Faculdade de Design,
Tecnologia e Comunicação
Universidade Europeia

Sistemas de Versionamento e Plataformas de Hospedagem de Repositórios

Fundamentos da Programação

Fernando Marson

fernando.marson@universidadeeuropeia.pt

O que é Controlo de Versão?

- É um sistema que **registra as mudanças** feitas **em um ficheiro** ou **conjunto de ficheiros ao longo do tempo**, permitindo que você **retorne** a versões anteriores, **compare alterações** e **colabore** com outras pessoas de forma eficiente.

- Este tipo de sistema é especialmente útil no **desenvolvimento de software**, onde várias pessoas podem estar trabalhando no mesmo código ao mesmo tempo.

Características Principais

- Rastreamento de Mudanças
- Colaboração
- Reversão
- *Branches*

Características Principais

- **Rastreamento de Mudanças**

  Cada vez que uma alteração é feita, o sistema armazena um novo ***snapshot*** do **estado** dos ficheiros em determinado ponto no **tempo**. Você pode ver quem fez a mudança, quando foi feita e por quê.

- **Colaboração**

  Em projetos com múltiplos colaboradores, o controlo de versão permite que todos trabalhem simultaneamente, integrando suas mudanças de forma organizada e evitando conflitos.

Características Principais

- **Reversão**

  Se uma mudança causar problemas, você pode facilmente reverter para uma versão anterior do ficheiro, garantindo que o trabalho não seja perdido.

- ***Branches***

  Muitos sistemas de controlo de versão permitem que você crie ***branches*** (ramificações) do código, onde você pode experimentar novas funcionalidades ou correções de bugs sem afetar a versão principal do projeto.

Modelos de Controlo de Versão

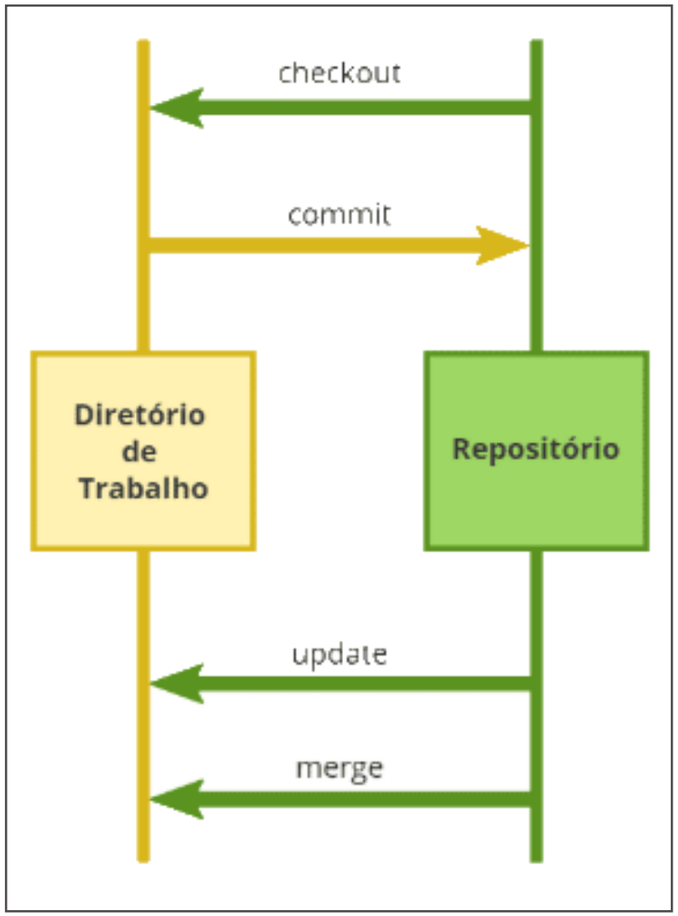
- Existem dois modelos de sistemas de controlo de versão: os **centralizados** (**CVCS**) e os **distribuídos** (**DVCS**).

- Nos **sistemas centralizados** existe um repositório central onde todos os ficheiros e histórico estão armazenados, enquanto nos **sistemas distribuídos**, cada desenvolvedor tem uma cópia completa do repositório, incluindo todo o histórico.

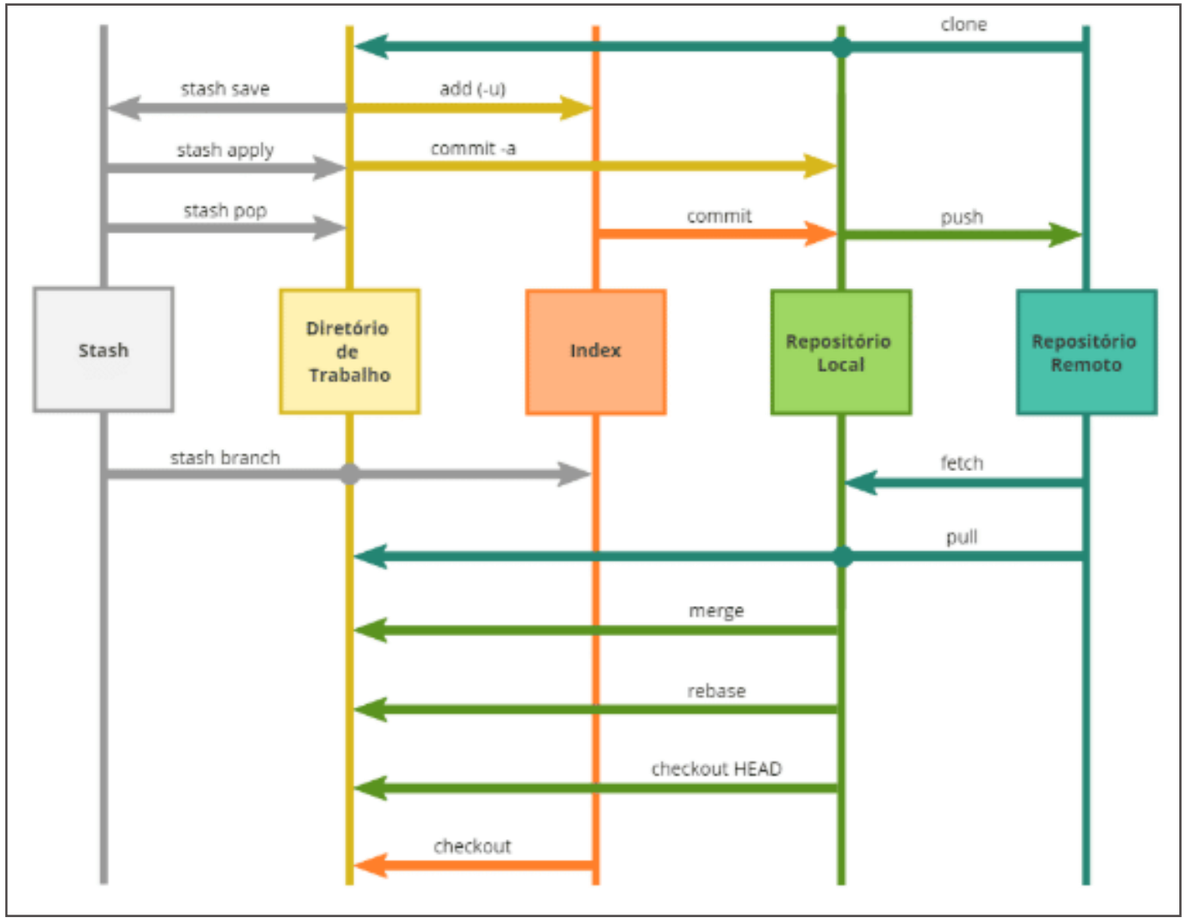
- Atualmente o modelo **centralizado** é implementado pelo **CVS** e pelo **Subversion**.

- Já o modelo **distribuído** é implementado pelo **Git** e pelo **Mercurial**.

Modelos de Controlo de Versão



Centralizado



Distribuído

Sistemas de Controlo de Versão

Sistemas de Controlo de Versão

- **Tipo**: Centralizado
- O **Subversion** é um sistema de controlo de versão que permite o gerenciamento de ficheiros e diretórios ao longo do tempo. Ele armazena todas as versões em um repositório central, onde os desenvolvedores podem fazer ***checkout*** e ***commit*** das suas alterações.
- **Vantagens**: Simplicidade na configuração e uso, ideal para projetos onde um repositório centralizado é preferido.
- **Desvantagens**: Dependência do servidor central para operações, o que pode limitar a colaboração offline e a flexibilidade.

Sistemas de Controlo de Versão

- **Tipo**: Distribuído

- O **Git** é um sistema de controlo de versão que permite que cada desenvolvedor tenha uma cópia completa do repositório, incluindo todo o histórico de mudanças. Ele é especialmente conhecido por sua velocidade e eficiência em operações como ***branching*** e ***merging***.

- **Vantagens**: Flexibilidade, trabalho offline, fácil criação de *branches* e um poderoso sistema de *merge*. É amplamente utilizado na comunidade de desenvolvimento de software.

- **Desvantagens**: A curva de aprendizado pode ser mais acentuada para iniciantes em comparação com sistemas centralizados.

Sistemas de Controlo de Versão

- **Tipo**: Distribuído
- **Descrição**: O Mercurial é semelhante ao Git, permitindo que cada desenvolvedor tenha uma cópia completa do repositório. Ele se destaca por sua simplicidade e interface amigável.
- **Vantagens**: Fácil de usar, bom suporte para projetos grandes e complexos, com forte foco na performance.
- **Desvantagens**: Menos popular que o Git, o que pode resultar em menos recursos e suporte da comunidade.

Plataformas de Hospedagem de Repositórios

- Além da **escolha do tipo de sistema de controlo**, também é possível escolher dentre várias opções, uma **plataforma para hospedagem** dos repositórios.

- A plataforma pode ser hospedada em **servidores próprios**, **locais** ou **remotos**.

- Existem **soluções *open source***, **gratuitas** e **pagas**, voltadas para desenvolvedores independentes, pequenos projetos ou mesmo para grandes empresas.

Plataformas de Hospedagem de Repositórios

Vantagens

- Grande comunidade e ecossistema, facilitando colaborações.
- Suporte robusto para *pull requests* e revisão de código.
- Integrações com diversas ferramentas de CI/CD e serviços de terceiros.
- Recursos como **GitHub Actions** para automação.

Desvantagens

- A versão gratuita tem limitações em repositórios privados (até um certo número de colaboradores).
- Dependência da plataforma online, sem opção de auto hospedagem.

Plataformas de Hospedagem de Repositórios

- **Git** é um sistema de controlo de versão distribuído usado para rastrear **mudanças no código-fonte**.

- Já o **GitHub** é uma plataforma de hospedagem de código que **utiliza Git**, oferecendo funcionalidades adicionais para colaboração e gerenciamento de projetos.

- Os dois são complementares: o **Git** é usado para controlo de versão e o **GitHub** para colaboração e compartilhamento.

Plataformas de Hospedagem de Repositórios

Vantagens

- Funcionalidades completas para todo o ciclo de vida do desenvolvimento (CI/CD, gerenciamento de projetos, etc.).
- Opção de auto-hospedagem disponível.
- Ferramentas integradas para revisão de código e monitoramento.

Desvantagens

- Pode ser mais complexo para novos usuários devido à sua ampla gama de recursos.
- A interface pode ser considerada menos amigável em comparação com o GitHub.

Plataformas de Hospedagem de Repositórios

Bitbucket

Vantagens

- Suporte para Git e Mercurial.
- Integração com ferramentas Atlassian, como Jira e Confluence.
- Controlo granular de permissões e acesso a repositórios.
- Oferece pipelines para CI/CD integrados.

Desvantagens

- Menos popular que o GitHub, resultando em uma comunidade menor.
- Interface pode ser menos intuitiva para novos usuários.

Plataformas de Hospedagem de Repositórios

Vantagens

- Leve e fácil de instalar, ideal para auto hospedagem.
- Interface simples e intuitiva, focada na usabilidade.
- Funcionalidades básicas para gerenciamento de projetos, *issues* e *pull requests*.

Desvantagens

- Menos recursos avançados em comparação com plataformas como **GitLab** e **GitHub**.
- Comunidade menor, o que pode resultar em menos suporte e plugins disponíveis.

Plataformas de Hospedagem de Repositórios

Vantagens

- Histórico longo e confiável na hospedagem de projetos de código aberto.
- Recursos para gerenciamento de projetos e distribuição de software.

Desvantagens

- Popularidade diminuída em relação a outras plataformas modernas.
- Interface pode parecer desatualizada e menos intuitiva.

Plataformas de Hospedagem de Repositórios

AWS CodeCommit

Vantagens

- Totalmente gerenciado e escalável, sem necessidade de manutenção de servidor.
- Integração forte com outros serviços da AWS.
- Segurança robusta com controlo de acesso baseado em IAM.

Desvantagens

- Menos funcionalidades específicas para colaboração em comparação com outras plataformas.
- Custo pode aumentar dependendo do uso, especialmente em grandes equipes.

Plataformas de Hospedagem de Repositórios

Vantagens

- Integração forte com ferramentas **Microsoft** e **Azure**.
- Funcionalidades completas para gerenciamento de projetos e CI/CD.
- Suporte a repositórios Git e TFVC.

Desvantagens

- Curva de aprendizado pode ser íngreme para novos usuários devido à complexidade da plataforma.
- Dependência do ecossistema da Microsoft pode ser um problema para algumas equipes.

Comparação – Uso pessoal vs profissional