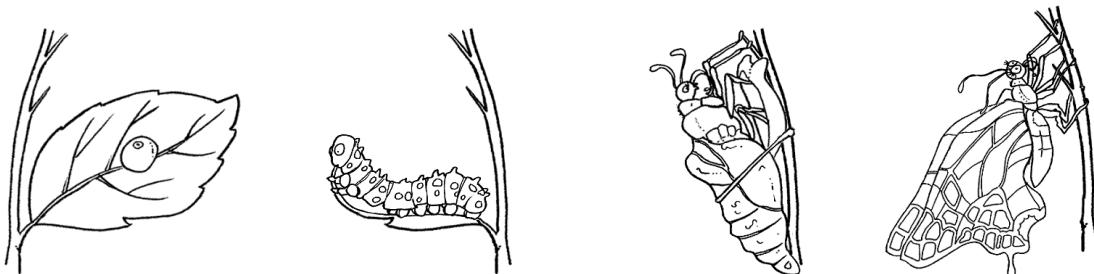


Gerência de Configuração

Leonardo Gresta Paulino Murta
leomurta@ic.uff.br

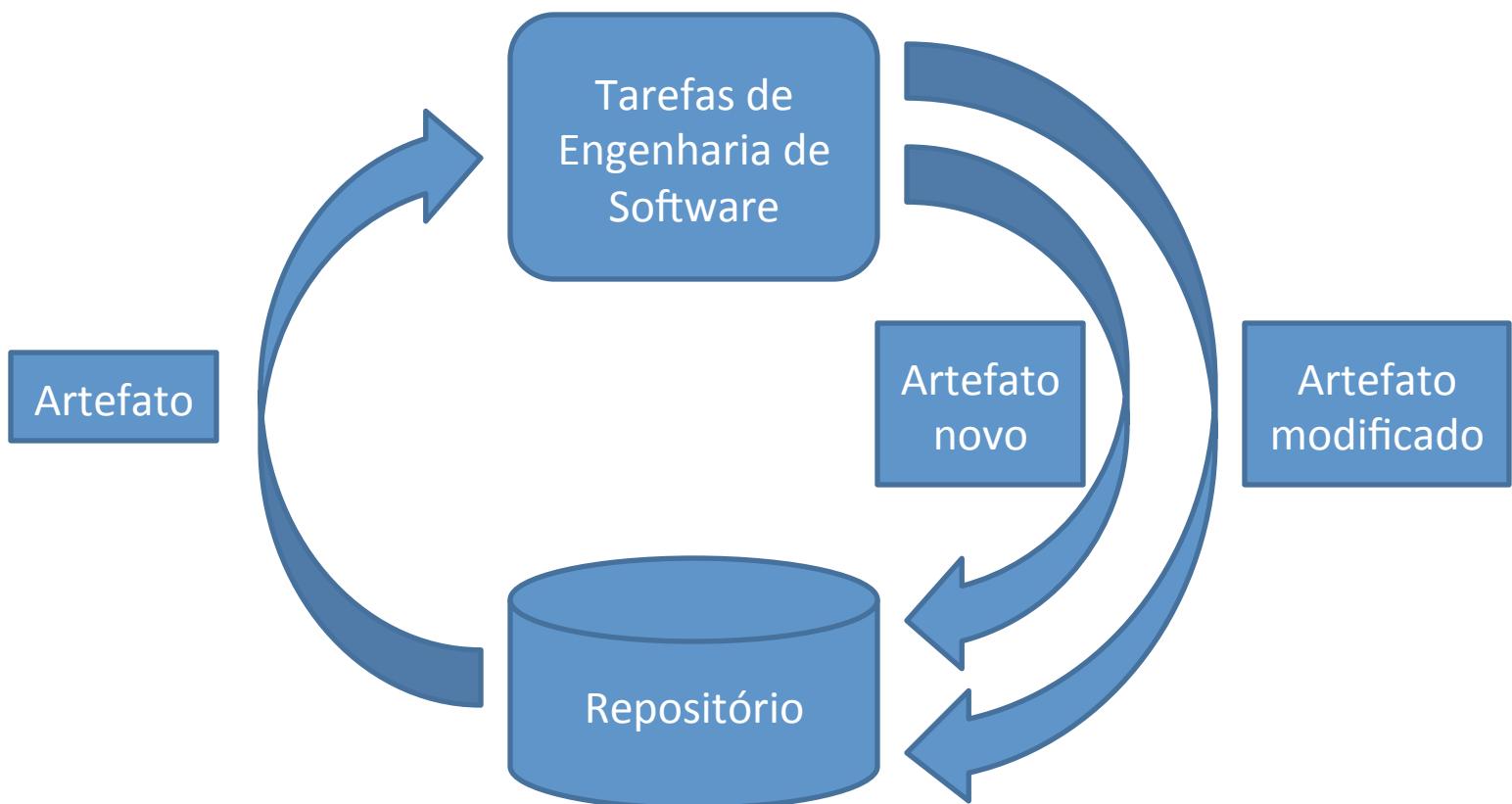
Introdução

- A Engenharia de Software...
 - Abordagem disciplinada para o desenvolvimento de software
 - Grande diversidade de metodologias
- Ponto em comum nas metodologias:
 - refinamentos sucessivos de artefatos



<http://www.colegiosaofrancisco.com.br>

Mas onde ficam esses artefatos?



O que são repositórios?

- Re却tórios
 - Lugar seguro onde versões de artefatos sã depositadas
 - Permitem armazenamento, busca e recuperaçõ
 - Servem como um ponto de referênci
 - Apóiam no aumento da memória organizacional



Gerência de Configuração

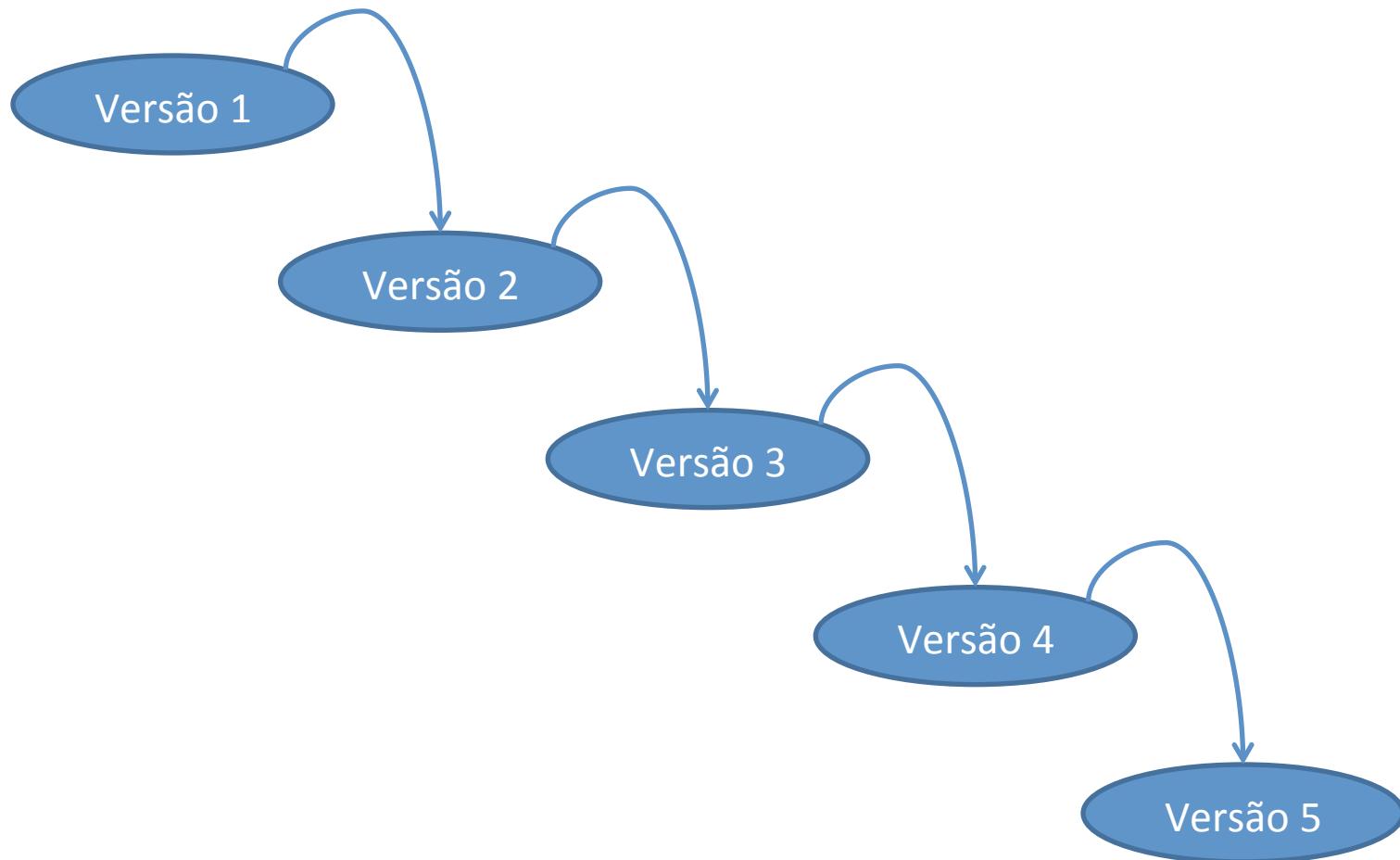
Gerência de configuração de software é uma **disciplina** para o **controle da evolução de sistemas de software** (Susan Dart, 1991)



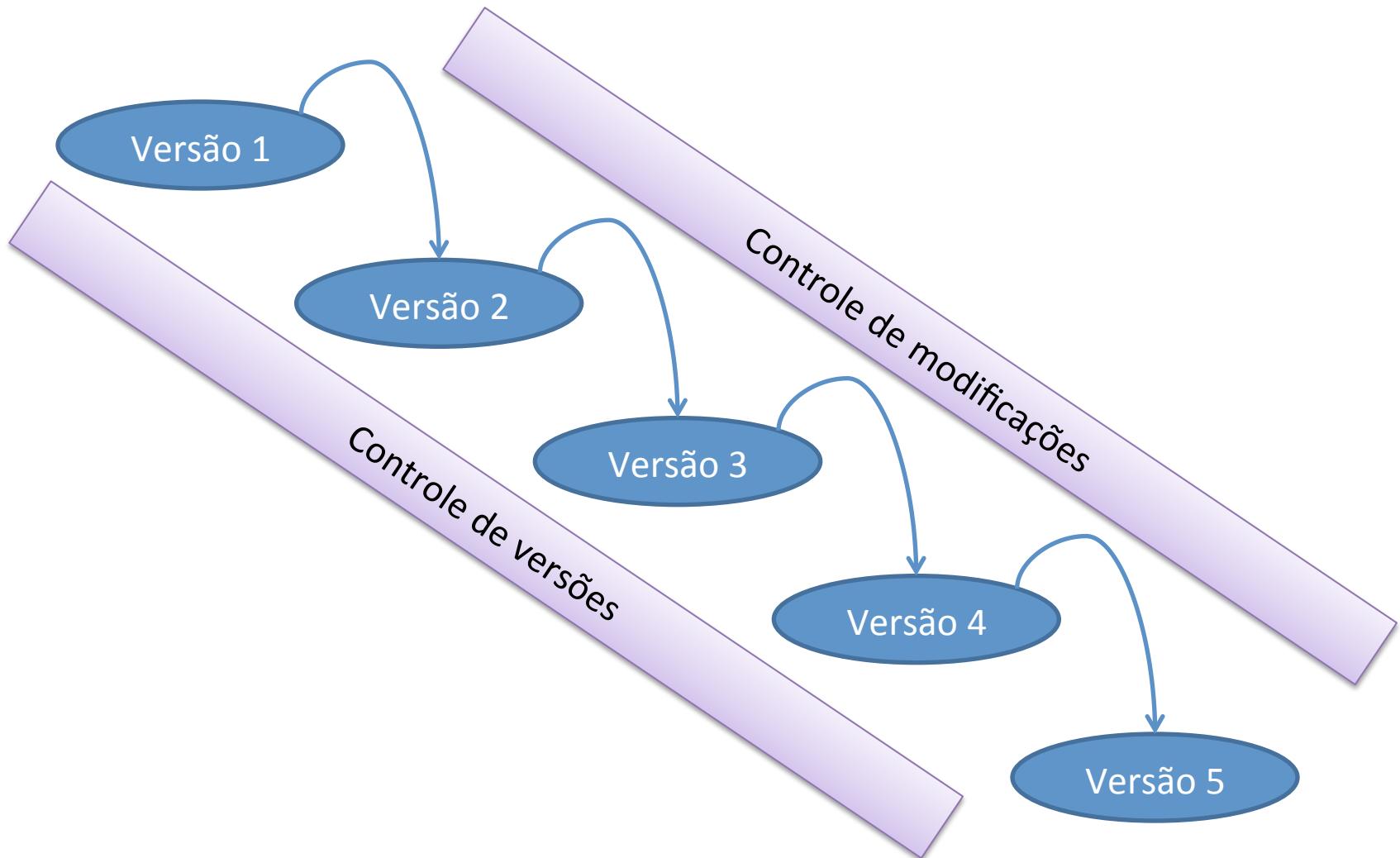
Histórico

- Anos 50
 - GC para produção de aviões de guerra e naves espaciais
- Anos 60 e 70
 - Surgimento de GCS (S = Software)
 - Foco ainda em aplicações militares e aeroespaciais
- Anos 80 e 90
 - Mudança de foco (MIL → EIA, IEEE, ISO, etc.)
 - Surgimento das primeiras normas internacionais
 - Assimilação por organizações não militares

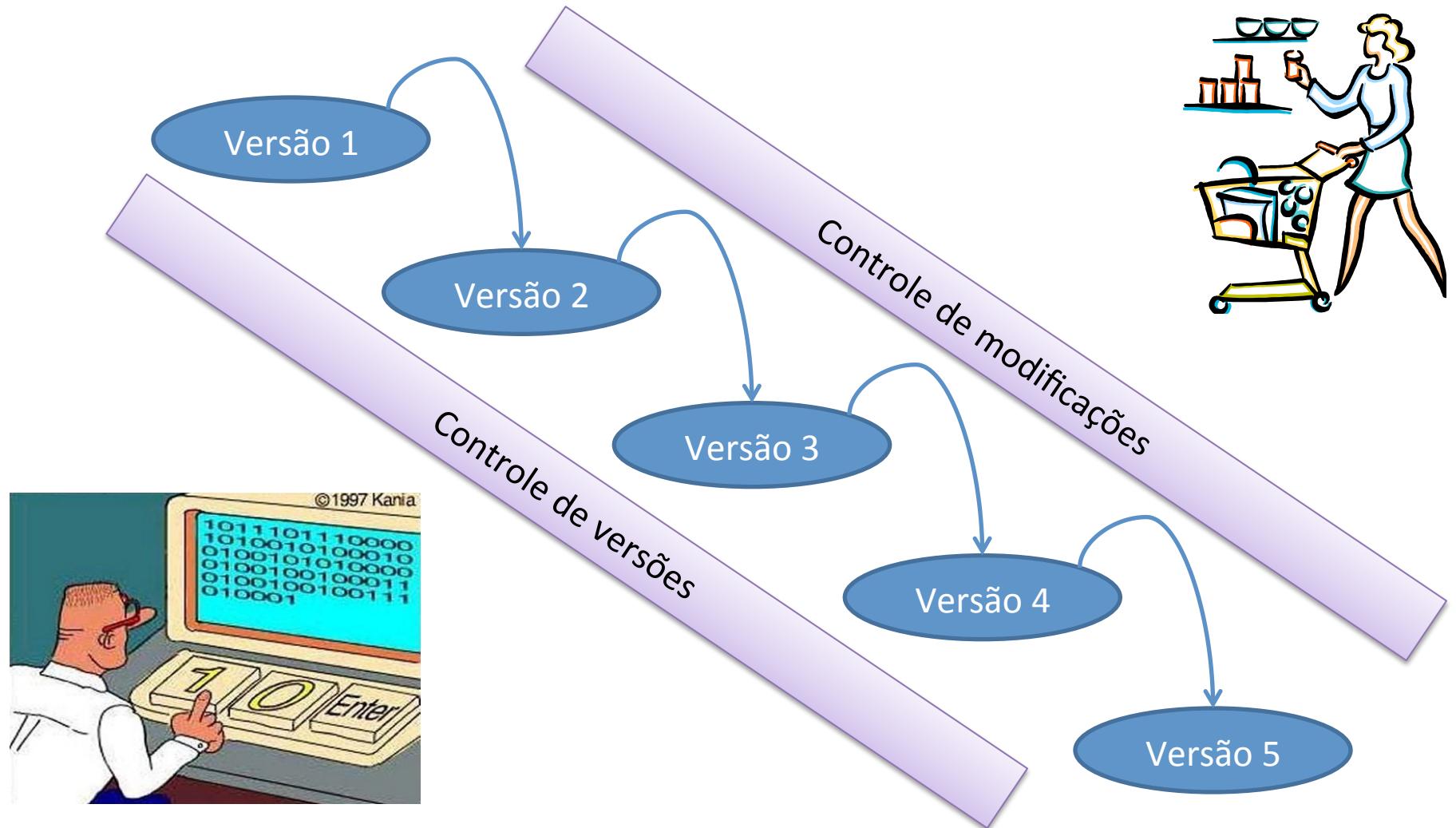
Sistema de Gerência de Configuração



Sistema de Gerência de Configuração



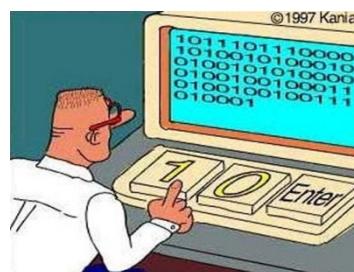
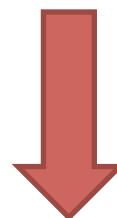
Sistema de Gerência de Configuração



Sistema de Gerência de Configuração



Controle de
Modificações



Controle de
Versões



Construção
e Release

Sistema x Funções de GC

Espaço de trabalho:

Ambiente de Desenvolvimento de Software

Processos:

Identificação Controle Contabilização Avaliação Liberação

Sistemas:

Controle de Modificações

Controle de Versões

Gerenciamento de Construção



Perspectiva de integração



Perspectiva gerencial

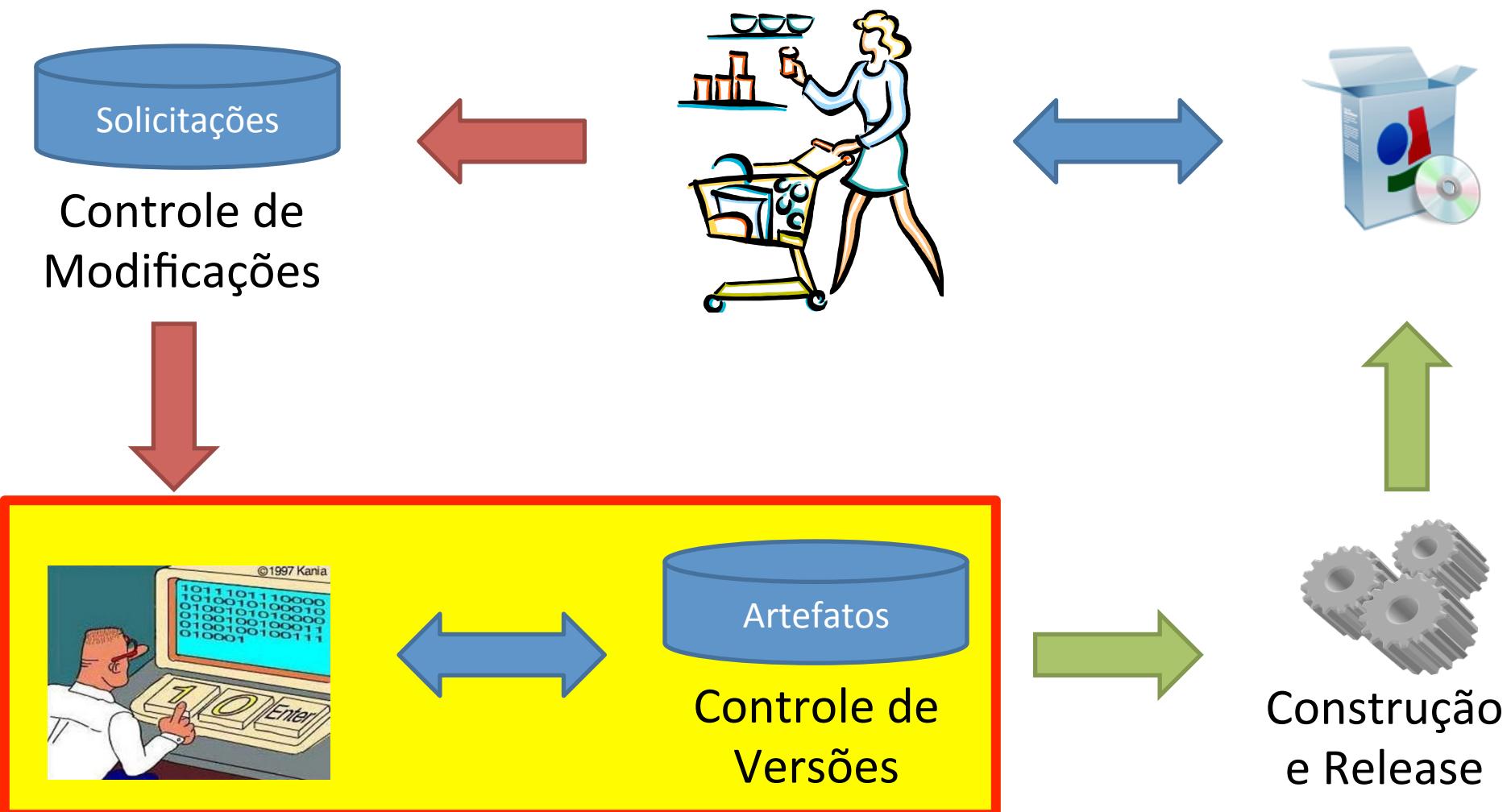


Perspectiva de desenvolvimento

Exercício

1. Descreva as 5 funções de gerência de configuração, citando exemplos
2. Estude uma ferramenta de gerenciamento de construção e release (make, ant, maven, etc.)
3. O que é integração contínua? Dê algum exemplo usando uma ferramenta (Cruise Control, Apache Continuum, Hudson, etc.)

Sistema de Gerência de Configuração

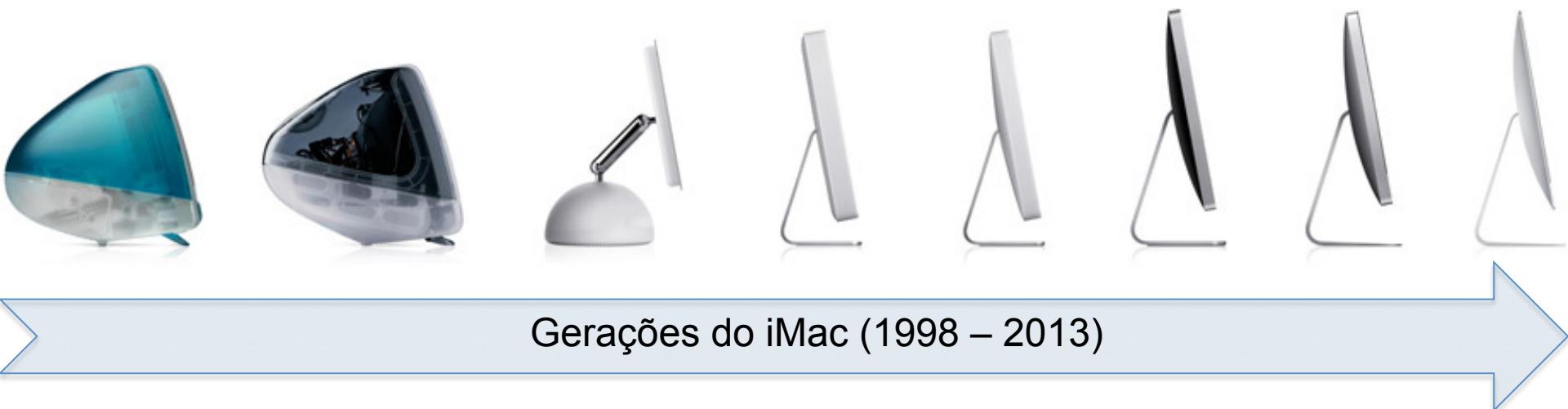


Tipos de Versão

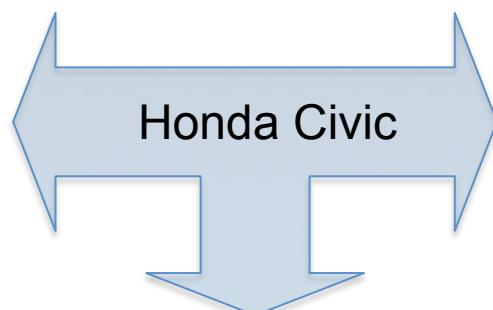
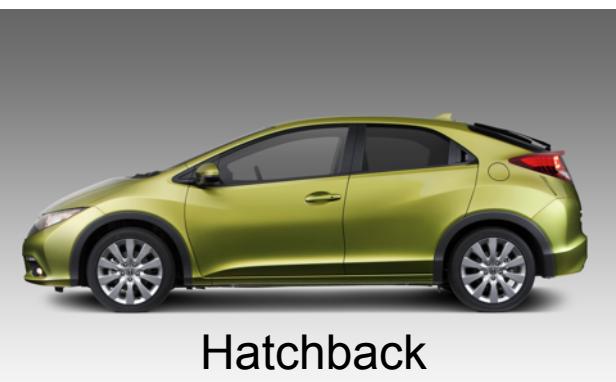


(Conradi and Westfechtel 1998)

Revisões



Variantes



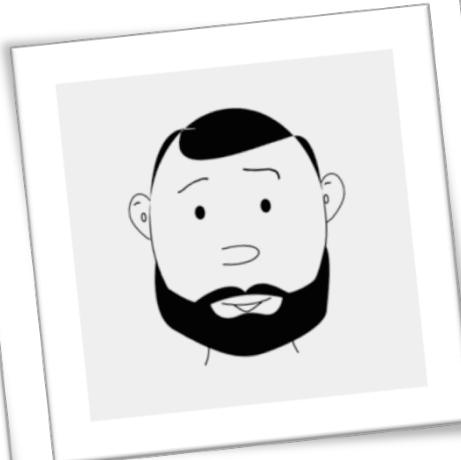
Cooperação (versões rascunho)



Versão base



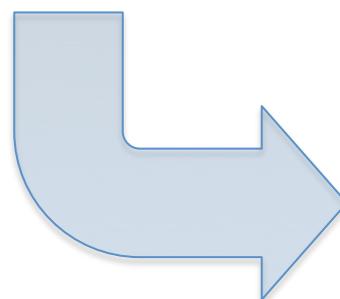
Espaço de trabalho do João



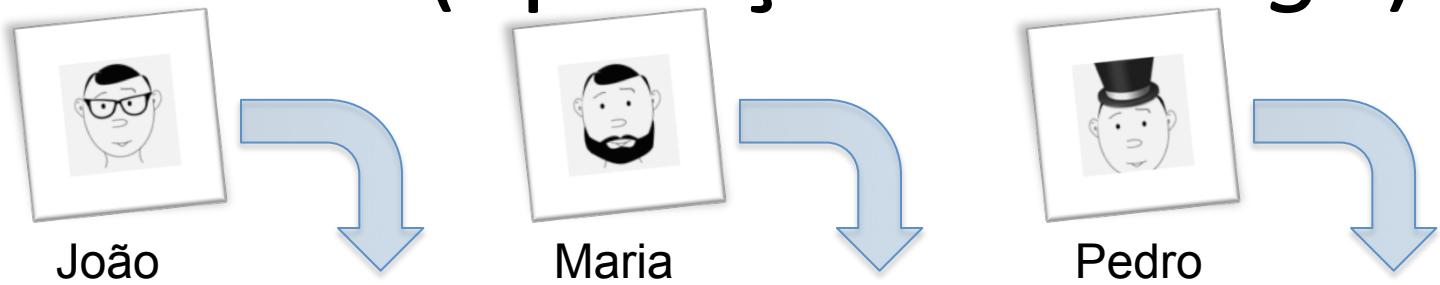
Espaço de trabalho da Maria



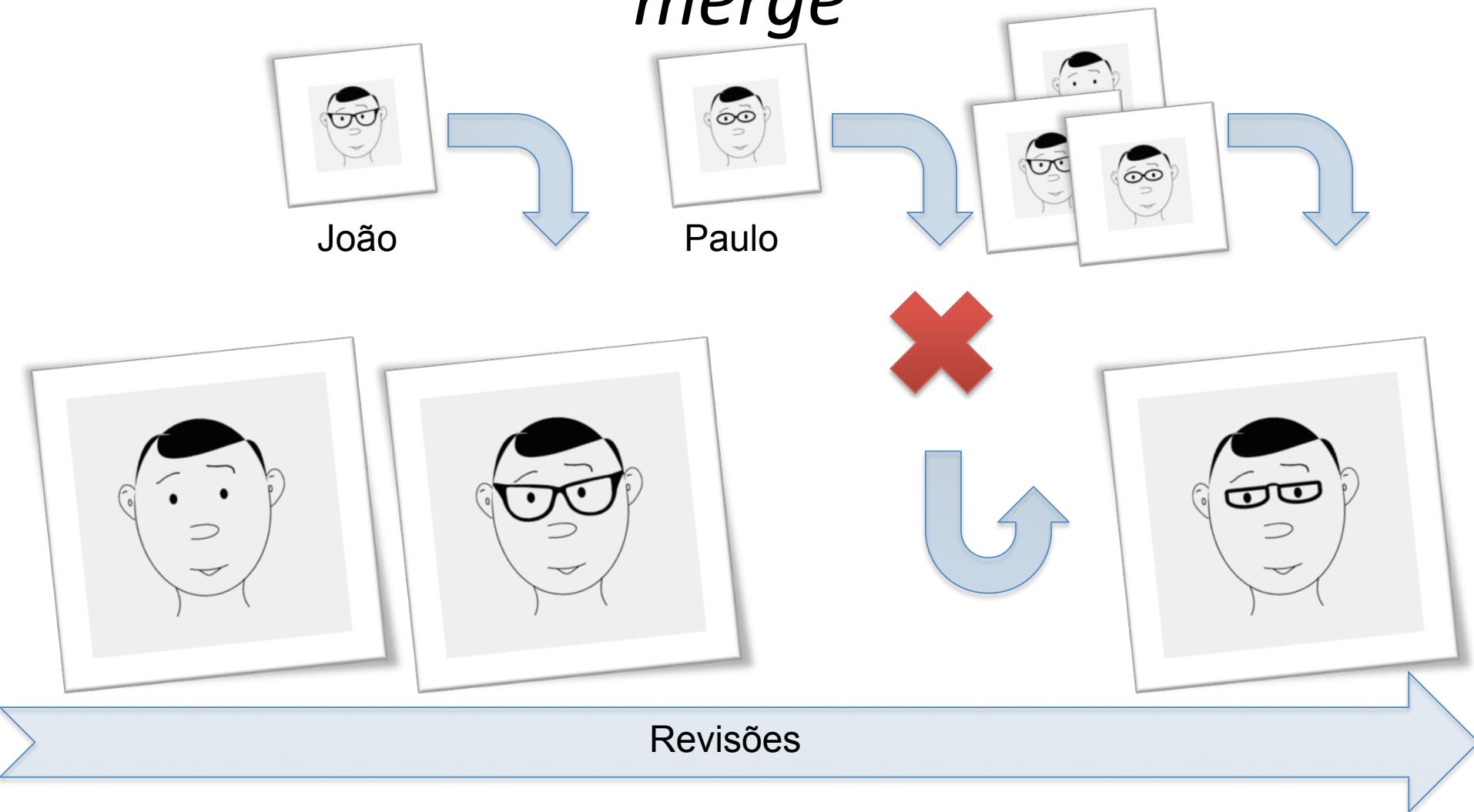
Espaço de trabalho
do Pedro



Versões de rascunho podem ser combinadas (operação de *merge*)



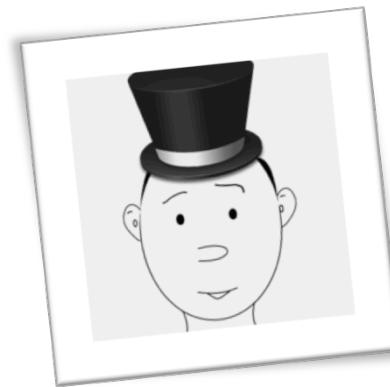
Conflitos podem ocorrer durante o *merge*



Outras duas operações importantes...



Diff



=



Patch



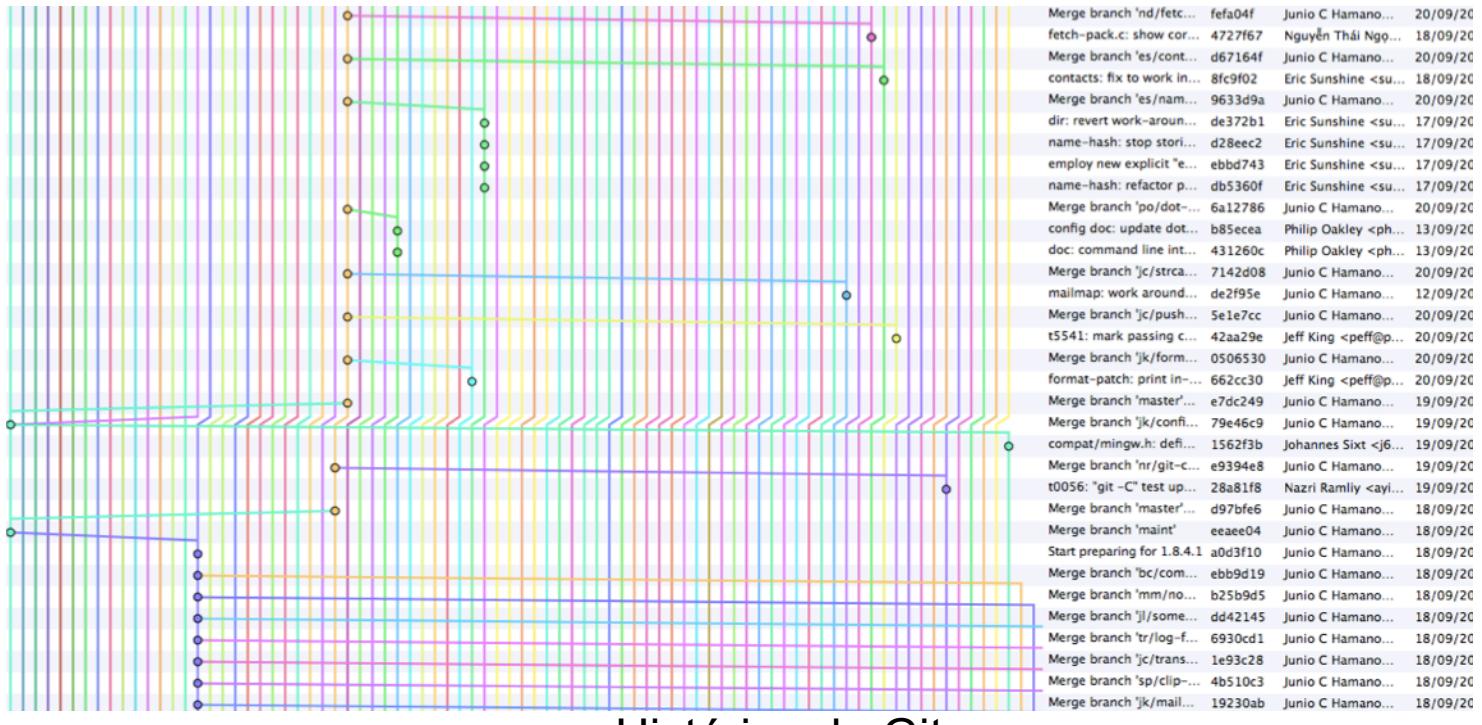
=



... para guardar, transferir e compreender versões.

Versões no mundo real

- Infinidade de revisões e variantes juntas (sem contar versões rascunho)

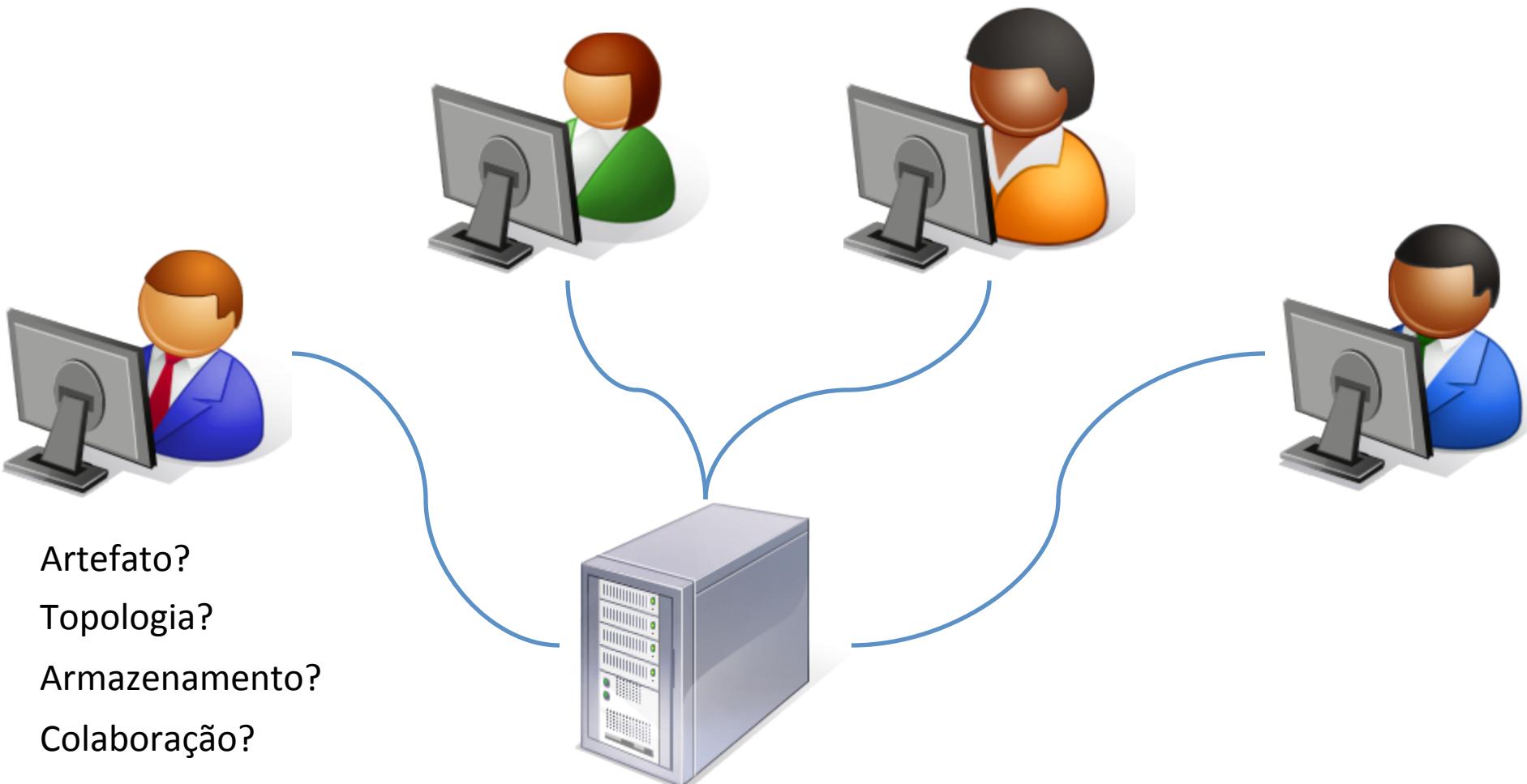


Histórico do Git

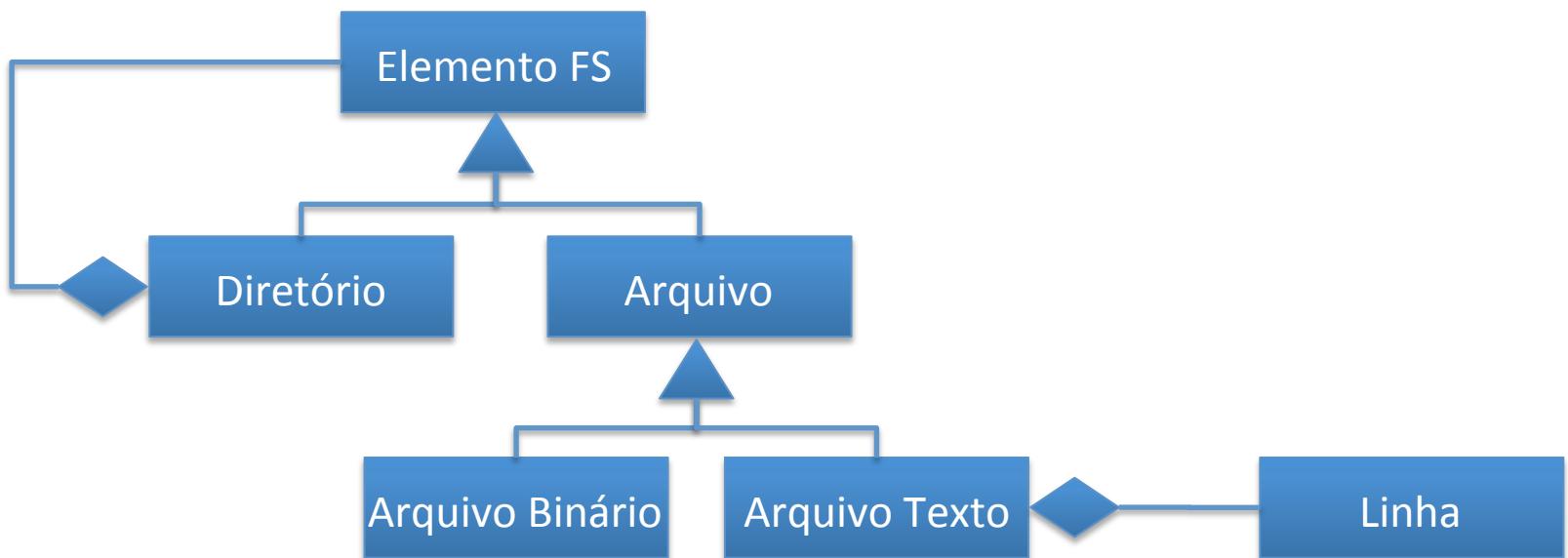
Mas afinal, para que servem versões?

- Sincronizar equipes
- Reproduzir configurações passadas
- Explorar possibilidades
- Segregar desenvolvedores
- Customizar produtos (LPS)
- Rastrear a introdução de bugs (bisect)
- Entender a evolução de software (MSR)
- Auditar mudanças (annotate)
- Etc.

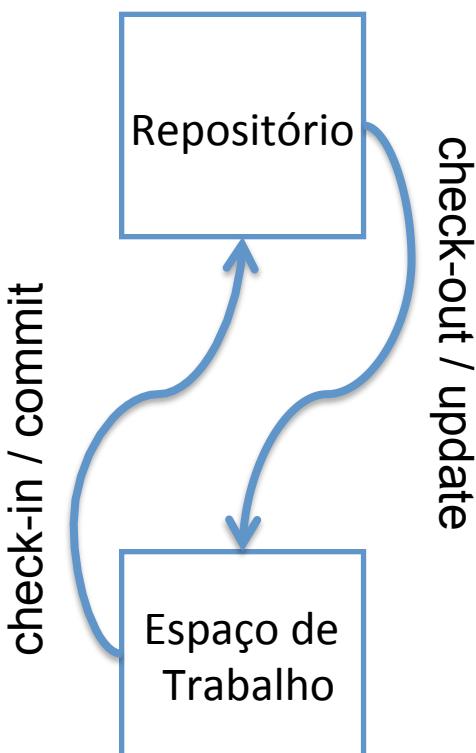
Controle de versões



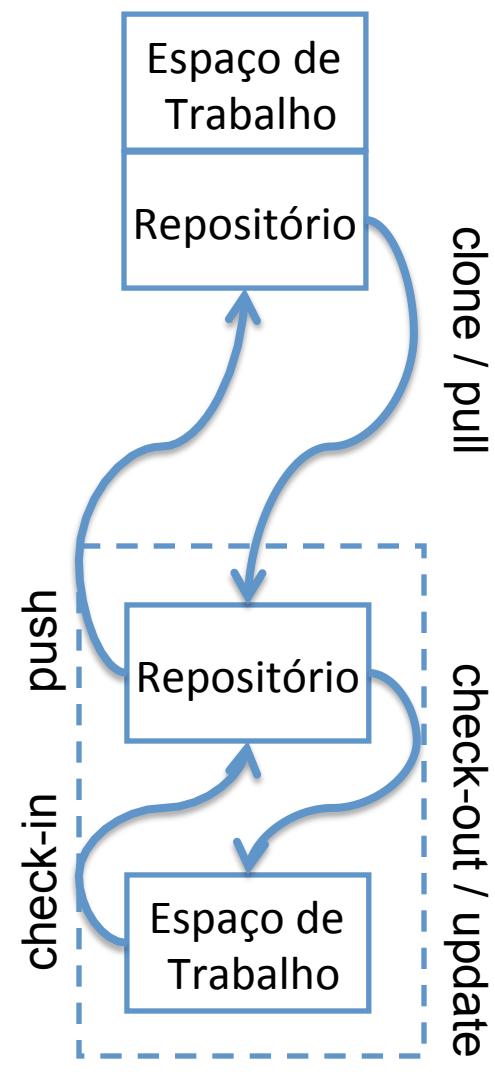
Artefato



Topologia

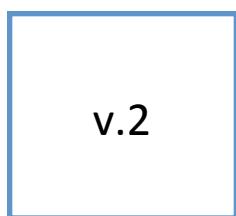


Centralizado



Distribuído

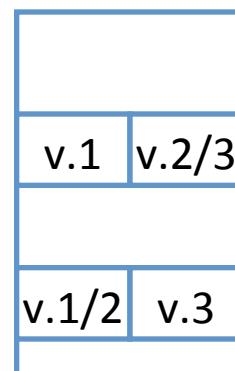
Armazenamento



Completo

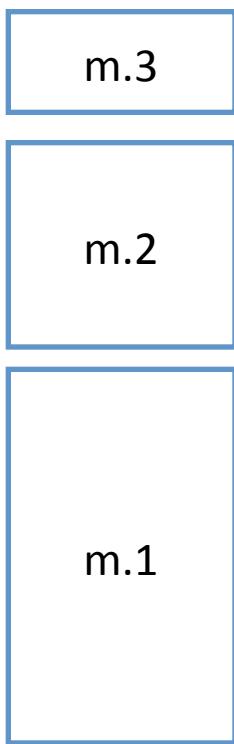
Forward

Reverse

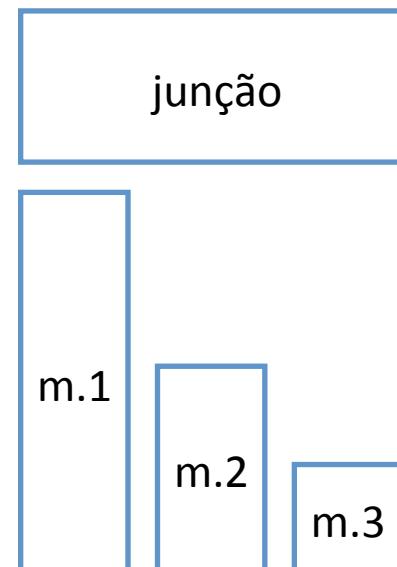


In-line

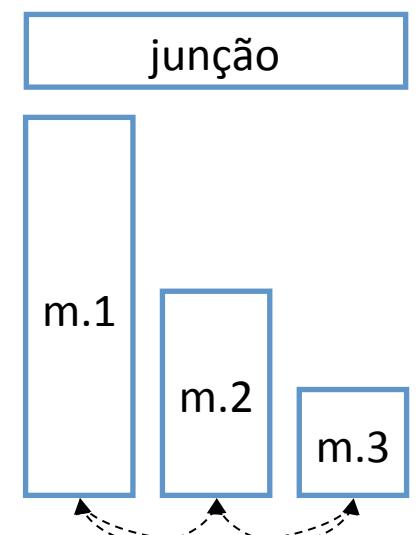
Colaboração



Pessimista

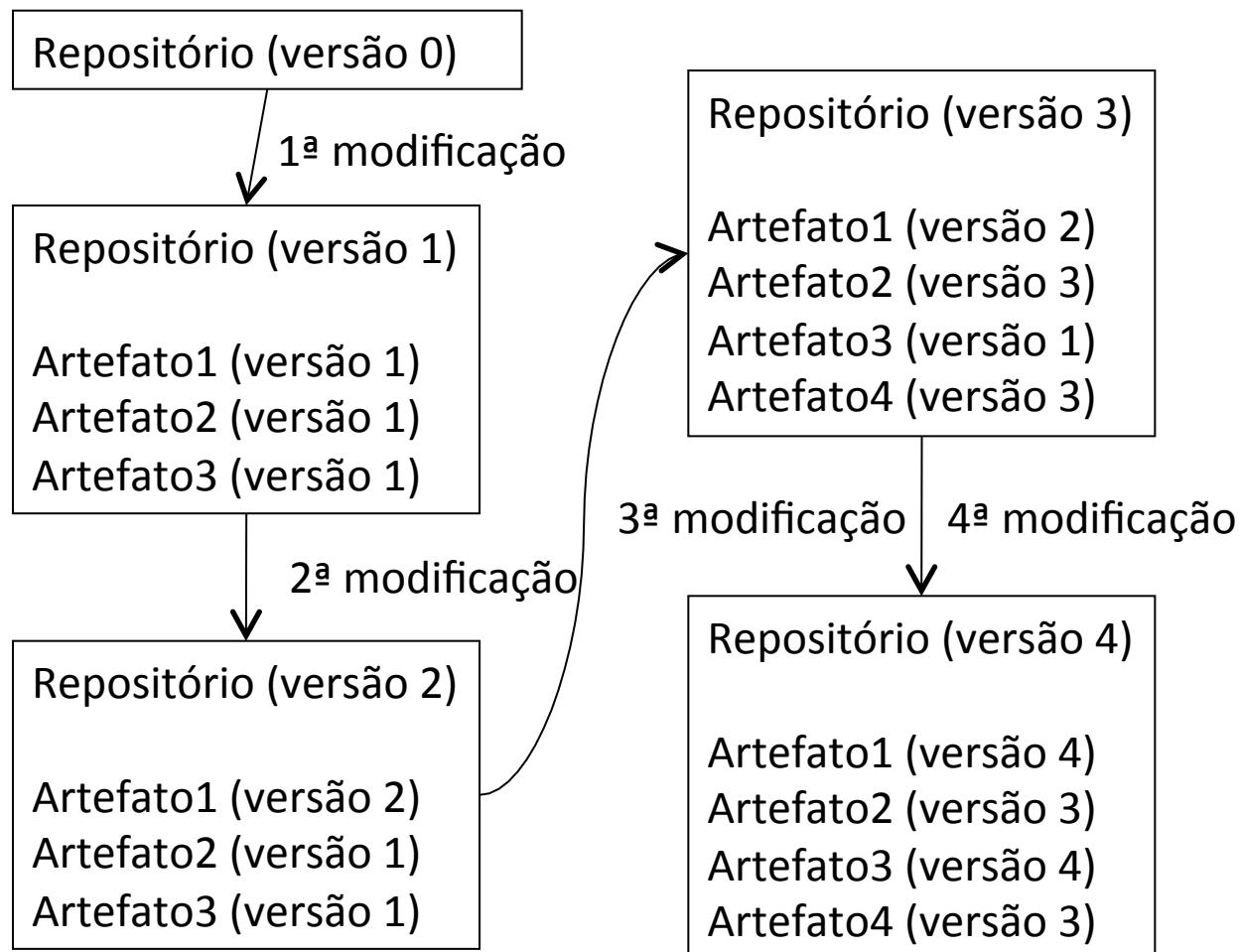


Otimista



Otimista com Percepção

Consulta



Consulta por artefato

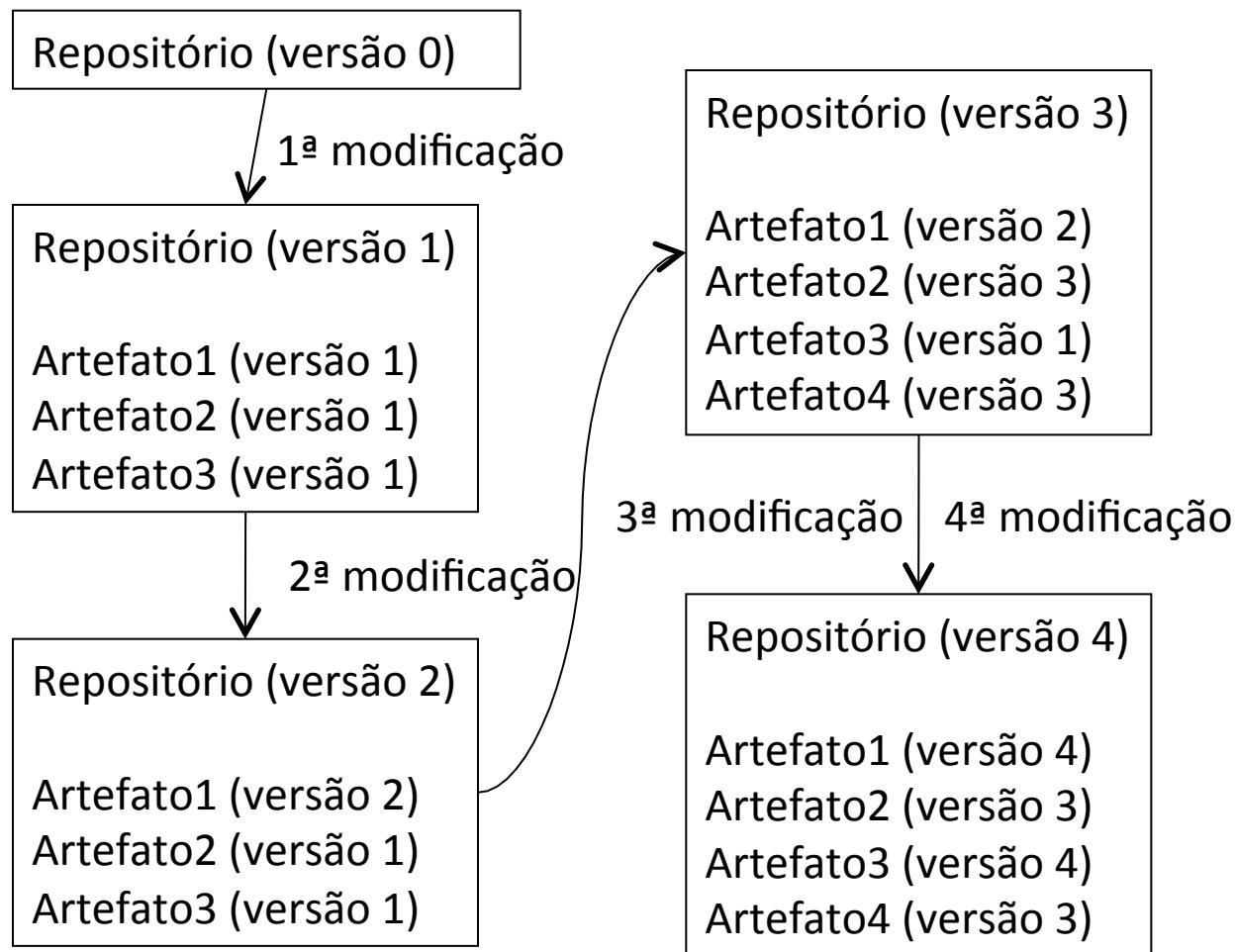
Artefato1
Versão 1
Versão 2
Versão 4

Artefato2
Versão 1
Versão 3

Artefato3
Versão 1
Versão 4

Artefato4
Versão 3

Consulta



Consulta por modificação

1ª modificação

- Artefato1 adicionado
- Artefato2 adicionado
- Artefato3 adicionado

2ª modificação

- Artefato1 modificado

3ª modificação

- Artefato2 modificado
- Artefato4 adicionado

4ª modificação

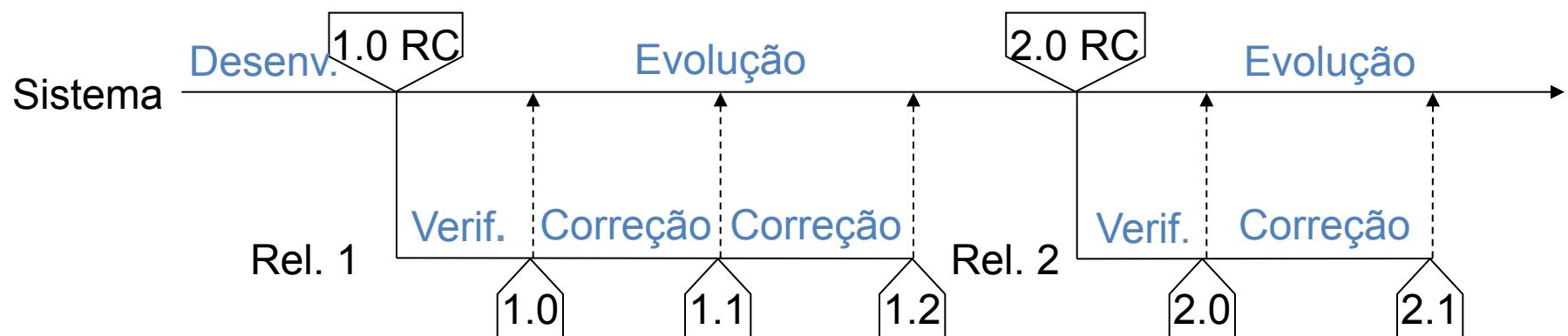
- Artefato1 modificado
- Artefato3 modificado

Tratamento de variantes em ramos (branches)

- Versões que não seguem a linha principal de desenvolvimento
- Fornecem isolamento para o processo de desenvolvimento
 - Ramos usualmente são migrados para a linha principal de desenvolvimento
 - A migração pode ser complicada no caso de isolamento longo
- Características dos ramos se comparados a espaços de trabalho
 - São compartilhados por outras pessoas (espaços de trabalho são isolados)
 - Residem no servidor (espaços de trabalho residem no cliente)
 - São históricos (espaços de trabalho são momentâneos)

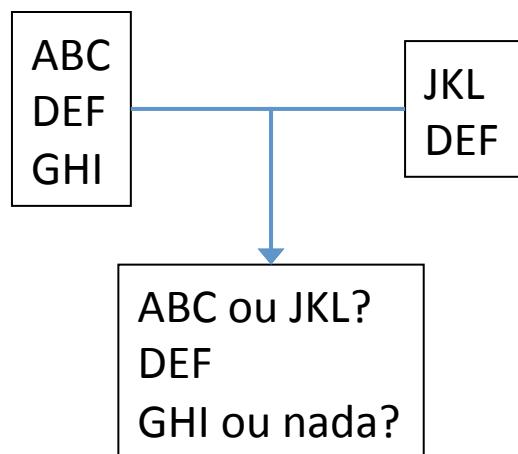
Estratégia básica de Ramificação

- Manutenção em série
 - Ramo principal: evolução
 - Ramos auxiliares: correções
- Foco
 - Desenvolvimento *in-house*
 - Cliente único (e.g.: aplicações Web)
- Dificuldade de manutenção de várias liberações em paralelo



Merge

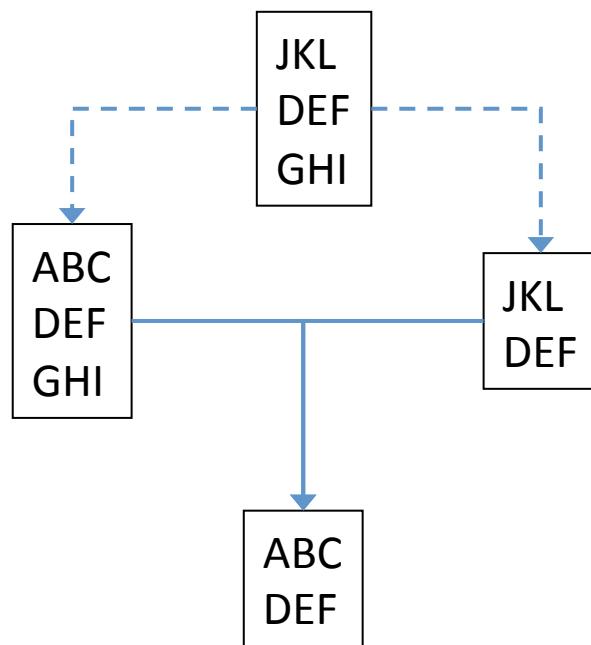
- Espaços de trabalho
- Ramos



2-way merge

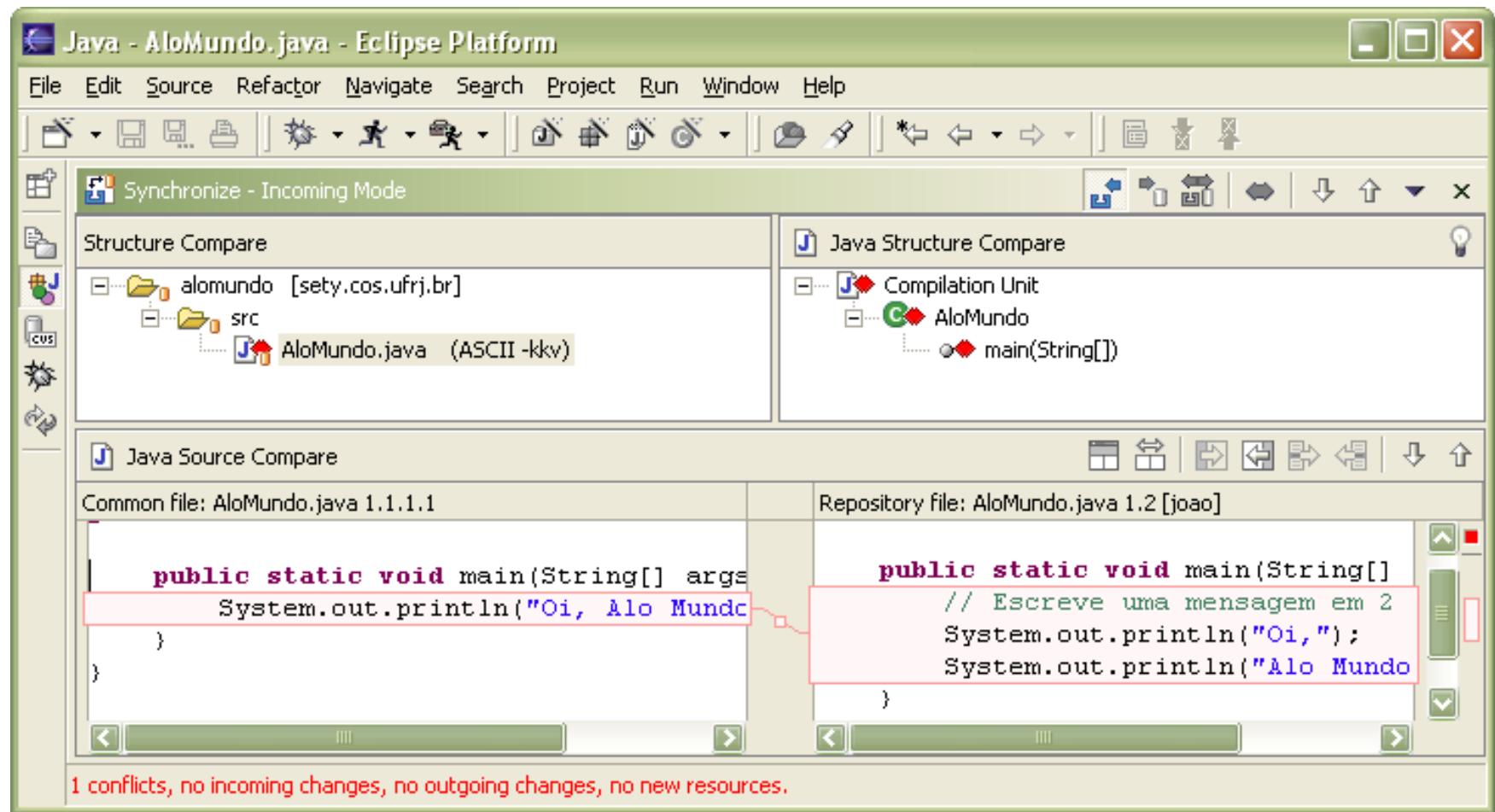
Merge

- Espaços de trabalho
- Ramos



3-way merge

Exemplo (merge no Eclipse)



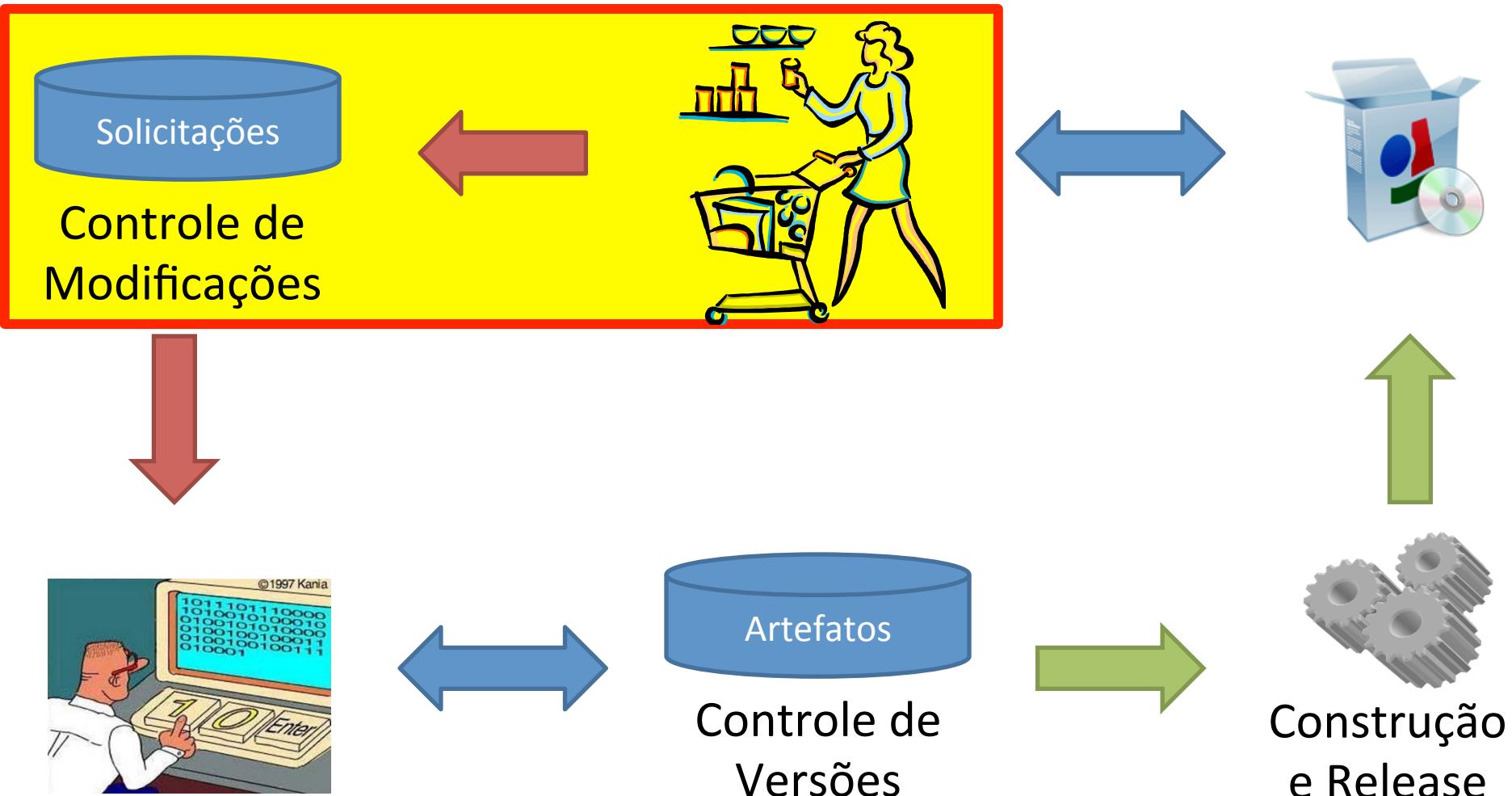
Principais sistemas de controle de versão open-source



git



Sistema de Gerência de Configuração



Baseline

- Configuração revisada e aprovada que serve como base para uma próxima etapa de desenvolvimento e que somente pode ser modificada via processo formal de GCS
- São estabelecidas ao final de cada fase de desenvolvimento
 - Análise (functional)
 - Projeto (allocated)
 - Implementação (product)
- Momento de criar: balanceamento entre controle e burocracia

Baseline (níveis de controle)



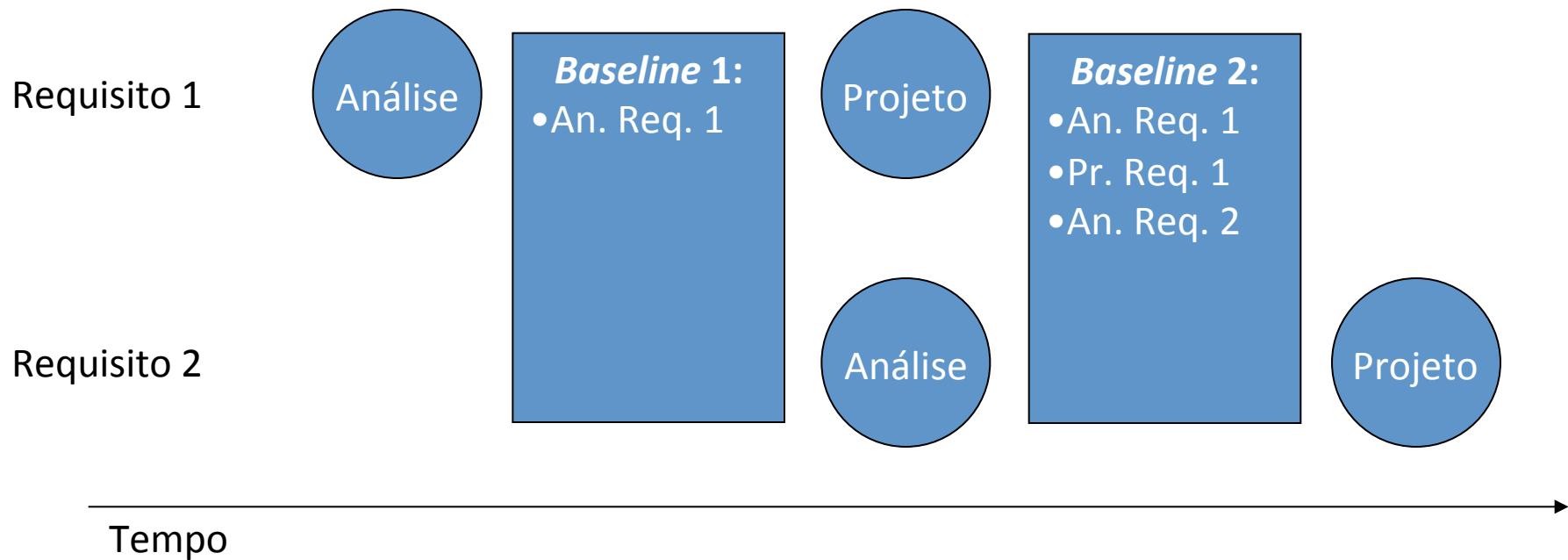
Pré baseline:

- Informal
- Sem requisição
- Sem aprovação
- Sem verificação
- Ágil
- Ad-hoc

Pós baseline:

- Formal
- Com requisição
- Com aprovação
- Com verificação
- Burocrático
- Planejado e Controlado

Baseline (níveis de controle)



Req.	Análise	Projeto
1	Inform.	-
2	-	-

Req.	Análise	Projeto
1	Inform.	-
2	-	-

Req.	Análise	Projeto
1	Formal	Inform.
2	Inform.	-

Req.	Análise	Projeto
1	Formal	Formal
2	Formal	Inform.

Controle de modificações

- Tarefas
 - Solicitação de modificação
 - Classificação da modificação
 - Análise da modificação
 - Avaliação da modificação
 - Implementação da modificação
 - Verificação da modificação
 - Geração de baseline

Controle de modificações

CHANGE REQUEST

CR No.: _____

Analysis Document No.: _____

System/project: _____ Item to be changed: _____

Classification: Enhancement / Bug fixing / Other: _____

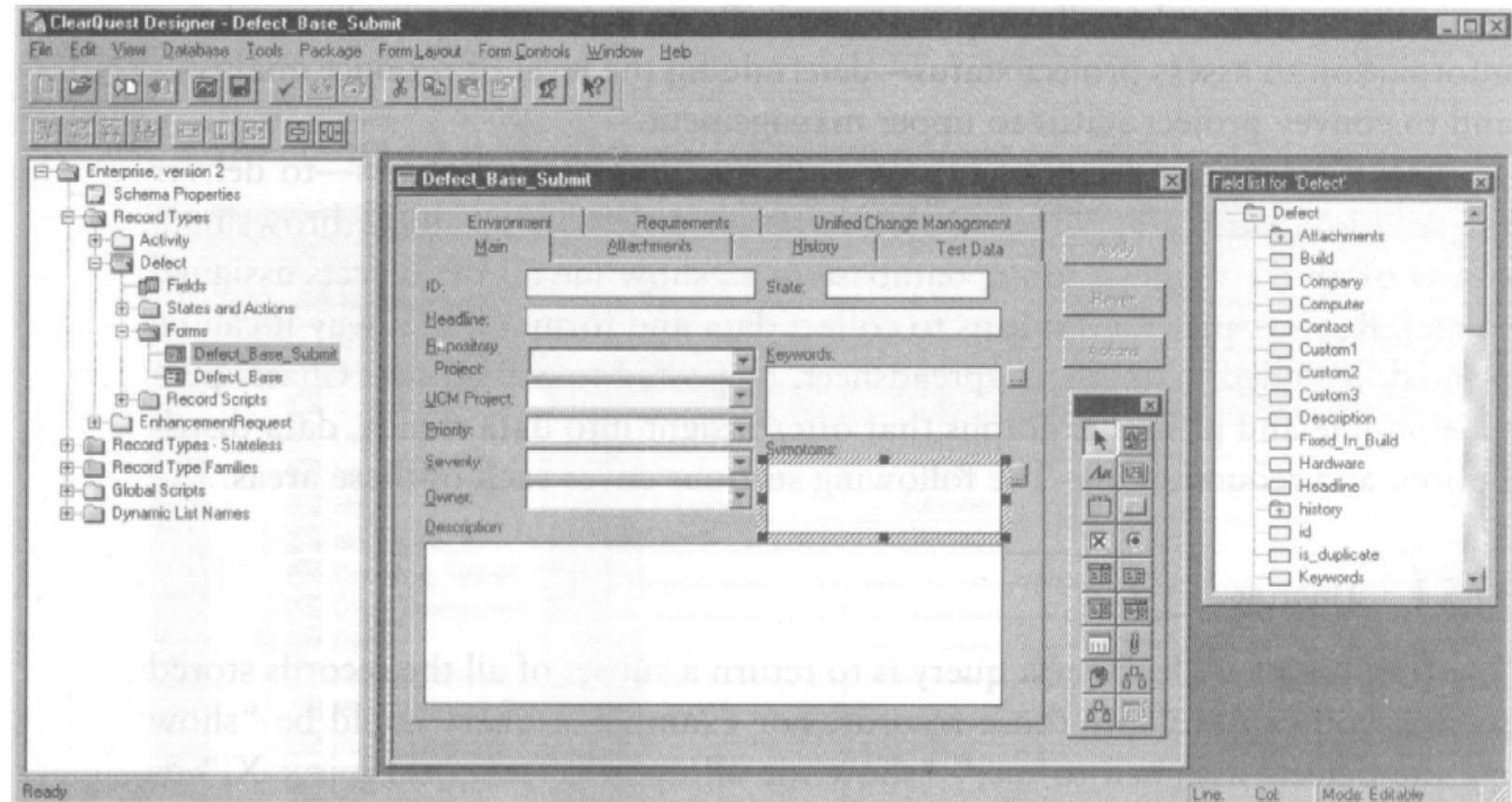
Priority: Immediate / Urgent / As soon as possible / Desirable

Change Description:

Status	Date	By	Remarks
Initiated			
Received			
Analyzed			
Action (A / R / D)*			
Assigned			
Check-out			
Modified and tested			
Reviewed			
Approved			
Check-in			
Baselined			

[Leon, 2000] Requisição de modificação

Controle de modificações



[White, 2000] Janela de criação de formulários do ClearQuest

Controle de modificações

- O critério de **classificação** da modificação deve estar explicitado no plano de GC
- 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
- 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

Controle de modificações

Change Analysis Document No.: _____

CR No.: _____

Date: _____

System/project: _____ Item to be analyzed: _____

Analyzed by: _____

Implementation alternatives:

Items affected

Item ID	Item description	Version no.	Nature of change

Estimated effort: _____

Impact on schedule: _____

Impact on cost: _____

Recommendation:

[Leon, 2000] Análise de modificação

Controle de modificações

- A **avaliação** utilizará a requisição de modificação e o laudo da análise para tomar a decisão
 - A requisição pode ser aceita, rejeitada ou adiada
- A **implementação** deve ser seguida por testes de unidade
- Durante a **verificação**, devem ser aplicados testes de sistema
- Após a **geração** da nova *baseline*, deve ser decidido se ela será considerada uma nova liberação

Controle de modificações

- Caso especial: Correções emergenciais
 - No caso de correções emergenciais, podem ser criados ramos sem a necessidade do processo formal
 - Em algum momento esses ramos deverão sofrer junção para a linha principal de desenvolvimento
 - Esse procedimento deve estar explicitado no processo!

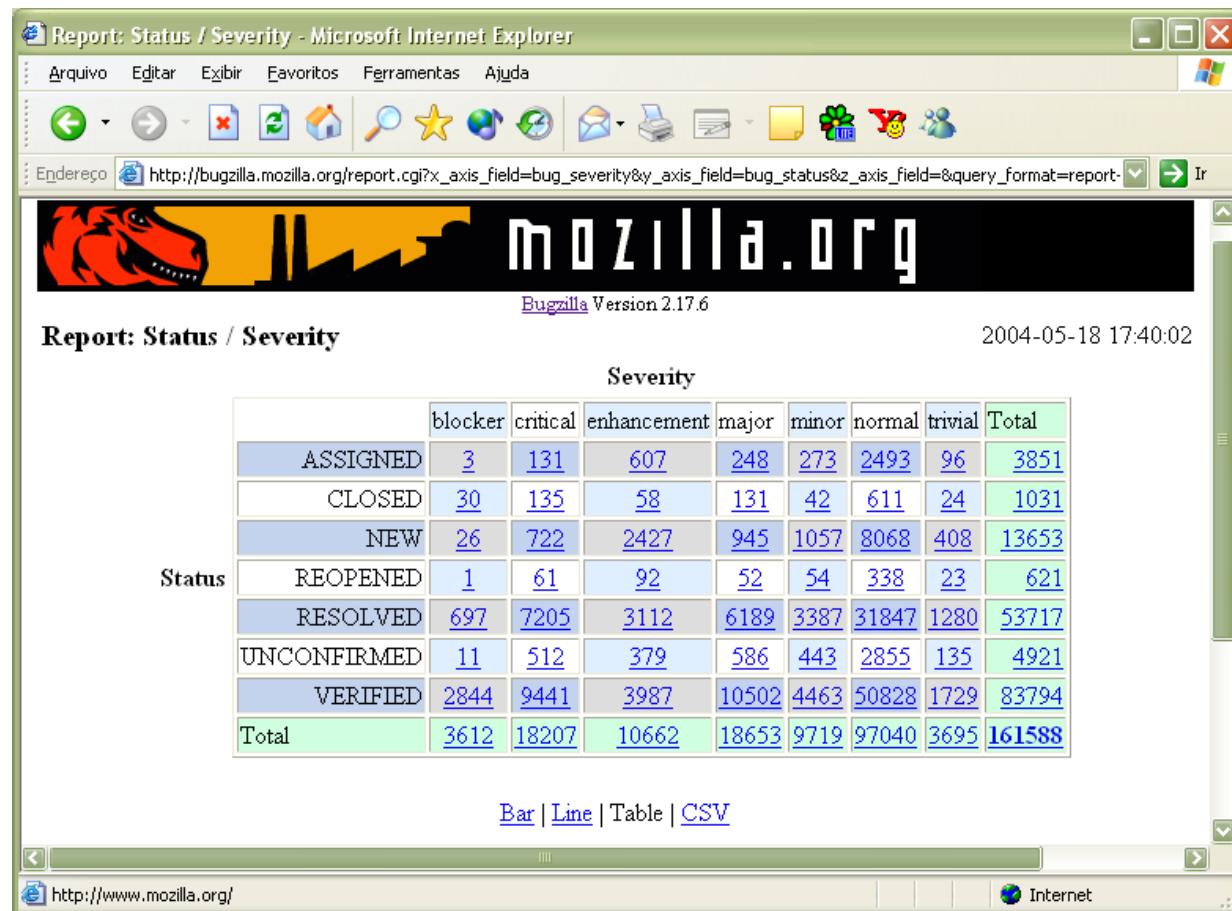
Controle de modificações

- Caso especial: Defeitos
 - Alguns sistemas tratam defeitos de forma diferente das demais requisições
 - A correção de defeitos é um tratamento sintomático
 - É importante descobrir o real motivo para o acontecimento do defeito para possibilitar a prevenção de defeitos futuros
 - A análise de causa é útil para descobrir falhas no processo de desenvolvimento (e.g. falta de treinamento, padrões inadequados, ferramentas inadequadas)

Contabilização da situação

- Tarefas
 - Armazenamento das informações geradas
 - Propagação dessas informações aos interessados através de relatórios
- Metáfora de conta bancária para item de configuração
- Permite que métricas sejam utilizadas com o intuito de melhoria do processo e estimativa de custos futuros
- Fornece relatórios gerenciais *ad-hoc*

Contabilização da situação



Report: Status / Severity

Bugzilla Version 2.17.6

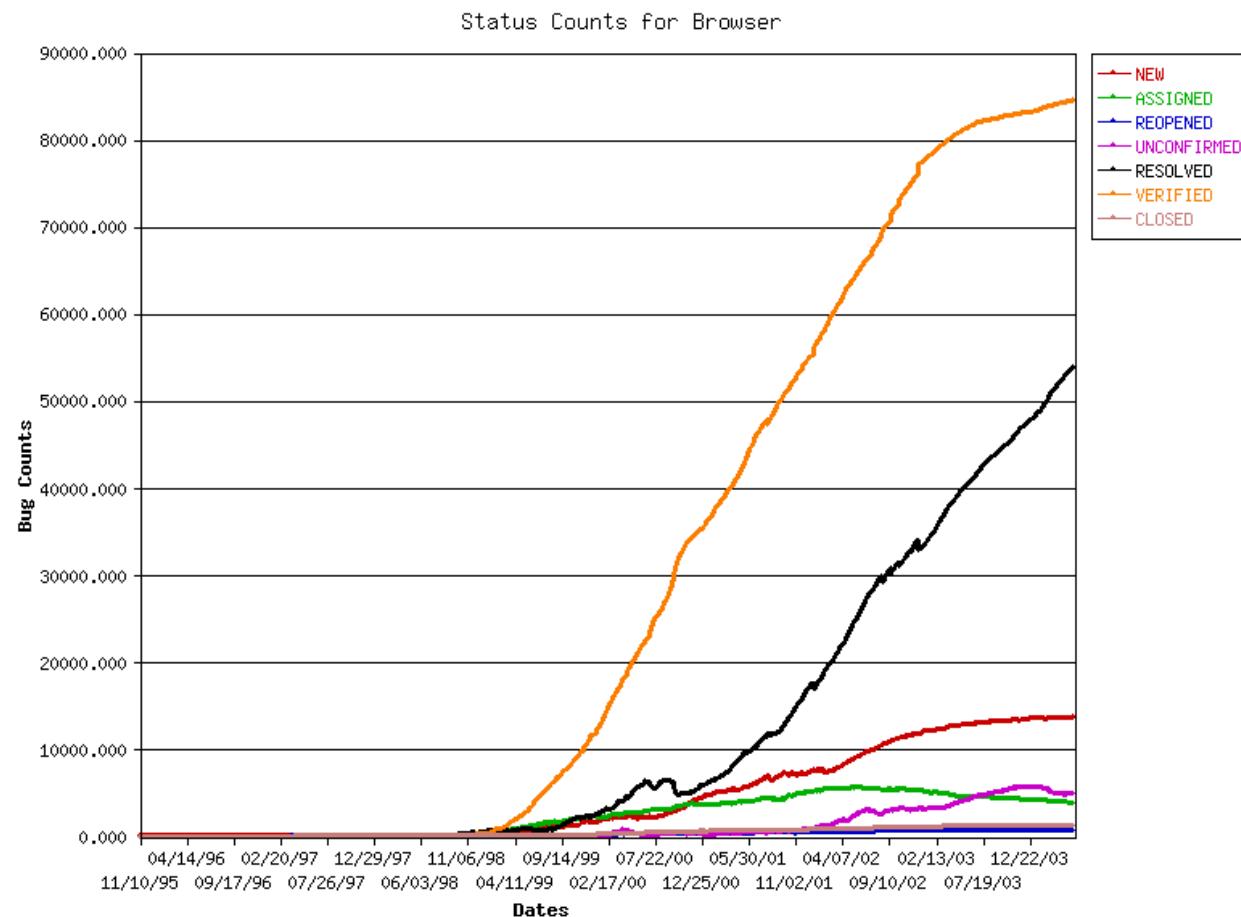
2004-05-18 17:40:02

		Severity							
		blocker	critical	enhancement	major	minor	normal	trivial	Total
Status	ASSIGNED	3	131	607	248	273	2493	96	3851
	CLOSED	30	135	58	131	42	611	24	1031
	NEW	26	722	2427	945	1057	8068	408	13653
	REOPENED	1	61	92	52	54	338	23	621
	RESOLVED	697	7205	3112	6189	3387	31847	1280	53717
	UNCONFIRMED	11	512	379	586	443	2855	135	4921
	VERIFIED	2844	9441	3987	10502	4463	50828	1729	83794
Total	3612	18207	10662	18653	9719	97040	3695	161588	

[Bar](#) | [Line](#) | [Table](#) | [CSV](#)

Resultado do relatório no modo tabular no Bugzilla

Contabilização da situação

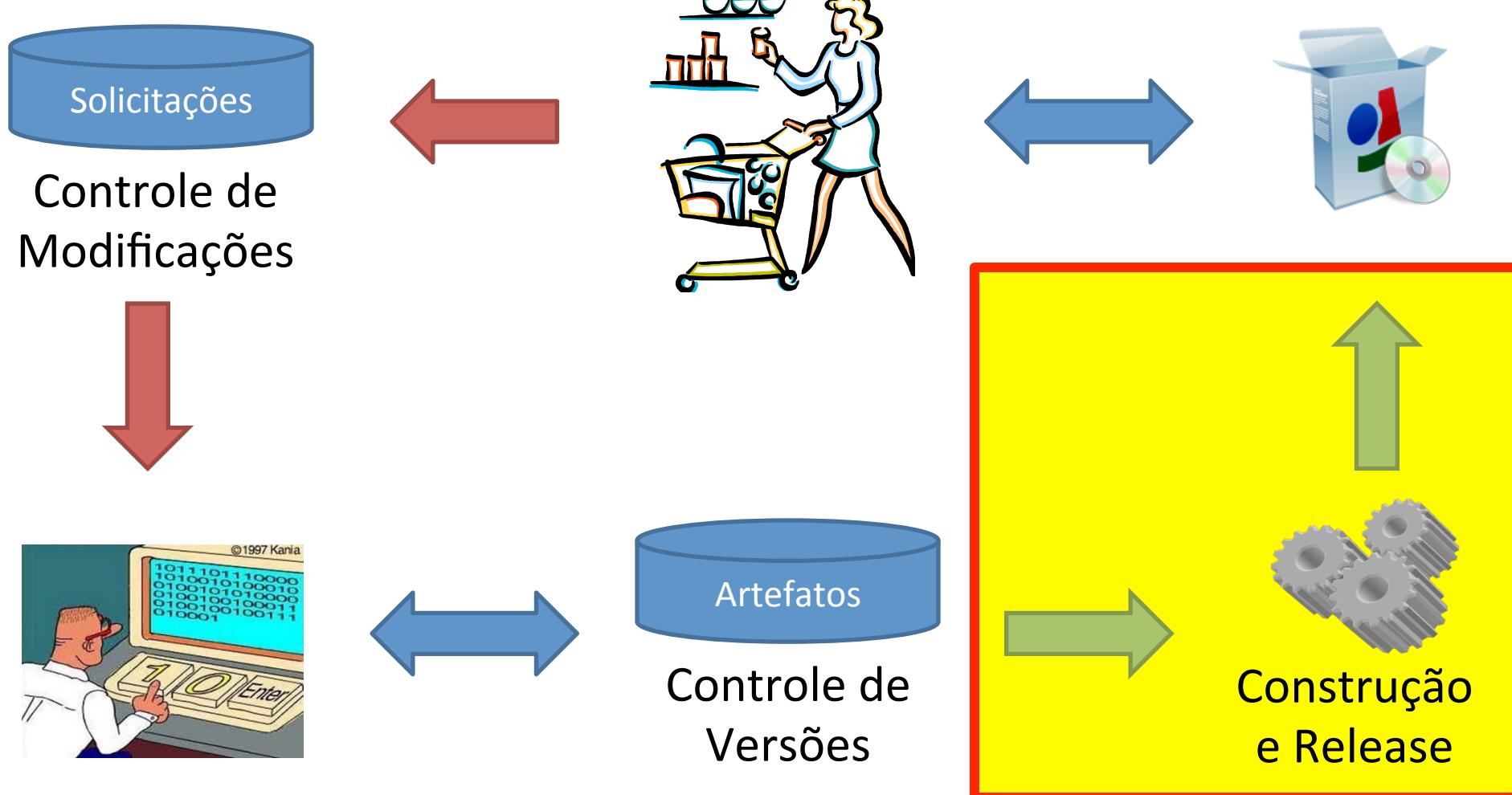


Resultado da consulta sobre séries no Bugzilla

Exemplo de ferramentas de controle de modificações

- Livre
 - Bugzilla
 - Mantis
 - Redmine
 - Trac
- Comercial
 - ClearQuest (IBM Rational)
 - JIRA (Atlassian)
 - StarTeam (Borland)
 - Synergy/Change (Telelogic)
 - TeamTrack (Serena)
 - Team Foundation Server (Microsoft)

Sistema de Gerência de Configuração



Auditoria da configuração

- Deve ocorrer ao menos antes de uma liberação (*release*)
- Tarefas
 - Verificação funcional, assegurando que a *baseline* cumpre o que foi especificado
 - Verificação física, assegurando que a *baseline* é completa (todos os itens de configuração especificados)
- Auditorias servem para garantir que os procedimentos e padrões foram aplicados

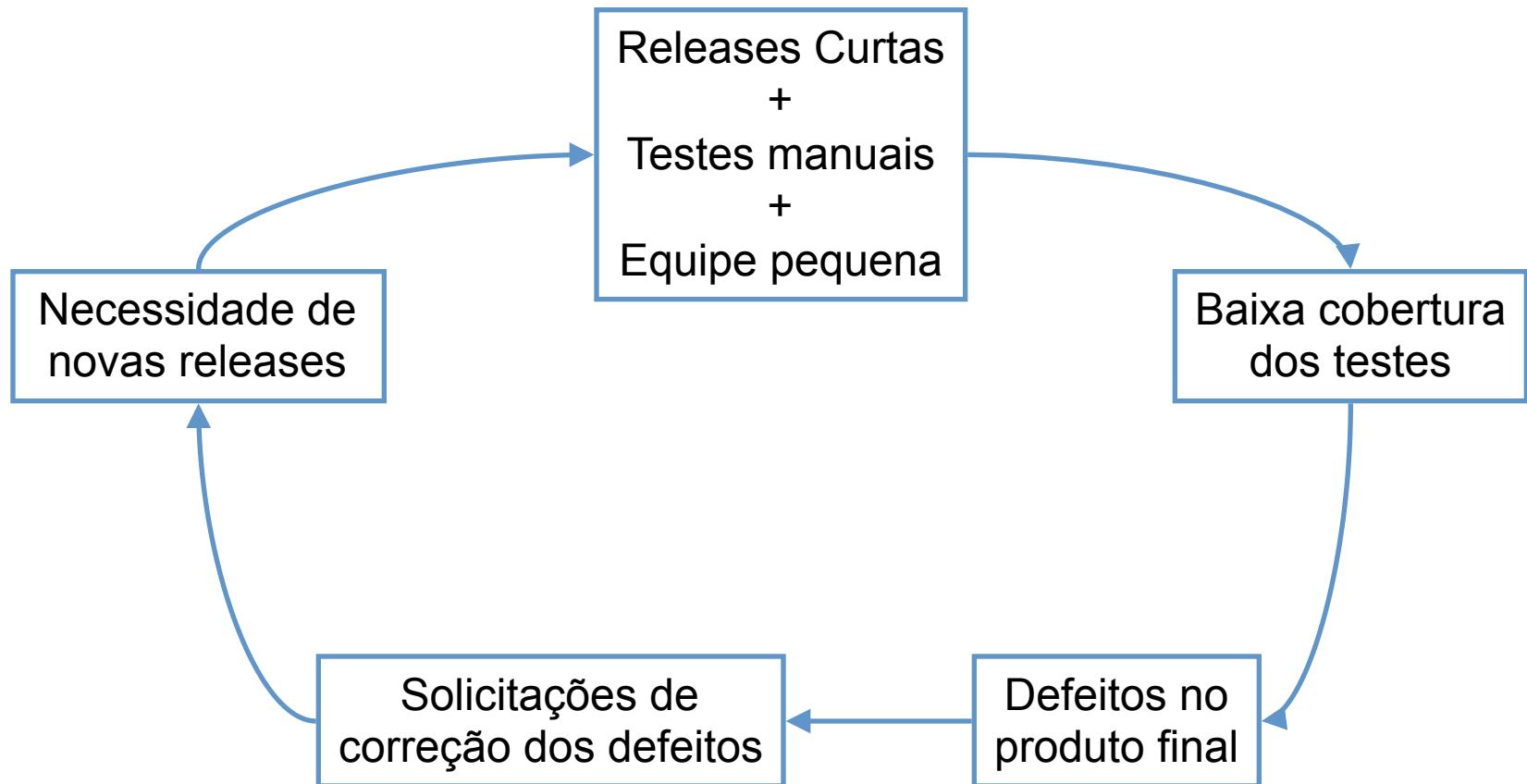
Auditoria da configuração

- 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 *baseline*
- A auditoria física é efetuada após a auditoria funcional
- Podem ocorrer **auditorias no próprio sistema de GC** pelos mantenedores do plano de GC, para verificar se as políticas e procedimentos estão sendo cumpridos

Gerenciamento de *releases*

- Descrição de como construir, liberar e entregar o sistema
 - Linguagem natural (conhecimento)
 - Linguagem computacional (automação)
 - Manter os descritores e documentos sob gerência de configuração!
- Definição das situações onde o processo pode ser temporariamente desviado
- Cuidado: *Releases* muito curtas podem levar a círculo-vicioso de defeitos...

Gerenciamento de *releases*

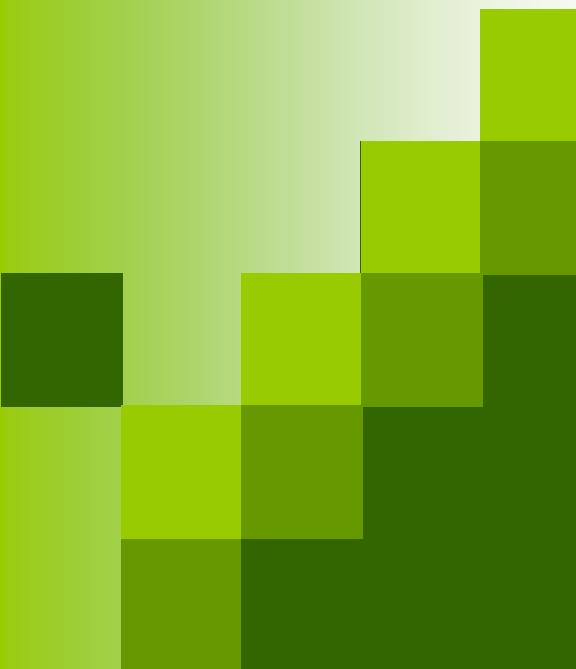


Exemplo de ferramentas de controle de construção e liberação

- Livre
 - Ant
 - NAnt
 - Make
 - Maven
 - Rake
- Comercial
 - ClearMake (IBM Rational)
 - MSBuild (Microsoft)
 - Synergy/CM Object Make (Telelogic)

Principais Referências Bibliográficas

- Alexis Leon, “A Guide to Software Configuration Management”, Artech House Publishers, 2000.
- Anne Hass, “Configuration Management Principles and Practices”, Boston, MA, Pearson Education, Inc.
- Conradi, R. and Westfechtel, B. Version Models for Software Configuration Management. ACM Computing Surveys, v.30, n.2, p. 232-282, 1998.
- Dart, S., 1991, “Concepts in Configuration Management Systems”, International Workshop on Software Configuration Management (SCM), Trondheim, Norway (June), pp. 1-18.
- Pressman, R. S. (1997). “Software Engineering: A Practitioner's Approach”, McGraw-Hill.



Gerência de Configuração

Leonardo Gresta Paulino Murta
leomurta@ic.uff.br