Curso DevOps

Aula 04 - Fluxos de Trabalho com Git

Prof. Esp. Guilherme Jorge Aragão da Cruz

guilherme.cruz@alumni.usp.br

linkedin.com/in/guijac

Roteiro

- Compreendendo Conflitos;
- Fluxos de Trabalho com Git;
- Alguns Modelos:
  - Trunk-Based Development;
  - Github Flow;
  - Git Flow;
  - Gitlab Flow;
  - Casos de Uso.
- Referências Bibliográficas.

Compreendendo Conflitos

- Dado um cenário com duas pessoas desenvolvedoras, John e Mary, trabalhando em tarefas separadas de um mesmo repositório;
- Em seu repositório local, John desenvolve normalmente, realizando seus próprios commits.

Fonte: Fluxo de trabalho do Git | Atlassian Git Tutorial

Compreendendo Conflitos

- Da mesma forma, Mary atua em seu repositório local, também realizando commits;
- Notem que ambos trabalhos, até o momento, são em repositórios locais.

Fonte: Fluxo de trabalho do Git | Atlassian Git Tutorial

Compreendendo Conflitos

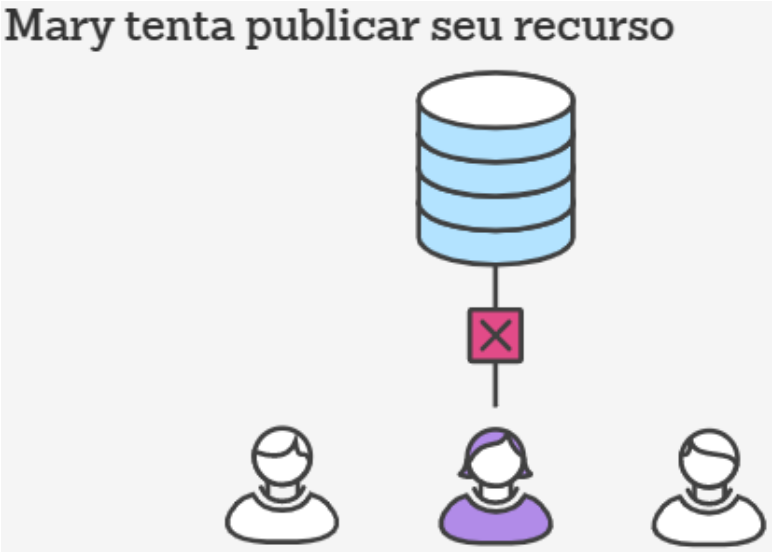
- Após a conclusão de uma tarefa, John efetivamente publica os seus *commits*, através do comando “git push”;
- Até este ponto, nenhum conflito irá ocorrer, pois John “chegou primeiro”.

Fonte: Fluxo de trabalho do Git | Atlassian Git Tutorial

Compreendendo Conflitos

- Por sua vez, notem o que ocorre após Mary concluir a sua tarefa e tentar executar o mesmo comando “git push”:

```
error: failed to push some refs to '/path/to/repo.git'
hint: Updates were rejected because the tip of your current branch is behind
hint: its remote counterpart. Merge the remote changes (e.g. 'git pull') hint: before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

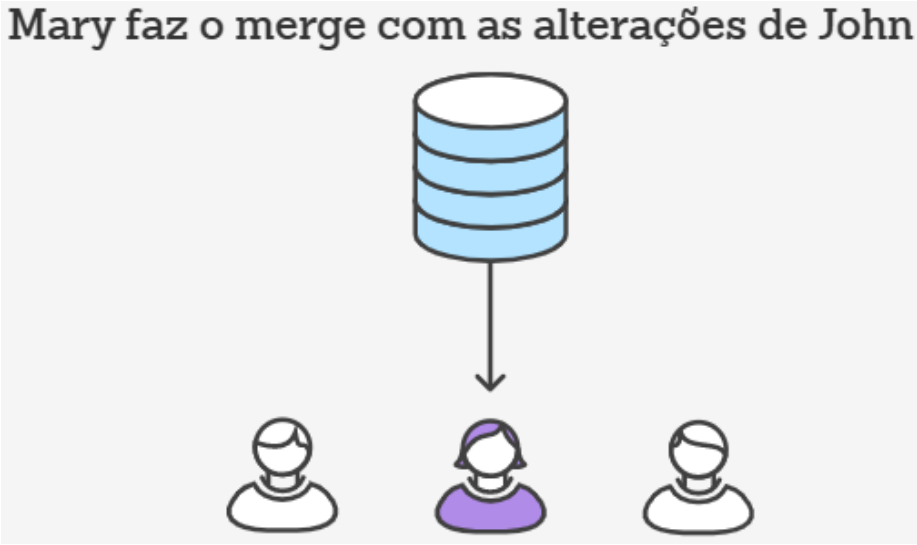



Fonte: Fluxo de trabalho do Git | Atlassian Git Tutorial

Compreendendo Conflitos

- Neste cenário Mary deverá obrigatoriamente resolver os conflitos, através de um *merge*.

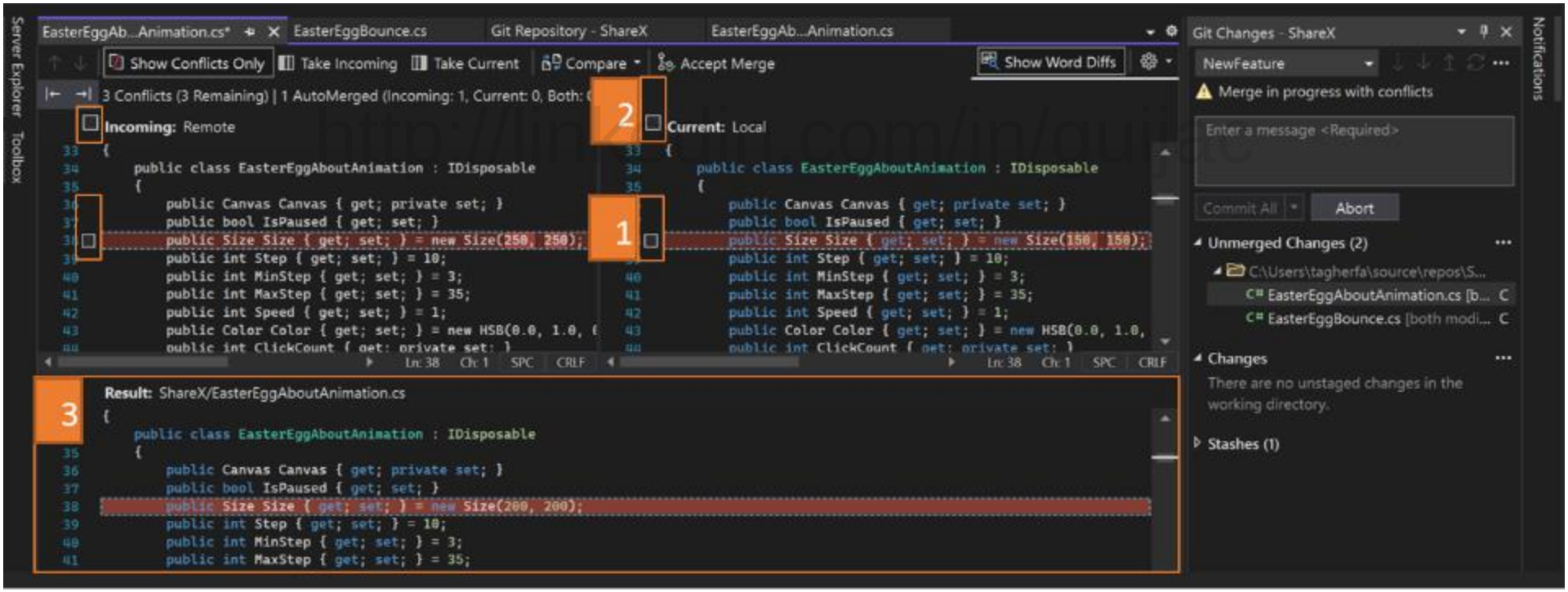
```
git merge main
Auto-merging script2.py
CONFLICT (content): Merge conflict in script2.py
Automatic merge failed; fix conflicts and then commit the result.
```

Fonte: Fluxo de trabalho do Git | Atlassian Git Tutorial

Compreendendo Conflitos

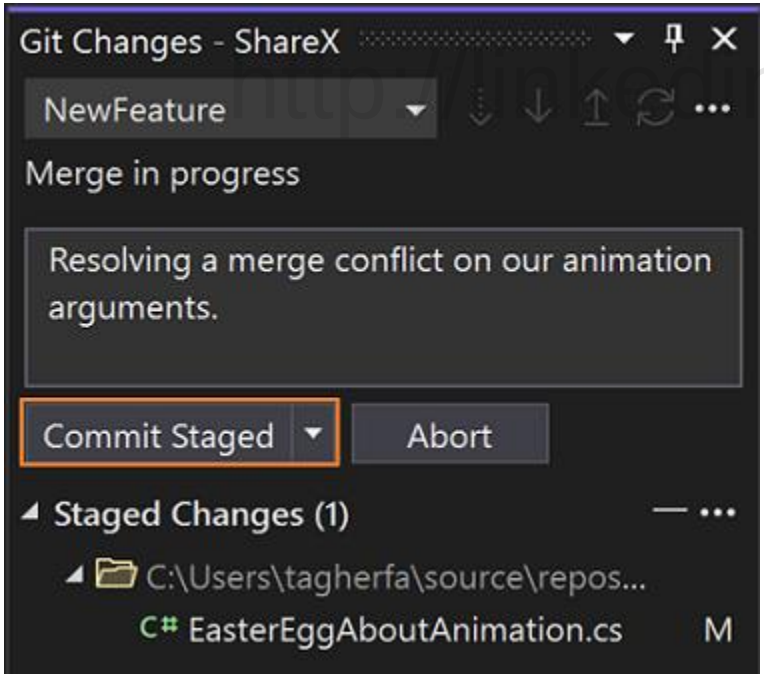
- O Visual Studio Code possui uma interface amigável para resolver conflitos, veremos no laboratório!



Fonte: [Resolve merge conflicts in Visual Studio | Microsoft Learn]

Compreendendo Conflitos

- Após a resolução do conflito, Mary pode enviar as suas alterações, através de um novo *commit*, que pode ser realizado através da interface do VSCode.

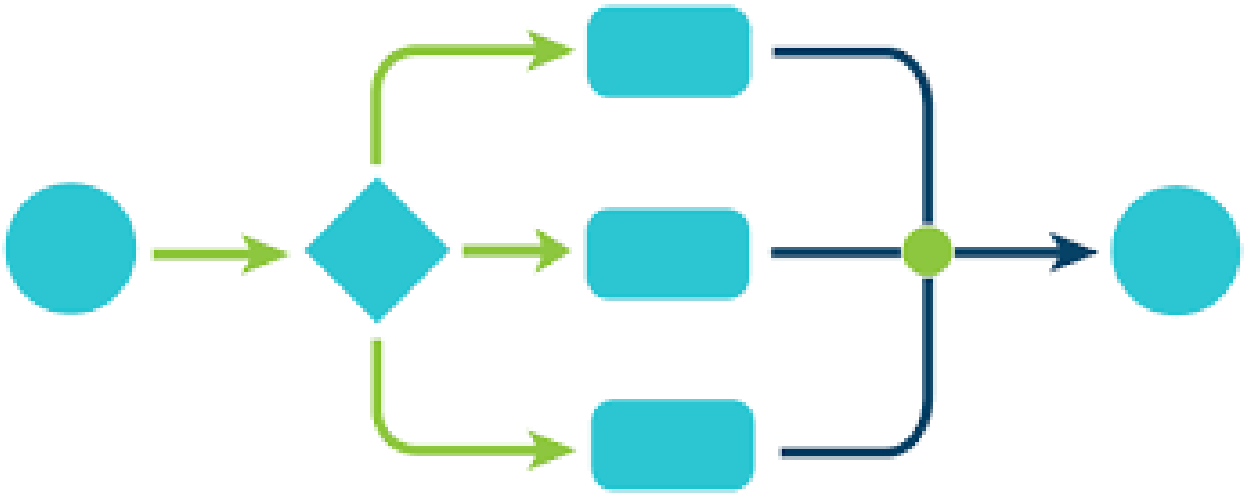


Fonte: Resolve merge conflicts in Visual Studio | Microsoft Learn

Fonte: Fluxo de trabalho do Git | Atlassian Git Tutorial

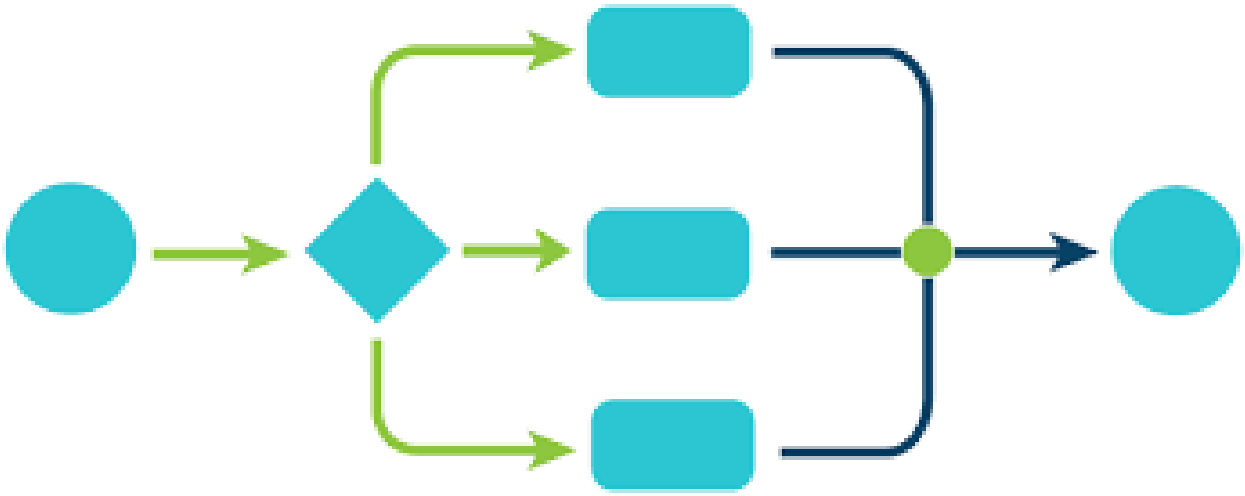
Fluxos de Trabalho com Git

- Dada as possibilidades de conflitos como vimos, temos a introdução do “Git Workflows”, que consistem em **recomendações** de **como utilizar** Git;
- Git é extremamente flexível, possibilitando o **uso** de **diversas formas**, não possuindo um único processo padronizado para seu uso.



Fluxos de Trabalho com Git

- Ao trabalhar em equipes com Git, é importante ter certeza de que **todas as pessoas compreendam** e estejam de acordo sobre o modelo adotado;
- Existem diversos modelos de fluxo de trabalho com Git, podendo, inclusive, haver uma **combinação entre eles**, conforme sua necessidade.



Fluxos de Trabalho com Git

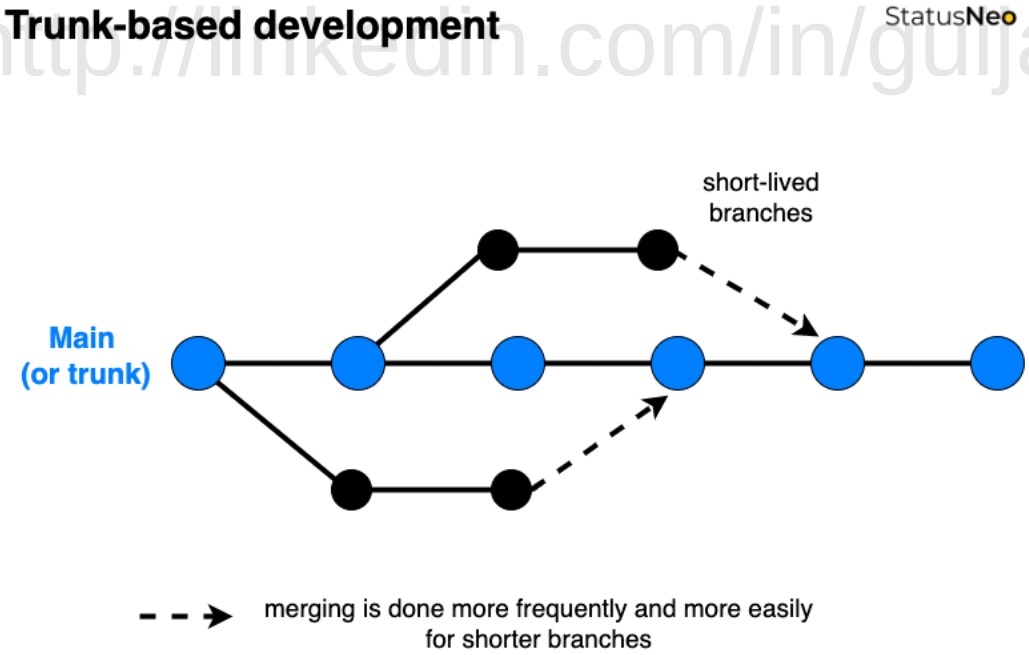
- Ao avaliar um fluxo de trabalho para sua equipe, o mais importante é considerar a **cultura** da equipe:

> *Este fluxo de trabalho tem **escalabilidade** para diferentes tamanhos de equipe?*
> *É fácil de desfazer erros e enganos com este fluxo de trabalho?*
> *Este fluxo de trabalho adiciona alguma **carga cognitiva** desnecessária à equipe?*

ATLASSIAN (2025)

Trunk-Based Development

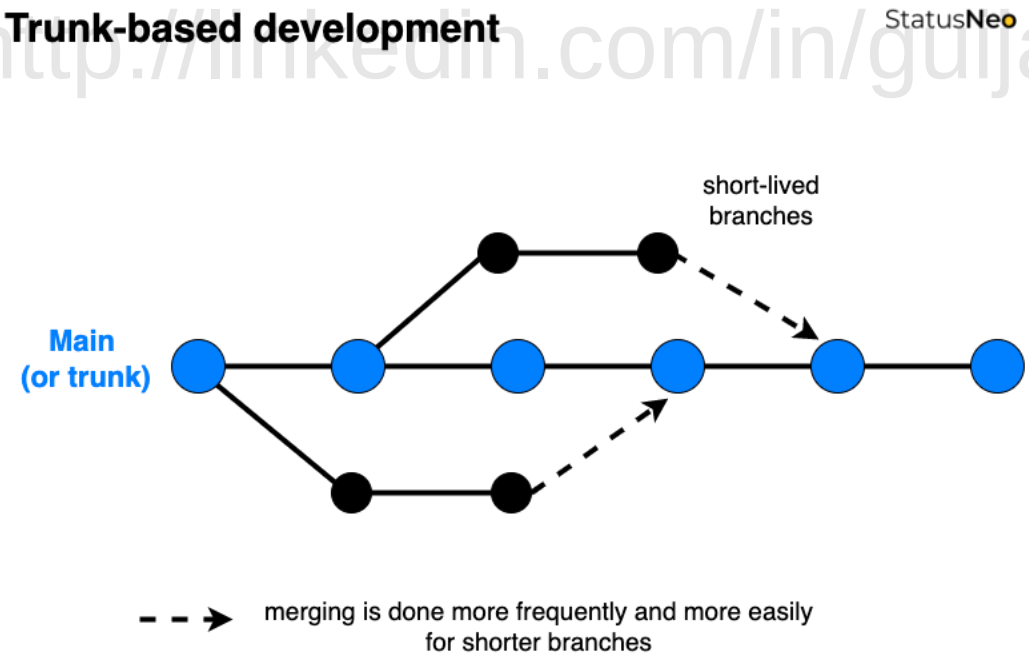
- Estratégia utilizada em equipes com **forte maturidade DevOps**, com integração contínua;
- Exige **alto grau de automação**, mas reduz a complexidade de merges e ocorrência de conflitos.



Fonte: Beginners Guide to Trunk-Based Development (TBD) - StatusNeo

Trunk-Based Development

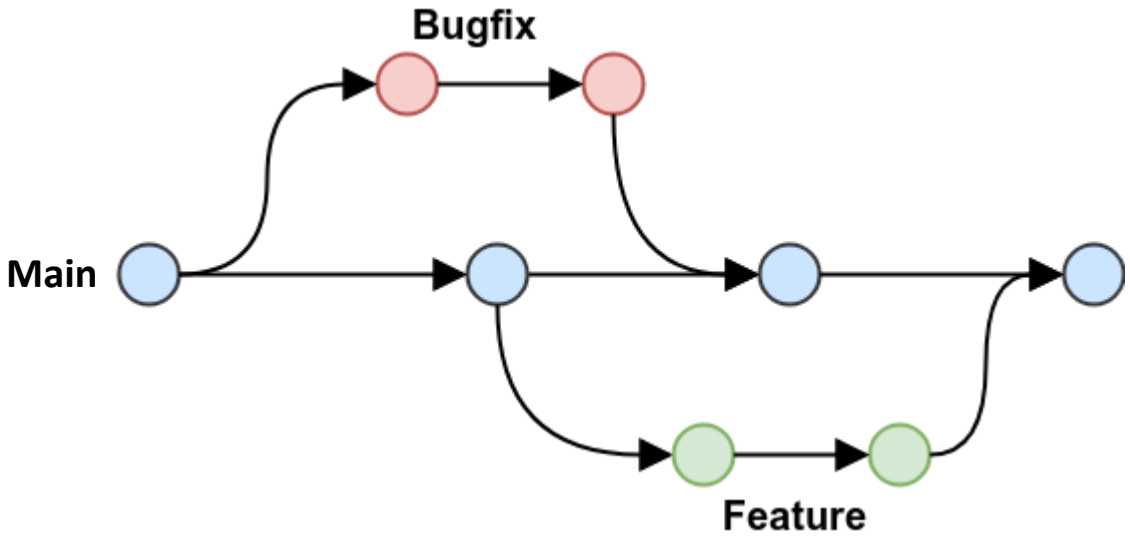
- Desenvolvimento em **pequenos lotes**;
- Revisões **síncronas** de código;
- Testes **automatizados** abrangentes;
- Builds rápidas.



Fonte: Beginners Guide to Trunk-Based Development (TBD) - StatusNeo

Github Flow (2011)

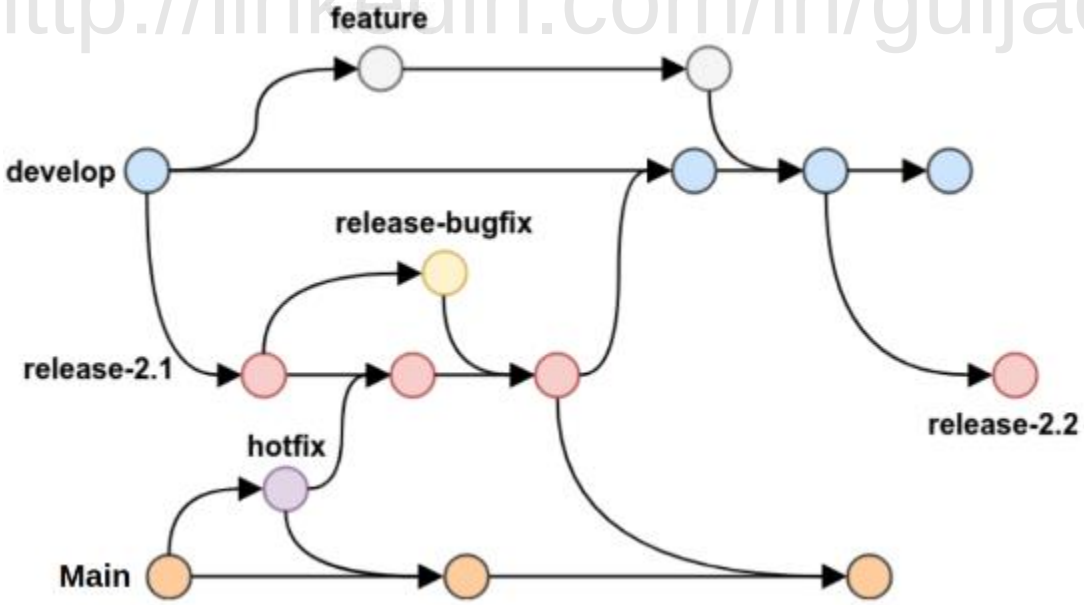
- Modelo **baseado em *branchs***, para implantações regulares, com o uso de duas *branches* de suporte, separadas em correção de *bugs* (*bugfix*) e novas funcionalidades (*feature*).



Fonte: Git Workflows | Programster's Blog

Git Flow (2010)

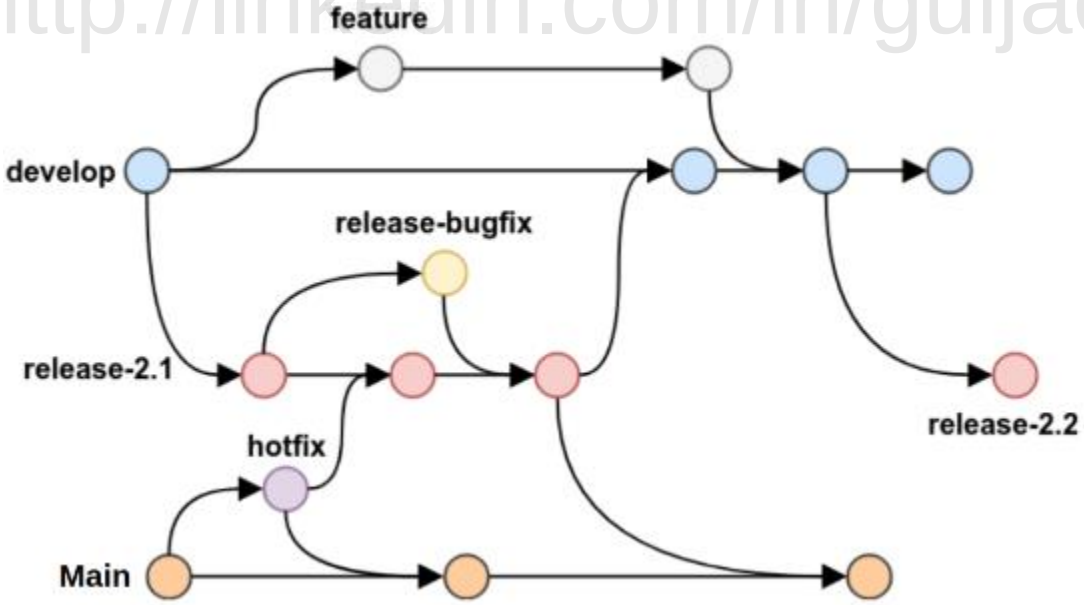
- Modelo **mais rigoroso**, com base em *releases (lançamentos)*, criado pelo engenheiro de software holandês, Vincent Driessen, com duas *branches* fixas:
  - Main: código de produção;
  - Develop: código para a **próxima entrega** em produção.



Fonte: Git Workflows | Programster's Blog

Git Flow (2010)

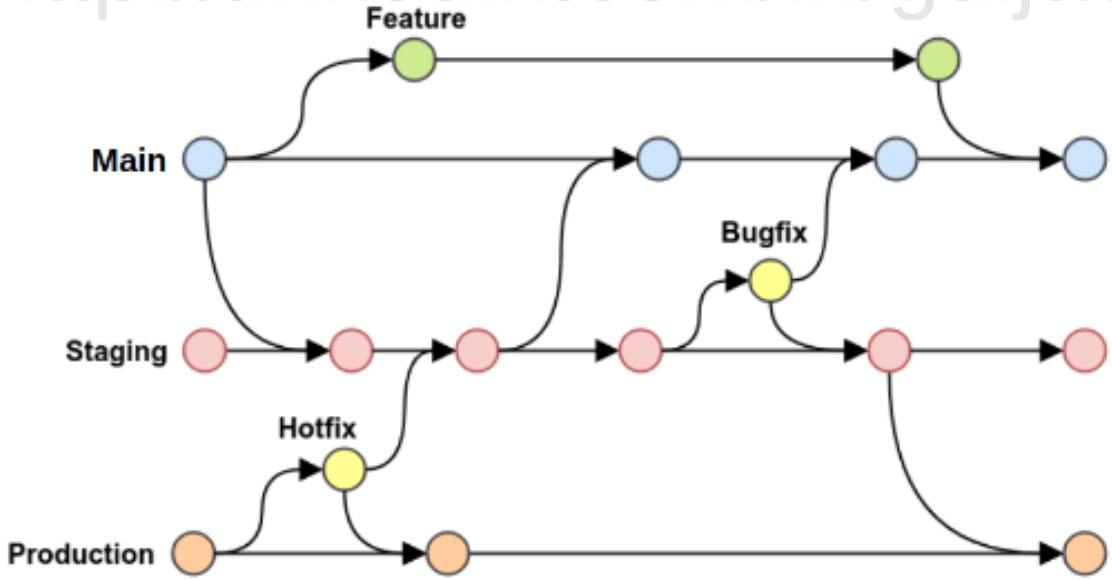
- Existem também quatro *branches* de suporte, sendo uma de *Feature* e as demais:
  - *Release*: Uma ponte para o merge da *Develop* com a Main, sendo utilizada em homologação;
  - *Bugfix*: Correções em tempo de desenvolvimento;
  - *Hotfix*: Correções em tempo de produção.



Fonte: Git Workflows | Programster's Blog

Gitlab Flow (2010)

- Similar ao Git Flow, porém propõe a organização das *branches* conforme o ambiente da aplicação:
  - Main: desenvolvimento;
  - Staging: homologação;
  - Production: produção.



Fonte: Git Workflows | Programster's Blog

Casos de Uso

Tipo de Produto e Método de Release	Tamanho da Equipe	Maturidade da Colaboração	Git Workflow Recomendado
Todos	Pequeno	Alto	Trunk-Based Development (TBD)
Produtos que suportem entrega e lançamento contínuos, como SaaS	Médio	Moderado	Github Flow e TBD
Produtos com uma janela definida para lançamentos ou cadência de versão periódica, como um aplicativos de dispositivos móveis	Médio	Moderado	Git Flow e Gitlab Flow
Produtos com foco na qualidade e que suportem implantação e lançamento contínuos	Médio	Moderado	Gitlab Flow
Produtos com foco na qualidade com longo ciclo de manutenção para versões lançadas	Grande	Moderado	Git Flow

Fonte: Adaptado de [Git Branching Strategies: GitFlow, Github Flow, Trunk Based... (flagship.io)]

Referências Bibliográficas

ANTUNES, Flavio. Git Workflow: o que é e seus principais tipos. Disponível em https://www.zup.com.br/blog/git-workflow. Acesso em 18 jan 2025;

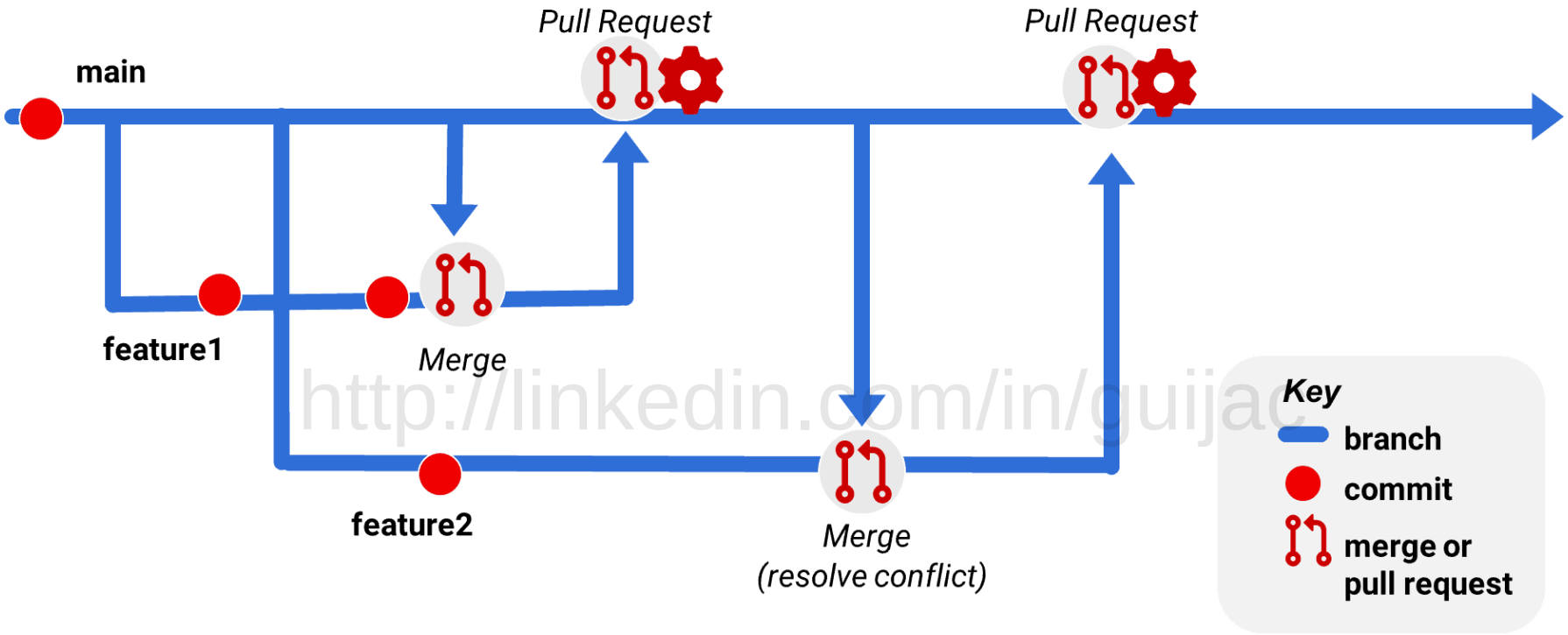
ATLASSIAN. **Comparando fluxos de trabalho do Git: o que você deve saber**. Disponível em https://www.atlassian.com/br/git/tutorials/comparing-workflows. Acesso em 18 jan 2025;

GITLAB. What is a Git workflow? Disponível em https://about.gitlab.com/topics/version-control/what-is-git-workflow/. Acesso em 18 jan 2025;

GOOGLE. **Tecnologia de DevOps: desenvolvimento baseado em troncos**. Disponível em https://cloud.google.com/architecture/devops/devops-tech-trunk-based-development?hl=pt-br. Acesso em 18 jan 2025;

PROGRAMSTER. **Git Workflows**. Disponível em https://blog.programster.org/git-workflows. Acesso em 18 jan 2025;

Por hoje (de teoria!) é só!



Fonte: [Feature branches and pull requests with Git to manage conflicts - Simple Talk (red-gate.com)]

Prof. Esp. Guilherme Jorge Aragão da Cruz

guilherme.cruz@alumni.usp.br

linkedin.com/in/guijac

Licença

- Este conteúdo está licenciado sob a Licença Creative Commons Atribuição-NãoComercial-CompartilhaIgual 4.0 Internacional (CC BY-NC-SA 4.0).
- Todos os direitos autorais sobre este conteúdo pertencem ao autor, e este material não pode ser usado comercialmente sem autorização expressa.
- Para ver o texto completo da licença, acesse o https://creativecommons.org/licenses/by-nc-sa/4.0/legalcode.

Prof. Esp. Guilherme Jorge Aragão da Cruz

guilherme.cruz@alumni.usp.br

linkedin.com/in/guijac