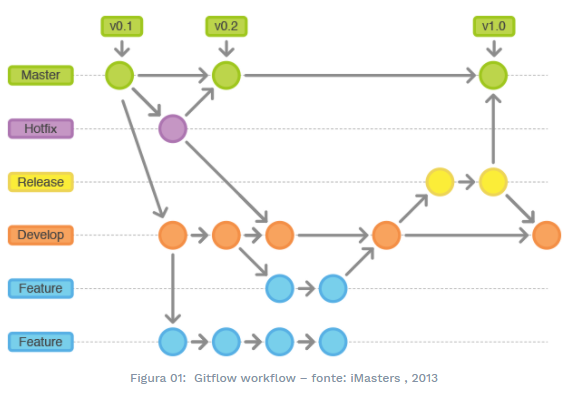
# **GitFlow**

Quando fala-se em desenvolvimento de software, onde tem-se várias equipes atuando sobre o mesmo código, como por exemplo o time responsável por novas “features” ou o time encarregado por corrigir erros, é preciso ter um gerenciamento de código preciso e sem falhas, durante todas as diversas etapas do ciclo de vida do projeto. Com a constante evolução do código, é extremamente necessário termos ferramentas e diretrizes para controlar estas transformações. E uma das maneiras de fazer esse controle é utilizar o GitFlow.

É comum termos diversos times trabalhando de forma concorrente em tarefas conflitantes, que exigem uma boa gestão do código para que nenhum trabalho já feito se perca ou para não haver problemas na consolidação de uma versão final aceitável do código. Estes tipos de conflitos são mais constantes hoje em dia devido a tecnologias tais como microserviços, pipelines de dados, entre outras. O Gitflow, um framework criado para trabalhar em conjunto com a ferramenta de controle de versão Git, promove substancialmente a melhora na organização entre os times, reduzindo drasticamente a ocorrência destes problemas.

Ele se baseia em criações de “branches”, onde cada uma delas possui um papel definido, assim como também a forma que cada uma deve interagir com a outra. Estas “branches” são categoricamente as seguintes:

* **Historic Branches:​** Ao invés de trabalhar apenas com o “branch master”, esse workflow utiliza dois “branches” principais para guardar histórico do projeto. O “branch master” guarda o histórico oficial das entregas, já o “branch develop” serve como integração entre todos os “branches” de funcionalidades (“feature branches”).
* **Feature Branches:​** Cada funcionalidade deve ter seu próprio “branch”, e ele deve ser criado a partir do “develop branch”. Quando uma funcionalidade for concluída, ela é mesclada (merged) novamente com o seu “branch” pai. As features nunca devem interagir diretamente com o “branch master’.
* **Release Branches:​** Quando o “branch develop” estiver com funcionalidades suficientes para uma entrega, cria-se um “branch” de entrega (“release branch”). Com isso, da-se início ao próximo ciclo de entrega, ou seja, nenhuma nova funcionalidade pode ser incluída a partir desse momento. Quando estiver tudo pronto para realizar a entrega, o “release” é mesclada com os “branches” “master” e “develop”.
* **Maintenance Branches:​** Também conhecidos como “hotfix” e/ou “bugfix”. Eles são usados para corrigir rapidamente algum problema em produção. Este é o único “branch” que deve ser criado a partir do “master”. Assim que a correção for finalizada, o “branch” é finalizado e mesclado com o “master” e “develop” “branches”, mantendo assim as linhas completamente atualizadas.



Ao utilizar a abordagem definida pelo gitflow, gera-se grandes benefícios no cliclo de desenvolvimento do projeto como um todo, entre elas:

* **Desenvolvimento em paralelo:** com o gitflow é possível implementar o desenvolvimento em paralelo facilmente, isolando um novo desenvolvimento de trabalhos já finalizados. Novos desenvolvimentos(como novas “features” ou correção de “bugs” não emergenciais) são acoplados dentro da “feature branch” e são inseridos dentro da “branch” principal somente quando estiver tudo testado e finalizado.
* **Colaboração:** “Feature branches” também torna mais fácil para dois ou mais desenvolvedores trabalharem na mesma “feature” porque cada “branch” é um ambiente totalmente apartado, onde as únicas alterações implementadas são as mudanças necessárias para o desenvolvimento do novo recurso.
* **Release staging area:** assim que um novo desenvolvimento estiver completo, ele é mesclado(“merged”) na “branch” de desenvolvimento(“develop branch”), o qual é uma “staging area” para todas as “features” finalizadas que ainda não sofreram “release”.
* **Suporte para correções emergenciais:** gitflow suporta “hotfix branches”, que são “branches” usadas para mudanças emergenciais que conterá apenas a específica correção necessária. Não há risco de acidentalmente se fundir em um novo desenvolvimento ao mesmo tempo.

Em resumo, o GitFlow é um forte aliado das equipes quando se fala em desenvolvimento de código, permitindo manutenções emergenciais, desenvolvimento de novas funcionalidades e rápida implementação sem deixar de lado o controle e versionamento das alterações realizadas. Além disso, permite que a equipe trabalhe paralelamente sem interferir no código que o outro está modificando.