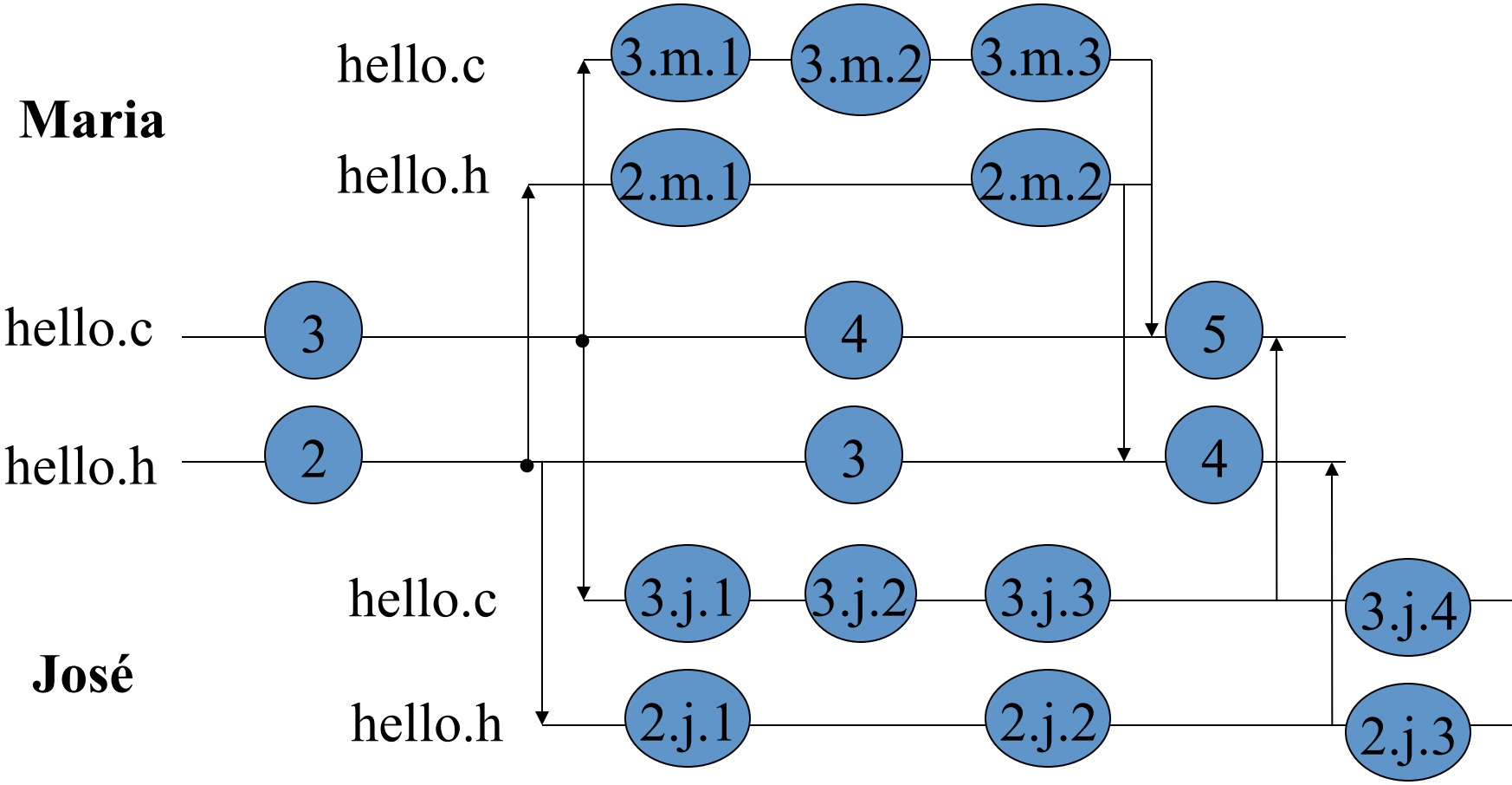
Branch (exemplo)



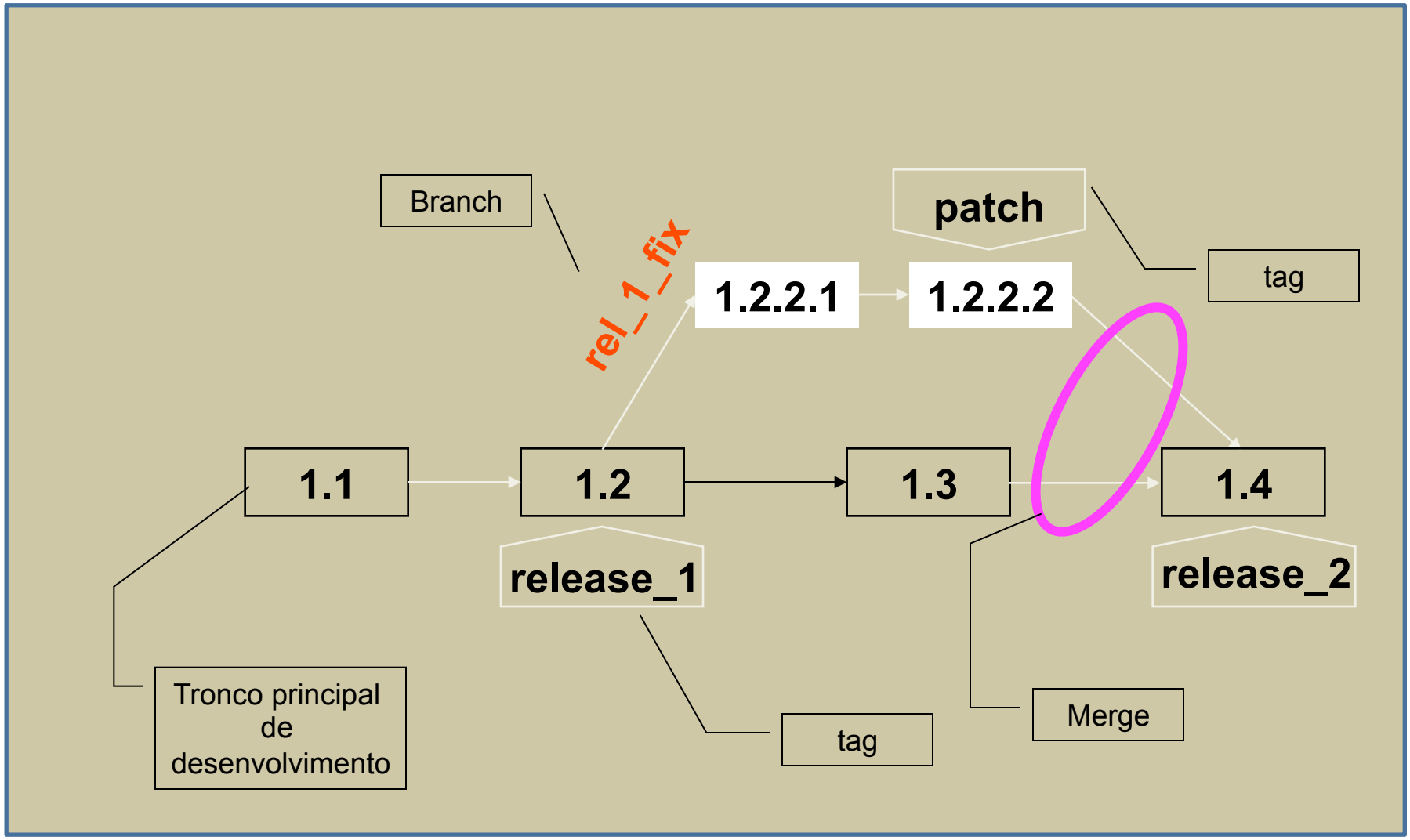
Merge

- Unificação de diferentes versões de um mesmo item de configuração
- Integração dos itens de configuração de um branch com os itens de configuração do fluxo principal
- Check-out atualizando a área local
- Algumas ferramentas fornecem um mecanismo automático para realização de merges
  - Mesmo com o uso de ferramentas, em vários casos há necessidade de intervenção humana

Merge (exemplo)



Branching e Merging: esquema gráfico



Oportunidades criadas com Gerenciamento de Configurações

- Reuso de itens de software
  - Artefatos
  - Componentes
- Automação de processo
  - Construção de *builds*
  - Geração de releases
  - Testes
  - Integração
- Aumento da produtividade das equipes
- Redução de re-trabalho
- Melhoria do acompanhamento do projeto

Controle de Mudanças

Contexto

- Desenvolvimento iterativo/incremental
- Novos conjuntos de requisitos, detalhados a cada iteração
- Mudanças em estratégias de negócio motivadas pelas mais diversas fontes: mercado, cultura, leis, etc

Problemas

- Controle do escopo do projeto
  - Modificações podem ampliar o leque de funcionalidades e aumentar significativamente o custo do projeto
  - Atrasos em entregas planejadas

- Controle de consistência dos artefatos
  - Uma mudança aparentemente localizada pode causar muito mais impacto do que o previsto
  - Degradação da qualidade do software (ex: abandono dos testes automatizados devido à inconsistência dos dados de teste)
  - Retrabalho

O que é Gerência de Mudanças?

- Gerência de Mudanças é o processo de avaliar, coordenar e decidir sobre a realização de mudanças propostas a itens de configuração (ICs)

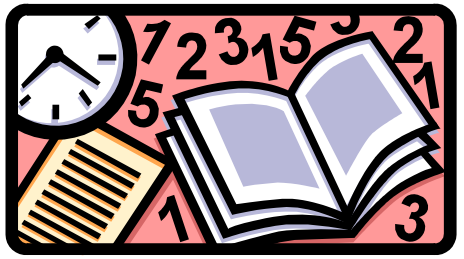
- Mudanças aprovadas são implementadas nos itens de configuração e nos dados e documentos relacionados

Objetivos da Gerência de Mudanças

- Garantir que os artefatos do sistema alcancem e mantenham uma estrutura definida através do seu ciclo de vida

- Definir procedimentos e documentação necessários para realizar modificações a ICs

- Prover os mecanismos necessários para conduzir mudanças de uma maneira controlada

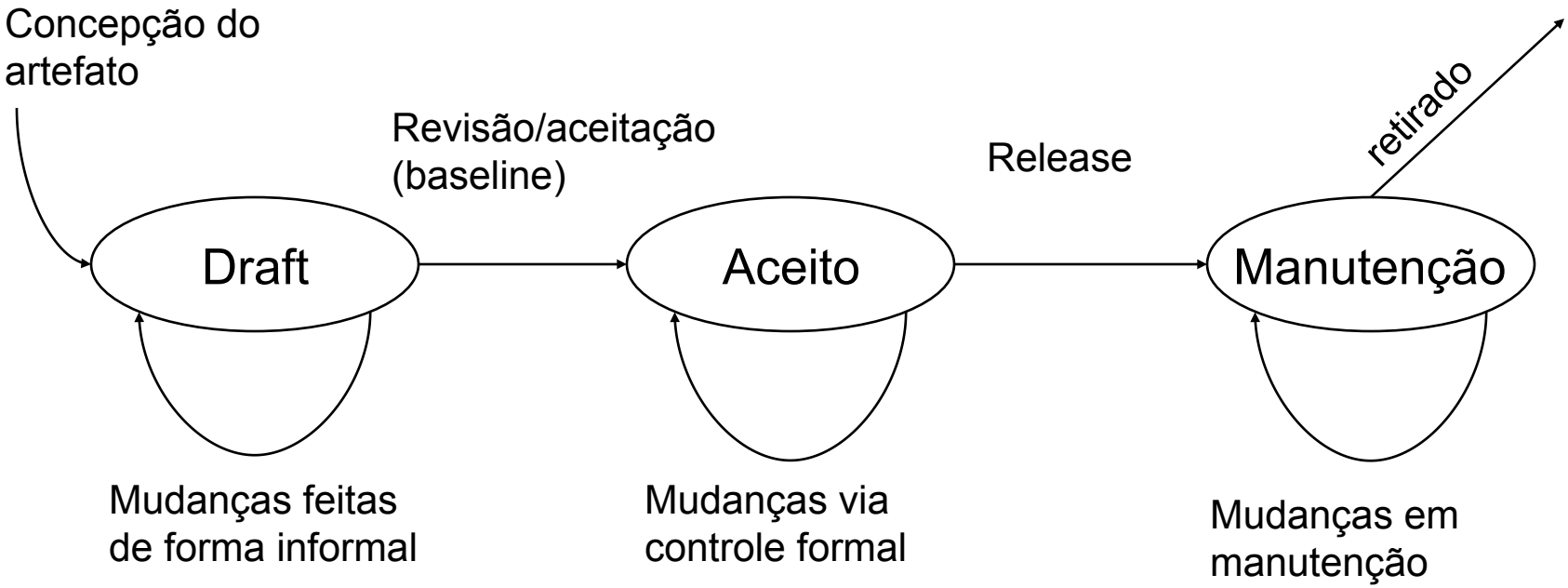


Benefícios

- Controle sobre o escopo do projeto
- Mais produtividade
  - cada solicitação será tratada de forma coordenada
  - Redução dos problemas de comunicação entre membros da equipe
- Mais qualidade, uma vez que cada mudança, antes de ser realizada, tem seu impacto avaliado
- Geração de dados para o acompanhamento (*tracking*) do projeto

Ciclo de vida de um artefato

Ciclo de vida de um artefato



Artefato Draft

- Mudanças frequentes e rápidas
- Demanda por agilidade
- Controle formal dificulta a criação do artefato
- Artefatos apenas gerenciados e controlados
  - Uso de controle de versão (CVS, Clear Case, entre outras ferramentas)

Artefato Aceito

- Artefato seguiu um processo de revisão, testes (se aplicável) e aceitação
- Inserido dentro do processo de controle de mudanças, tornando-se de fato **item de configuração**
- Mudanças via **solicitação formal**
- Presença do grupo gestor de mudanças (**CCB**) para avaliar e priorizar mudanças

Artefato em Manutenção

- Após a entrega de uma versão do produto, os artefatos passam para a fase de manutenção
- Controle de mudanças permanece formal para os artefatos de um *baseline*
- Novos artefatos podem ser desenvolvidos usando o mesmo modelo de ciclo de vida
- Sistema pode ser descontinuado ou removido do ambiente de produção

Processo de Gerência de Mudanças

Motivação

- Mudança é inevitável
- Mudar é fácil – controlar diversas mudanças simultâneas é difícil
- A gerência de mudanças introduz controle sobre as mudanças de maior relevância
  - Todas as mudanças são analisadas
  - Apenas as aprovadas são realizadas
  - Sempre se sabe quem modificou o que, onde e quando

Responsabilidades do CCB (grupo gestor de mudanças)

- Analisar as solicitações de mudança
- Controlar o escopo do projeto
- Reuniões com frequência adequada ao ritmo das solicitações de mudança
- Envolver *stakeholders* no processo de priorização, no processo de decisão
- Balanço entre o nível de controle desejado e *overhead* suportado
  - Questões menores devem ser resolvidas pelo líder do projeto junto à equipe, reduzindo o *overhead* do CCB

Características do CCB

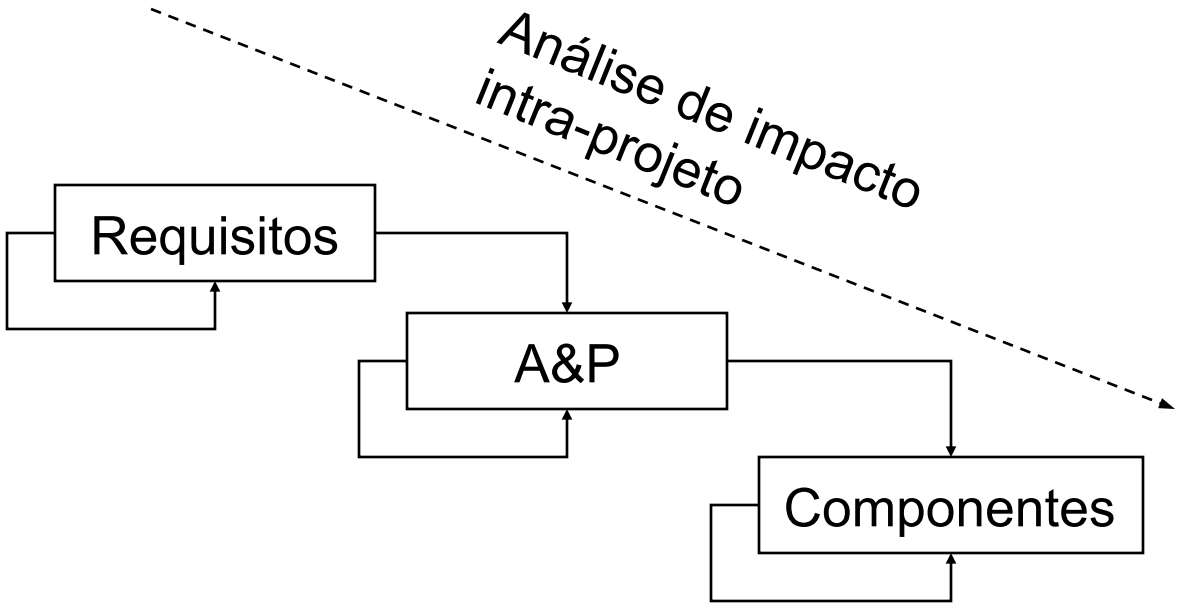
- Composição multidisciplinar
  - gerente, cliente, arquiteto
- Profissionais com grande capacidade de comunicação e negociação
- Pode apresentar uma estrutura hierárquica dependendo do tamanho e da quantidade de *stakeholders* e sistemas envolvidos (integrações)

Análise de impacto

- Mudanças de grande impacto devem ser comunicadas aos stakeholders envolvidos
- Análises de custo x benefício produzidas pelos stakeholders
- Priorização de mudanças
- Mudança pode ser rejeitada se o CCB perceber que o custo será mais caro que o benefício percebido
- Por questões de eficiência, algumas solicitações de mudança podem ser agrupadas por tema, subsistema ou área de negócio

Importância da análise de impacto

- Dentro do projeto
- Análises inter-sistemas também devem ser consideradas
  - Exemplo: frameworks, componentes ou bancos de dados compartilhados



Sobre o Processo de Gerência de Mudanças

- Deve ser definido um documento padrão para que mudanças possam ser solicitadas
- Esse documento normalmente se chama Solicitação de Mudança (SM, Em inglês CR)
- A um conjunto de pessoas (CCB), deve ser dada a autoridade para decidir se uma mudança será ou não implementada
- O processo é necessário para garantir que apenas mudanças avaliadas e aprovadas são realizadas em Itens de configuração (ICs)

Solicitações de Mudança

- Algumas informações que podem estar incluídas em uma SM:
  - Identificação única
  - Solicitante
  - Sistema/Projeto
  - Item a ser modificado
  - Classificação (melhoria, correção de defeito, outra)
  - Prioridade
  - Descrição
  - Situação (nova, atribuída, finalizada, verificada, fechada)

Estrutura de um registro de solicitação de mudança

1. IDENTIFICADOR DA SOLICITAÇÃO

<Um código (normalmente numérico) que identifica unicamente a solicitação de mudança.>

2. IDENTIFICAÇÃO DO SOLICITANTE

<O nome do indivíduo que solicitou a mudança, possivelmente incluindo informação adicional como posição, matrícula, etc.>

3. SISTEMA DESENVOLVIDO

3.1. NOME DO SISTEMA

<O nome do sistema no qual está sendo solicitada a mudança.>

3.2. NOME DO MÓDULO

<O nome do módulo no qual a mudança está sendo solicitada.>

3.3. NOME DA FUNCIONALIDADE

<O nome da funcionalidade na qual a mudança será efetuada.>

Estrutura de um registro de solicitação de mudança

4. CLASSIFICAÇÃO

<O tipo de mudança que está sendo solicitada. Normalmente três tipos de mudança são realizados: adição de nova funcionalidade, melhoria de funcionalidade já existente e correção de defeitos. Também é comum que a classificação seja feita com relação à natureza da mudança. Por exemplo: mudança de requisitos, de projeto, de implementação, etc.>

5. DESCRIÇÃO

<Uma descrição da mudança que está sendo solicitada. A descrição deve ser o mais não-ambígua e objetiva possível. Ao mesmo tempo, deve incluir toda informação necessária para implantar a mudança.>

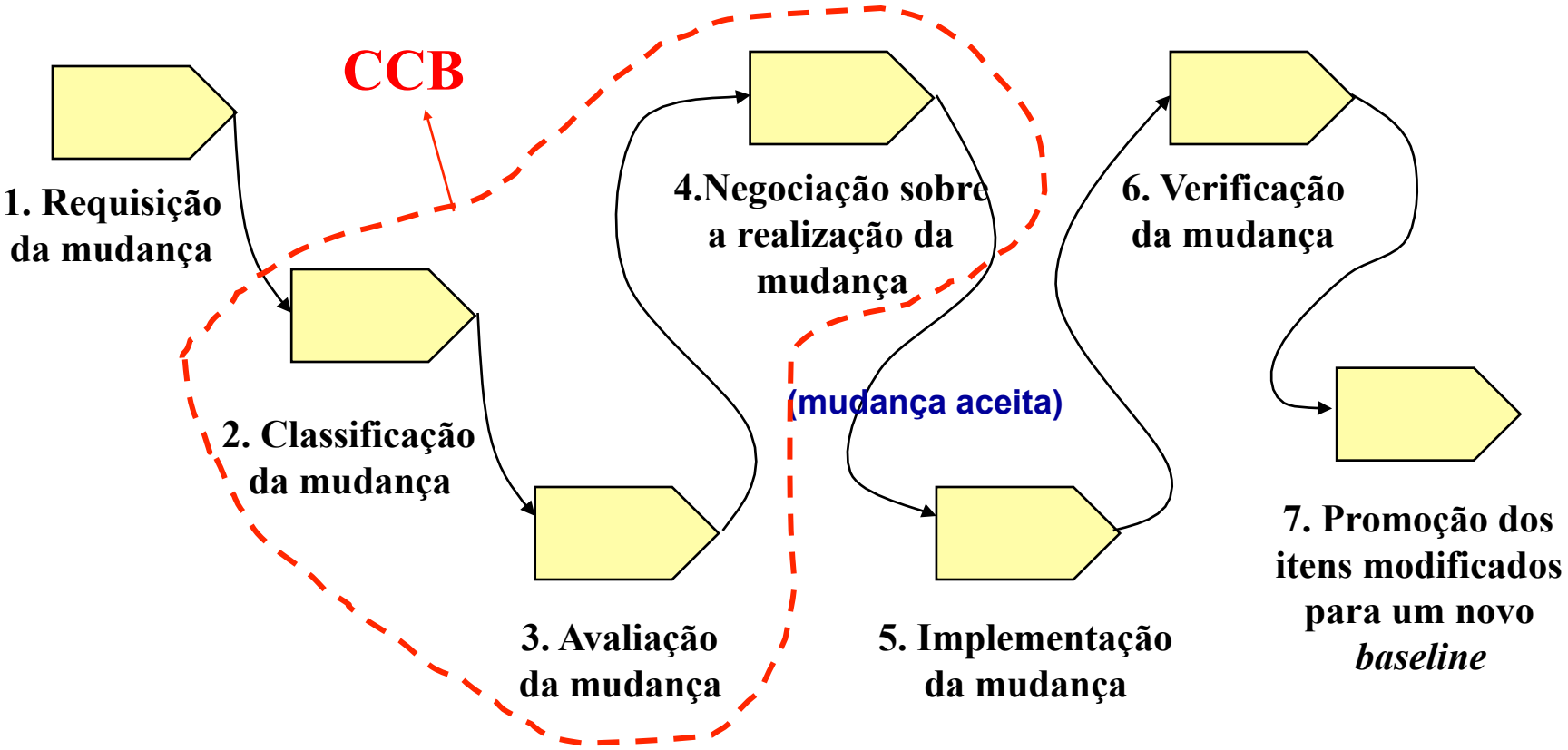
6. STATUS

<A situação atual da mudança. Por exemplo: aprovada, rejeitada, em implantação, postergada, etc. Essa informação deve ser mantida sempre atualizada.>

7. OBSERVAÇÕES GERAIS

<Informações adicionais sobre a solicitação de mudança. Por exemplo: se o solicitante já souber de módulos que serão afetados pela implantação da mudança, pode enumerá-los nesta seção.>

Etapas do Processo de Gerência de Mudanças Genérico



Correções Emergenciais

- Em algumas situações, não há tempo para seguir os procedimentos padrão para a realização de mudanças
- Defeitos não são normalmente processados pelo CCB, salvo se envolverem algum questionamento relativo ao escopo do projeto
- Mesmo nessas situações, porém, é muito importante que seja criada uma solicitação de mudança
- O objetivo é garantir um mínimo de ordem, mesmo em uma situação caótica

Correções Emergenciais

- Mudanças realizadas nessas circunstâncias podem comprometer a arquitetura ou inserir bugs
- Decisão:
  - Desfazer correção ou
  - Transformar a correção: refactoring, acréscimo de novos casos de teste

Exemplos de Status dos Defeitos

Estados Abertos		Próximos Estados
NEW	*Bug* inserido por alguém (automático)	Aceito ⇨ **ASSIGNED** Reatribuído ⇨ **NEW** Resolvido ⇨ **RESOLVED**
ASSIGNED	Atribuído à pessoa apropriada	Reatribuído ⇨ **NEW** Resolvido ⇨ **RESOLVED**
REOPENED	Reaberto: foi constatado que ainda não tinha sido resolvido	Aceito ⇨ **ASSIGNED** Reatribuído ⇨ **NEW** Resolvido ⇨ **RESOLVED**
UNCONFIRMED	Não confirmado que existe	Confirmado ⇨ **NEW** Resolvido ⇨ **RESOLVED**

Exemplos de Status dos Defeitos

Estados Fechados		Próximos Estados
RESOLVED	Foi resolvido (só está esperando a homologação)	Não foi resolvido ⇨ **REOPENED** Está ok ⇨ **VERIFIED** Está ok e pode ser fechado ⇨ **CLOSED**
VERIFIED	A correção foi homologada	Defeito é fechado ⇨ **CLOSED**
CLOSED	O *bug* é tido como resolvido	Não foi resolvido ⇨ **REOPENED**

Release notes

- Relação de solicitações de mudanças implementadas e testadas
- Pode ser parcialmente automatizado
- Comentários adicionais
  - Limitações atuais, problemas não resolvidos

Id	*Descrição*
1	Problema de performance na validação de pedido
2	Nova rotina de validação de crédito conforme normas de dezembro de 2002
…	…

Desafios

- Cultura organizacional
  - Agrupamento de solicitações em releases bem definidos e estabelecidos deve ser negociado com os *stakeholders* do projeto
  - Releases internos utilizados de forma efetiva como ferramenta de gestão de projeto
- Integração entre sistemas de controle de versão e mudanças

Ferramentas de Apoio à Gerência de Configuração

Ferramenta de Controle de Versões (CVS, por exemplo)

- Manter todos os arquivos em um repositório central
- Controlar o acesso a esse repositório, de modo a garantir a consistência dos artefatos

Ferramentas de Geração de Builds (Ant, por exemplo)

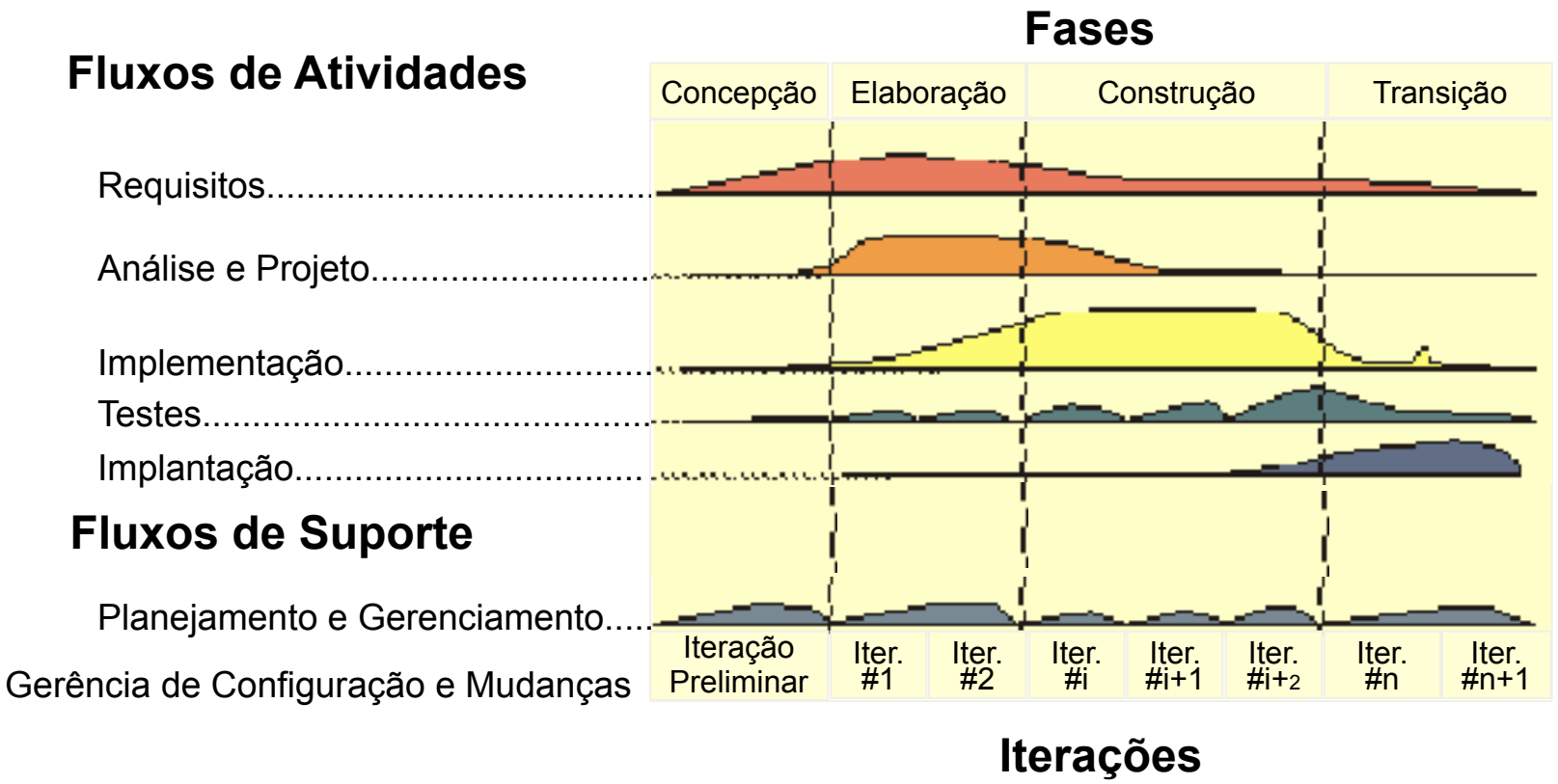
- Automatizar o processo de geração de *builds*

Ferramentas de Gestão de Solicitações de Mudanças (Bugzilla, por exemplo)

- Automatizar o processo de submissão e gestão de SMs

Gerência de Configuração no Desenvolvimento Iterativo - Relação com as Fases e Disciplinas de Desenvolvimento do RUP

Fases, iterações e disciplinas



Relação com as Fases de Desenvolvimento e com as Outras Disciplinas

- Tem uma maior concentração na fase de concepção
- Nas iterações das fases seguintes, o nível de esforço é mantido constante
- Acontece em paralelo e com uma forte integração com a disciplina de planejamento e gerenciamento
- Algumas atividades relacionadas com a gerência da configuração ocorrem em outras disciplinas como a implementação e a implantação

Conclusões

- GC é um fluxo de apoio ao projeto como um todo
- Requer uma certa disciplina na manipulação de itens de configuração e apoio de ferramentas sempre que possível

Referências

- Descrição do workflow de gerência de configuração e mudanças do RUP
- Configuration Management Today - http://cmtoday.com
- Software Release Methodology, M.E.Bays, Prentice Hall, 1999.

Tarefa

- Pesquisa e apresentação de sistemas de controle de versão. Um sistema por grupo
  - Bazaar - http://bazaar.canonical.com/en/
  - CVS - http://savannah.nongnu.org/projects/cvs/
  - Subversion - http://subversion.apache.org/
  - Git - http://git-scm.com/