**FACULDADE DE TECNOLOGIA SENAC GOIÁS**

**Auditoria e Qualidade de Software**

LogoSenac

Thiago Pereira dos Santos

**Gerência de Configuração**

Elias Batista Ferreira

GOIÂNIA,

2016

# RESUMO

Este trabalho traz explicações sobre Gerência de Configuração de Software e GitHub, trazendo uma visão ampla sobre o conceito. Exemplificando a necessidade da utilização deste conceito, sugere-se um ambiente de produção que constantemente está gerando uma quantidade de informação grande suficiente para não se poder gerenciá-la, então a metodologia aqui trazida se torna algo de imensa ajuda, a partir deste conceito que mostra que é necessário registrar toda alteração (em todos níveis) para poder garantir a recuperação de dados e redução de retrabalho.

GitHub é uma ferramenta para controle de versão. Disponibiliza ao usuário uma oportunidade de ter um sistema onde é possível controlar toda alteração feita tanto em documentos quanto programas de computadores.

Palavras-chave: Gerência de Software. Software configuration management. Git, GitHub.

# SUMÁRIO

[RESUMO 2](#_Toc447466676)

[1 SUMÁRIO 3](#_Toc447466677)

[2 INTRODUÇÃO 5](#_Toc447466678)

[3 OBJETIVO 5](#_Toc447466679)

[4 GERÊNCIA DE CONFIGURAÇÃO 6](#_Toc447466680)

[4.1 O QUE É QUALIDADE DE SOFTWARE? 6](#_Toc447466681)

[5 ITENS DE CONFIGURAÇÃO 7](#_Toc447466682)

[6 MPS.BR E GCO 8](#_Toc447466683)

[6.1 COMO ALCANÇÁ-LOS: 8](#_Toc447466684)

[6.1.1 GCO 1. Um Sistema de Gerência de Configuração é estabelecido e mantido; 8](#_Toc447466685)

[6.1.2 GCO 2. Os itens de configuração são identificados com base em critérios estabelecidos; 9](#_Toc447466686)

[6.1.3 GCO3 - Os itens de configuração sujeitos a um controle formal são colocados sob baseline 9](#_Toc447466687)

[6.1.4 GCO4 - A situação dos itens de configuração e das baselines é registrada ao longo do tempo e disponibilizada 10](#_Toc447466688)

[6.1.5 GCO5 - Modificações em itens de configuração são controladas 10](#_Toc447466689)

[6.1.6 GCO6 - O armazenamento, o manuseio e a liberação de itens de configuração e baselines são controlados 10](#_Toc447466690)

[6.1.7 GCO7 - Auditorias de configuração são realizadas objetivamente para assegurar que as baselines e os itens de configuração estejam íntegros, completos e consistentes 11](#_Toc447466691)

[7 GIT HUB 11](#_Toc447466692)

[7.1 O QUE É GIT? 12](#_Toc447466693)

[7.1.1 FERRAMENTAS DE VERSIONAMENTO DE SOFTWARE 12](#_Toc447466694)

[7.1.2 SUBSVERSION 12](#_Toc447466695)

[7.1.3 TRAC 13](#_Toc447466696)

[8 INSTALAÇÃO GITHUB 14](#_Toc447466697)

[8.1 DOWNLOAD 15](#_Toc447466698)

[8.2 CLONANDO UM REPOSITÓRIO 16](#_Toc447466699)

[8.3 CRIANDO UM REPOSITÓRIO LOCAL 16](#_Toc447466700)

[8.4 PRIMEIRO COMMIT 18](#_Toc447466701)

[9 PUBLICANDO NOSSO REPOSITÓRIO NO GITHUB 19](#_Toc447466702)

[10 CONCLUSÃO 20](#_Toc447466703)

[11 REFERÊNCIAS BIBLIOGRÁFICAS 20](#_Toc447466704)

[11.1 LINKS 20](#_Toc447466705)

# INTRODUÇÃO

Como a produção de software é algo que está em sempre mudando devido ao fato de que a todo momento um projeto de software pode sofrer alterações, desde a mudança nos requisitos até quanto à correção de defeitos durante seu desenvolvimento, a Gerência de Software é algo que ajuda toda a equipe responsável a não perder as informações, documentos gerados no decorrer do projeto uma forma de oportunizar estes registros de mudanças entra a ação da aplicação GitHub.

# OBJETIVO

Este documento tem como objetivo mostrar de forma clara o que é Gerência de Software e uma breve apresentação sobre a ferramenta GitHub e como utilizá-la.

# GERÊNCIA DE CONFIGURAÇÃO

Como já visto, independentemente do onde você estiver durante o ciclo de vida de um software, o sistema irá mudar e, o desejo de mudá-lo vai persistir durante todo o seu desenvolvimento. Uma nova funcionalidade, a necessidade de adaptação a alguma norma padrão (ISO), a modificação de algum artefato implica muito na qualidade do software.

Para discutir sobre o que é Gerência de Configuração é indispensável o entendimento do conceito de Qualidade de Software.

## O QUE É QUALIDADE DE SOFTWARE?

Atualmente o mercado traz uma alta competitividade entre as organizações na disputa por espaço de mercado e/ou clientes. Com este cenário a busca pela melhoria dos processos, tanto organizacionais quanto de produção tornam-se algo de muito estudo e otimização. O Japão foi um precursor no quesito qualidade com o Controle da Qualidade Total (Total Quality Control – TCQ), que foi rapidamente reproduzida nos Estados Unidos da América com o conceito Gerenciamento da Qualidade Total (Total Quality Management), ambos largamente usados pelo mundo.

Quanto ao conceito de informática, surge em 1906 a IEC (International Electrotechnical Commission) Comissão Eletrotécnica Internacional. Alguns dos seus padrões são desenvolvidos juntamente com a Organização Internacional para Padronização. Com a criação da ISO (International Organization for Standardization) Organização Internacional de Normalização a necessidade de definição de padronização de processos industriais tanto aos serviços quanto a gerenciamento foi formalizado o que contribuiu muito com a evolução deste setor após sua implementação.

A ISO/IEC 12207 é a norma ISO/IEC que define processo de desenvolvimento de software. A norma internacional ISO/IEC 12207 tem como objetivo principal estabelecer uma estrutura comum para os processos de ciclo de vida e de desenvolvimento de softwares visando ajudar as organizações a compreenderem todos os componentes presentes na aquisição e fornecimento de software e, assim, conseguirem firmar contratos e executarem projetos de forma mais eficaz.

O conceito de qualidade já é conhecido por todos. Quanto a software, é um conjunto de características a serem satisfeitas em um determinado grau de modo que o software satisfaça às necessidades de seus usuários.

A noção de qualidade de software pode ser descrita por um grupo de fatores, requisitos ou atributos, tais como: confiabilidade, eficiência, facilidade de uso, modularidade, legibilidade, etc. Podemos classificar estes fatores em dois tipos: externos e internos.

Baseando-se neste contexto de qualidade entra então a Gerência de Configuração. Sob o aspecto de constantes mudanças a Gerência de Configuração de Software vem definir critérios quem permitem manter a consistência e a integridade do software com as especificações.

Para manter ou até aumentar a qualidade dos projetos e softwares cabe então como vimos, um projeto deve seguir as normalizações cumprindo com estas normas de padronização exigidas e, como já visto, remetemos àquela situação citado no primeiro momento: a geração de informações a todo instante, aí entra a necessidade de se manter registrado toda e qualquer alteração durante o decorrer do projeto, tanto no produto quanto às necessidades de adaptação do ambiente de desenvolvimento.

Em um ambiente de desenvolvimento deve-se manter controlados os artefatos que incluem documentação, os modelos do próprio código (fonte e executáveis). Isto vale tanto para projetos pequenos, pois um mínimo controle de versionamento reduz gravemente o desperdício de trabalho.

Com o controle de versão, mantém-se as versões mais antigas dos artefatos que, por sua vez podem ser necessários sua reutilização e, evitando assim que as versões mais antigas tomem conta das versões atuais do projeto, o que é comum em projetos individuais. Partindo deste ponto entende-se que a GCS (Gerência de Configuração de Software) não proíbe alterações, ele permite que elas ocorram de forma controlada sempre que necessário.

# ITENS DE CONFIGURAÇÃO

“Cada um dos elementos de informação que são criados durante o desenvolvimento de um produto de software, ou que para este desenvolvimento sejam necessários, que são identificados de maneira única e cuja evolução é passível de rastreamento” (Pressman em [PRE 92]).

Pressman define também que o conjunto de toda a informação produzida como parte de um processo é chamado de configuração de software enquanto que cada item de informação considerado isoladamente é chamado de item de configuração de software.

Já Sommerville trata como itens de configuração de software todos os documentos que podem ser úteis para a futura manutenção do sistema. Grande parte dos itens de configuração assume a forma de documentos: cronogramas, especificações, lista de requisitos, etc.

Todas estas definições dizem que então, toda informação gerada em particularidade, é um item de configuração: código-fonte, programas, dados de testes entre outros.

O documento de Especificação do Sistema atinge o documento de Plano do Projeto de Software, que por sua vez abrange o documento de Especificação de Requisitos de Software e, assim formam uma hierarquia de informações que aumentam a cada fase do ciclo de vida do software, a cada etapa de sua evolução com criação de novos artefatos que sempre serão gerados até a concepção do software.

Qualquer Item de Configuração constitui uma funcionalidade dentro do ciclo de vida que possuem GCS próprios. Todo sistema em desenvolvimento deve se subdividir em IC’s e seu desenvolvimento deve ser visto como desenvolvimento de cada Item de Configuração particularmente e, assim sucessivamente, são divididos em outros IC’s até que chegue a não poder mais dividi-los, e então trabalhados em um ciclo de vida propriamente definido.

Com esse caos de dados gerados é comum a necessidade de organização para manutenção do software em seu decorrer daí entra algumas ferramentas para controle de versionamento e, gerência de configuração; a ferramenta GITHUB.

# MPS.BR E GCO

O CMMI (Capability Maturity Model Integration) e o MPS.BR (Melhoria do Processo de Software Brasileiro) são metodologias voltadas às áreas de desenvolvimento de software. O MPS.BR é baseado no CMMI porém adaptado à realidade do mercado brasileiro, traz guias para conseguir manter uma estrutura para a melhoria dos processos de desenvolvimento e manutenção de software para as empresas brasileiras.

No Guia Geral MPS de Software está citado e explicado o GCO (Processo de Gerência de Configuração) que tem por objetivo, segundo o próprio guia “O propósito do processo Gerência de Configuração é estabelecer e manter a integridade de todos os produtos de trabalho de um processo ou projeto e disponibilizá-los a todos os envolvidos”

Este processo aparece inicialmente no nível F – Gerenciado, que exige que a organização atinja alguns resultados esperados, são eles:

* **GCO 1**. Um Sistema de Gerência de Configuração é estabelecido e mantido;
* **GCO 2**. Os itens de configuração são identificados com base em critérios estabelecidos;
* **GCO 3**. Os itens de configuração sujeitos a um controle formal são colocados sob baseline;
* **GCO 4**. A situação dos itens de configuração e das baselines é registrada ao longo do tempo e disponibilizada;
* **GCO 5**. Modificações em itens de configuração são controladas;
* **GCO 6**. O armazenamento, o manuseio e a liberação de itens de configuração e baselines são controlados;
* **GCO 7**. Auditorias de configuração são realizadas objetivamente para assegurar que as baselines e os itens de configuração estejam íntegros, completos e consistentes.

## COMO ALCANÇÁ-LOS:

### GCO 1. Um Sistema de Gerência de Configuração é estabelecido e mantido;

O primeiro resultado esperado exige que um sistema de gerência de configuração seja implantado na organização. Para que seja implantado esse sistema é necessário à implementação de três subsistemas:

1. Controle de versão;
2. Controle de modificação;
3. Gerenciamento de construção;

O controle de versão é responsável em sistematizar os versionamentos dos produtos de trabalho de forma segura e controlada, assegurando que versões anteriores possam ser resgatadas.

No que diz respeito à segurança, pode-se definir políticas de controle de acesso, enquanto que políticas de controle de concorrência podem ser implementados para garantir uma maior organização e controle nos produtos de trabalho (SOFTEX, 2012). O controle de modificação, gerência a evolução de problemas identificados nos produtos de trabalho desde sua solicitação até a sua conclusão. Por fim, a função do gerenciamento de construção é transformar itens de configuração em produtos para o usuário final, um exemplo disso é tornar código fonte em um executável (SOFTEX, 2012).

Um ponto importante o qual deve ser observado na implantação de um sistema de gerência de configuração é a possibilidade de implementação pela organização sem o auxílio de ferramentas automatizadas (SOFTEX, 2012). Porém, o auxílio de uma ferramenta torna-se indispensável, pois existem operações difíceis de gerenciar apenas com documentação, como regaste de históricos e versões anteriores, evoluções de versões paralelas de um dado projeto, controle sobre o ciclo de vida de solicitações de mudanças, mapeamento entre itens de configuração e baselines.

O segundo resultado esperado,

### GCO 2. Os itens de configuração são identificados com base em critérios estabelecidos;

GCO2, necessita que a organização estabeleça critérios para definir o que será um item de configuração (IC). Esses critérios, geralmente, encontram- se no Plano de Gerência de Configuração (SOFTEX, 2012).

Com os critérios definidos, a identificação de IC envolve (1) avaliar os produtos de trabalho de acordo com os critérios estabelecidos que definam os itens de configuração; (2) atribuir identificadores únicos aos itens selecionados, isto é, estabelecer um padrão de nomenclatura para os itens de configuração; e (3) listar os itens de configuração detalhando suas características (atributos como nome, conteúdo, responsáveis, autores, localização no repositório, nível de controle desejado).

### GCO3 - Os itens de configuração sujeitos a um controle formal são colocados sob baseline

No GCO3, pede-se que em cada ponto do ciclo de vida de um software (geralmente os marcos) seja selecionado um grupo itens de configuração definidos pelo GCO2, e a partir deles gerar uma baseline. Para definir uma baseline é necessário um controle formal sobre cada item de configuração selecionado, além disso, os critérios para a aprovação desta estarão contidos dentro do Plano de Gerência de Configuração (SOFTEX, 2012).

O Guia de Implementação (SOFTEX, 2010) sugere que pode ser utilizado o mecanismo de criação de rótulos (tags), presente em ferramentas de controle de versões, sobre uma configuração de versões de IC, o qual é suficiente para implementar o conceito de baselines.

Conforme os itens de configuração evoluem ao longo do projeto e baselines são definidas, surge a necessidade de identificar, diferenciar e recuperar o conteúdo de itens de configuração em diferentes etapas do projeto, ou seja, é necessário assegurar que todos os interessados tenham acesso e conhecimento sobre o histórico e situação específica de itens de configuração ou baselines ao longo do tempo. As ferramentas de controle de versão armazenam o histórico das alterações realizadas sobre os itens do repositório, e é através desta funcionalidade das ferramentas que se pode alcançar o GCO 4 (SOFTEX, 2012).

### GCO4 - A situação dos itens de configuração e das baselines é registrada ao longo do tempo e disponibilizada

Os históricos das ferramentas de controle de versão devem fornecer informações suficientes para um mapeamento preciso dos componentes de uma determinada baseline, apresentando a versão específica de cada item de configuração e permitindo a recuperação desta configuração. Com essas informações é possível fazer comparações entre releases, contabilizando o que foi feito ao longo do projeto. Aliado ao sistema de controle de mudanças é possível um controle ainda maior sobre o andamento do projeto.

### GCO5 - Modificações em itens de configuração são controladas

O GCO5 é referente à gestão do ciclo de vida de uma solicitação de mudança, o qual será acompanhado desde a sua solicitação até a conclusão. Porém, antes de uma solicitação de mudança ser efetivamente realizada, é necessário ser avaliada a viabilidade e os impactos que a mudança causará no projeto a partir de um processo formal utilizando critérios estabelecidos na organização. Esses critérios devem estar contidos no Plano de Gerência de Configuração (SOFTEX, 2012).

Segundo o guia de implementação, para que haja um bom controle nas modificações, seria interessante: designar um responsável para efetuar a mudança; registrar todas as mudanças realizadas com suas justificativas; revisões para que as mudanças não causem efeitos indesejáveis; obter uma autorização antes de implementar a modificação a uma nova versão; disponibilizar aos interessados e autorizados (SOFTEX, 2012).

### GCO6 - O armazenamento, o manuseio e a liberação de itens de configuração e baselines são controlados

No sexto resultado esperado, GCO6, espera-se que a organização armazene todos os itens de configuração seguindo as especificações definidas no Plano de Gerência de Configuração. Além disso, deverá ser definido o método de acesso concorrente aos IC do repositório e o sistema de autenticação para acessá-los, levando em conta também a definição de meios de acesso através de canais seguros.

Ainda no GCO6, é necessário estabelecer a liberação de baselines para os interessados, gerando versões de baseline para produção e produtos de trabalho fechados. As baselines de produção são utilizadas pela equipe de desenvolvimento para evolução do produto de forma incremental. Os produtos de trabalho fechados são as baselines que serão enviadas para o cliente, essa versão só será liberada depois da aprovação do processo de auditoria (SOFTEX, 2012).

### GCO7 - Auditorias de configuração são realizadas objetivamente para assegurar que as baselines e os itens de configuração estejam íntegros, completos e consistentes

Por fim, o último resultado esperado, GCO7, refere-se às auditorias de gerência de configuração, que irá avaliar se todos os procedimentos previstos no Plano de Gerência de Configuração estão sendo seguidos, além de verificar a consistência, completitude e integridade dos IC e baselines.

As datas das auditorias são definidas previamente no Plano de Gerência de Configuração e são auditadas por alguém que não está diretamente ligado ao projeto, sendo possível o auxílio de um checklist para aumentar a objetividade da auditoria (SOFTEX, 2012).

Outra auditoria importante é referente à baseline, no qual são efetuados dois tipos de auditorias: a funcional e a física. Na auditoria funcional são revisados documentos de teste, planos, metodologia para verificar se a baseline está correta, ou seja, avalia se ela cumpre o seu propósito, todas as versões dos IC estão de acordo com o previsto. No entanto, a auditoria física verifica a completitude da baseline, avaliando se todos os IC estão presentes (SOFTEX, 2012).

# GIT HUB

O GitHub é um serviço de hospedagem de repositórios Git disponível gratuitamente. Por meio dele você pode compartilhar o desenvolvimento de um projeto com desenvolvedores do mundo todo e ainda contar com alguns componentes de rede social e wiki para interagir, divulgar e documentar seu software.

**Conceitos do Git**

Antes de vermos mais detalhes de como trabalhar com o Git, precisamos conhecer alguns conceitos básicos que são necessários para entender como ele funciona.

**Branch**: ramificação do projeto, cada *branch* representa uma versão do seu projeto e podemos seguir uma linha de desenvolvimento a partir de cada *branch*.

**Clone**: cópia local de todos os arquivos de um repositorio git.

**Commit**: coleção de alterações realizadas, é uma espécie de *checkpoint*, sempre que necessário você pode retroceder até algum *commit* existente.

**Fork**: uma bifurcação do projeto, uma cópia do projeto existente para seguir uma nova direção.

**Master**: branch padrão de um repositório Git.

**Merge**: é a capacidade de incorporar alterações do git, quando acontece uma junção de diferentes *branches*.

**Pull**: puxa as alterações do repositório remoto.

**Push**: empurra as suas alterações para o repositório remoto.

**Repositório**: local onde ficam todos os arquivos do projeto, inclusive o histórico e versões.

## O QUE É GIT?

GIT é um serviço para ajuda no controle de versionamento onde o projeto é compartilhado com pessoas que possam de alguma forma contribuir em seu desenvolvimento.

No cenário de desenvolvimento de software, como discutido nos itens anteriores, gera a todo momento uma quantidade de informações suficientes para que não se possa manter um controle, o conceito de GCS entra na questão para ajudar o condicionamento para a evolução do software durante seu ciclo de vida. Entende que IC (Item de Configuração) é toda informação criada como parte de engenharia de software.

Além dos IC derivados dos artefatos de software, muitas organizações de engenharia de software também colocam ferramentas sob o controle de configuração. As ferramentas de controle de versão de software tanto quanto aos compiladores, browsers e outras ferramentas automáticas intrínsecas a Configuração de Software, estas ferramentas são usadas no decorrer do desenvolvimento do ciclo de vida, e elas geram documentos, códigos-fonte e dados e, portanto, devem estar disponíveis sempre quando requisitadas.

Um repositório GCS é um conjunto de mecanismos e estruturas de dados que permitem a uma equipe de software gerenciar alterações de maneira eficaz. Ele proporciona as funções óbvias de um sistema moderno de gestão de banco de dados garantindo a integridade dos dados, compartilhamento e integração. Além disto, um repositório proporciona um (*hub*) para a integração das ferramentas de software, está no centro do fluxo do processo de software e pode impor estrutura e formato uniformes para os artefatos.

### FERRAMENTAS DE VERSIONAMENTO DE SOFTWARE

### SUBSVERSION

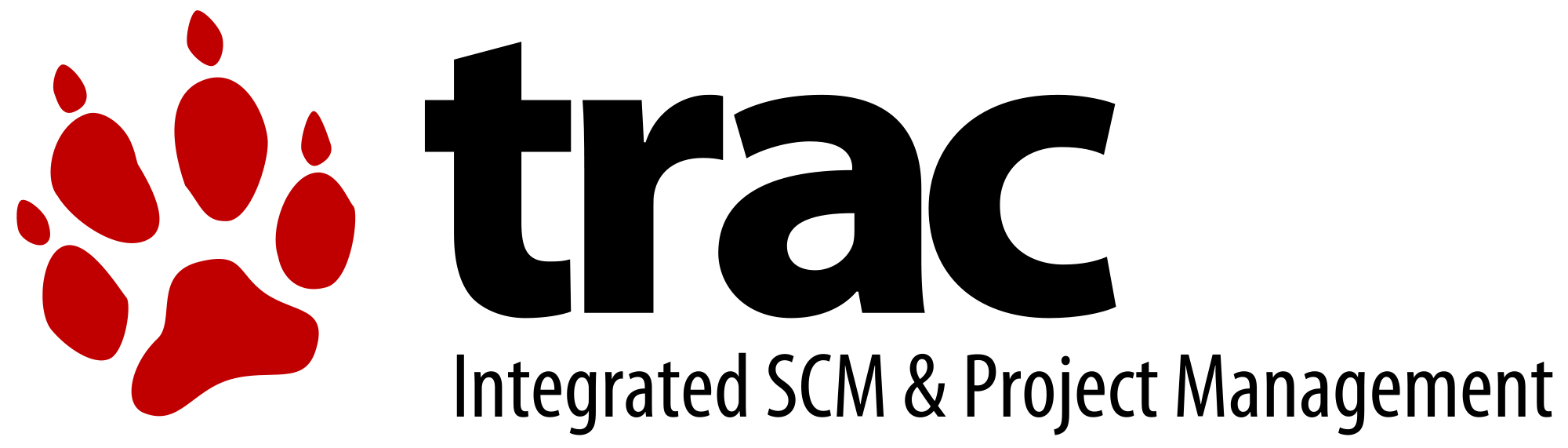
Subversion é um sistema de free / open source de controle de versão (VCS). Isto é, Subversion gerencia arquivos e diretórios, e as mudanças feita a eles, ao longo do tempo. Isso permite que você recupere versões antigas de seus dados ou examinar a história de como seus dados alterados.

A este respeito, muitas pessoas pensam de um sistema de controle de versão como uma espécie de "máquina do tempo".

Subversion pode funcionar em rede, o que permite que ele seja usado por pessoas em diferentes computadores. Em algum nível, a capacidade de várias pessoas para modificar e gerenciar o mesmo conjunto de dados a partir de seus respectivos locais promove a colaboração. O progresso pode ocorrer mais rapidamente, sem um único canal através do qual deve ocorrer todas as modificações. E porque o trabalho está versionado, você não precisa medo de que a qualidade é o trade-off para perder essa via-se alguma mudança incorreta é feita para os dados, apenas desfaça essa mudança.

Alguns sistemas de controle de versão também são sistemas de gerenciamento de configuração de software (SCM). Estes sistemas são especificamente adaptados para gerenciar árvores de código-fonte e têm muitas características que são específicas para desenvolvimento de software-como a compreensão nativamente linguagens de programação, ou fornecer ferramentas para construir software. Subversion, no entanto, não é um destes sistemas. É um general sistema que pode ser usado para gerenciar qualquer coleção de arquivos. Para você, estes arquivos podem ser código-fonte-para outros, qualquer coisa de listas de compras de supermercado a arquivos de edição de vídeo digital e além.

### TRAC



Trac é um sistema de código aberto, gerenciamento de projeto baseado na Web e de rastreamento de bugs. O programa é inspirado CVSTrac, e foi originalmente chamado svn trac devido à sua capacidade de interagir com o Subversion. Ele é desenvolvido e mantido pela Edgewall Software.

Trac é escrito na linguagem de programação Python. Até meados de 2005, ele estava disponível sob a licença GNU General Public. Desde a versão 0.9, foi lançado sob uma licença BSD modificada.

Trac é um sistema wiki e acompanhamento de problemas avançada para projetos de desenvolvimento de software. Trac usa uma abordagem minimalista para a gestão de projetos de software baseado na web. Nossa missão é ajudar os desenvolvedores escrever bons softwares enquanto permanecer fora do caminho. Trac deve impor o mínimo possível no processo de desenvolvimento estabeleceu uma equipe e políticas.

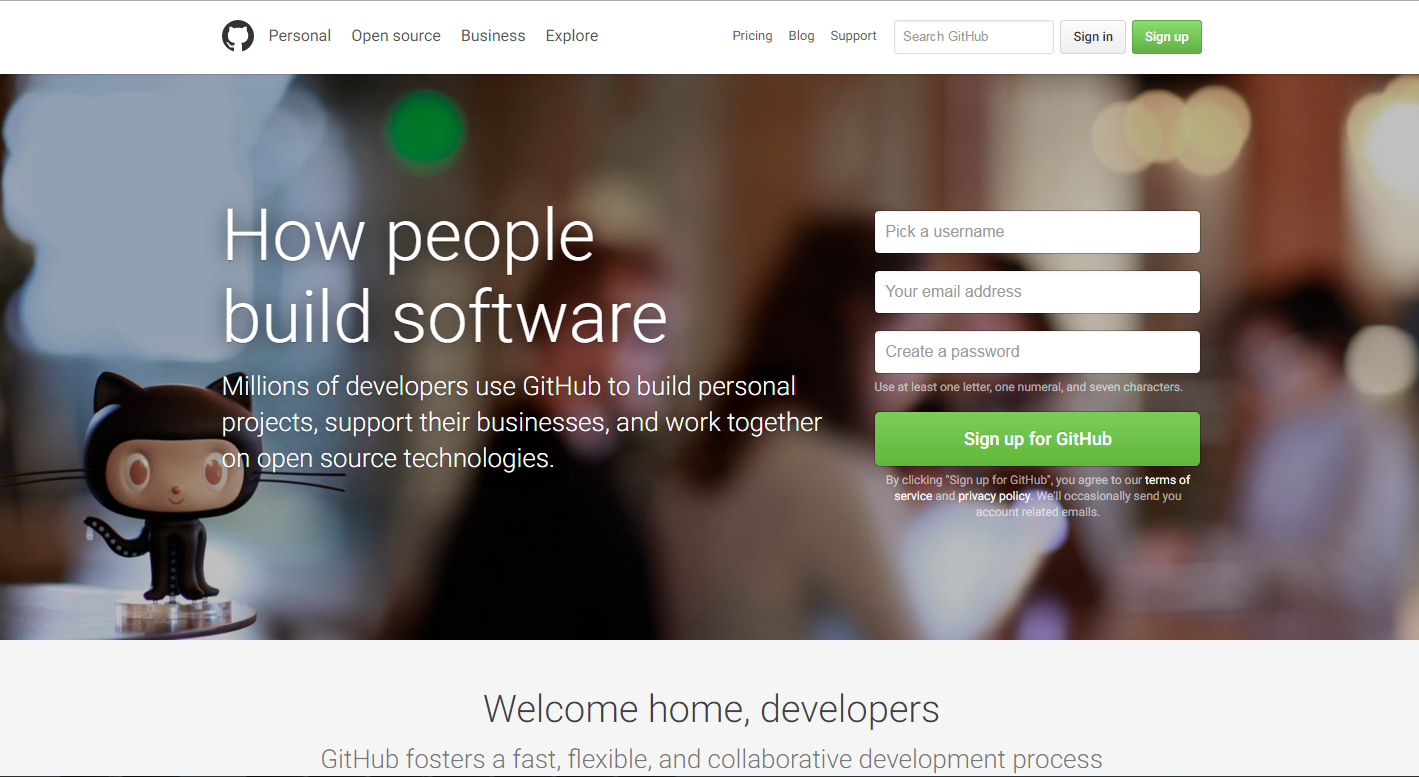
Ele fornece uma interface para (outros sistemas de controle de versão ou) Subversion e Git, um Wiki integrado e recursos de relatórios convenientes.

Trac permite marcação wiki em descrições de emissão e de mensagens de commit, criando links e referências sem costura entre bugs, tarefas, changesets, arquivos e páginas wiki. Uma linha do tempo mostra todos os eventos do projeto atuais e passados em ordem, tornando a aquisição de uma visão geral do projeto e acompanhamento do progresso muito fácil. O roteiro mostra a estrada à frente, listando as próximas etapas.

# INSTALAÇÃO GITHUB

O Git é um programa de controle de versões, onde você trabalha sempre em um mesmo diretório, fazendo alterações em seu projeto, gravando documentação e comentários.

Ele guarda tudo que você faz. Todas as vezes que você salvar e “aprovar” no Git, ele registra. Os registros são para que você possa voltar ao anterior quando acontecer algum problema.

Já entendido o conceito de GIT então seguimos com a ferramenta **GITHUB**.Você pode primeiramente realizar seu cadastro no site <https://github.com/>:

Na página seguinte você poderá escolher entre cinco planos pessoais diferentes. A opção gratuita (*free*) já virá selecionada por padrão, assim confirme sua opção clicando no botão *Chosen* do plano *Free*.

Agora basta clicar no botão Finish sign up para concluir a criação da conta.

Com sua conta criada você já pode começar a utilizar o GitHub. Para ativação da conta é necessária confirmação de seu endereço de e-mail. Acesse a caixa de entrada do e-mail que você informou e procure uma mensagem com o título *[GitHub] Please verify your email address*. Abra este e-mail e clique em *Verify email address*.

## DOWNLOAD

Você pode baixar seu aplicativo pelo link [https://desktop.github.com](https://desktop.github.com/).

Este novo cliente unifica a experiência para Mac (OS X) e PC (Windows). Usuários de Linux, como na versão anterior, terão que recorrer a clientes de terceiros como o [git-cola](http://git-cola.github.io/) ou[SmartGit](http://www.syntevo.com/smartgit).

Após baixar e executar o instalador o GitHub Desktop irá solicitar seu nome de usuário e senha. Informe-os e clique em continue. Na tela seguinte informe o seu e-mail.

## CLONANDO UM REPOSITÓRIO

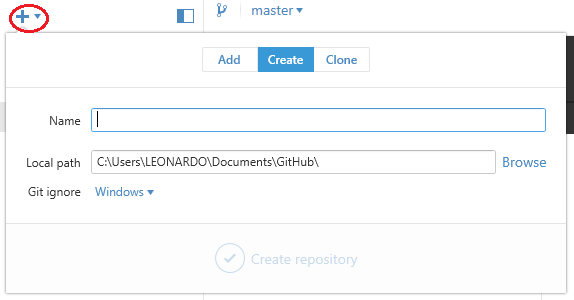
É necessário a clonagem de um repositório, o que será solicitado inicialmente quando usar o programa, ou também este método pode ser feito também na página web.

Para podermos colaborar com um projeto precisamos, antes de mais nada, baixar o código-fonte dele para o nosso computador. E isto é feito por meio da criação de um clone.

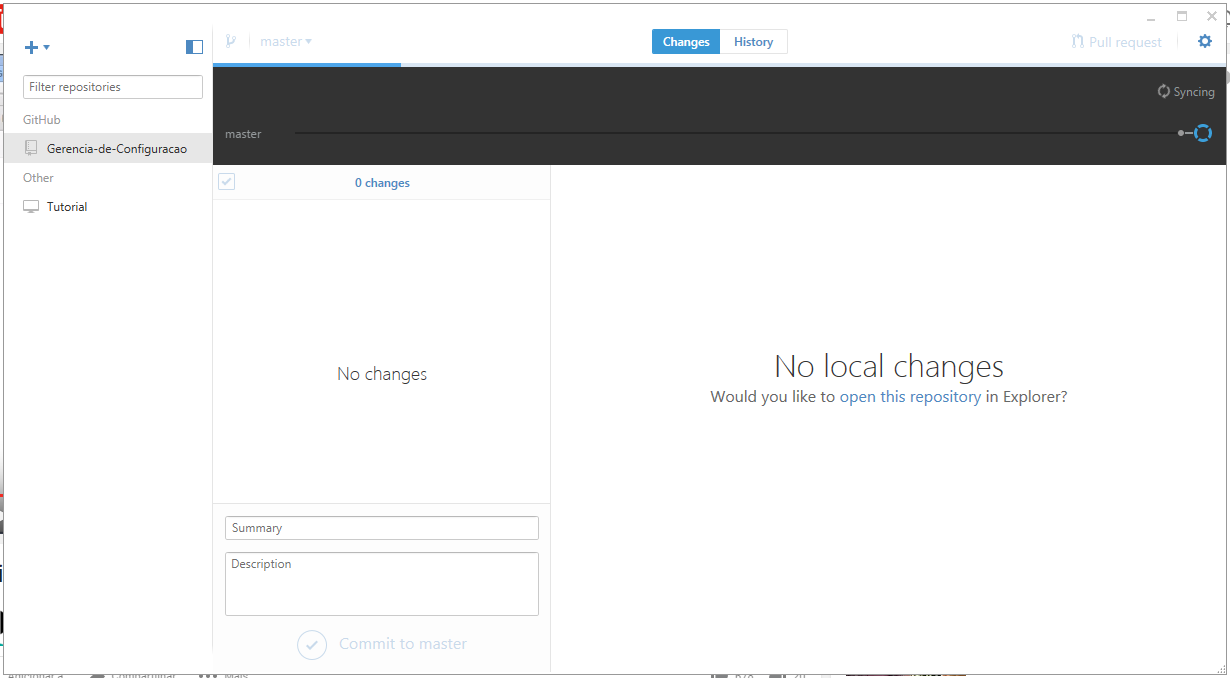
Para isso clique no Botão + para adicionar um repositório remoto. Clique na aba clone e depois no link explore GitHub. Isto irá abrir o site do GitHub no navegador. Logue com seu usuário que nós criamos a pouco caso você não esteja logado.

## CRIANDO UM REPOSITÓRIO LOCAL

Clique no botão + do cliente e depois na aba *create*, infome um nome para o repositóro, verifique o *Local Path* (Caminho) onde será criado o repositório, por padrão o diretório Documentos do seu usuário no OS X e a pasta Documentos/GitHub do seu usuário no Windows, e clique em *Create Repository*.

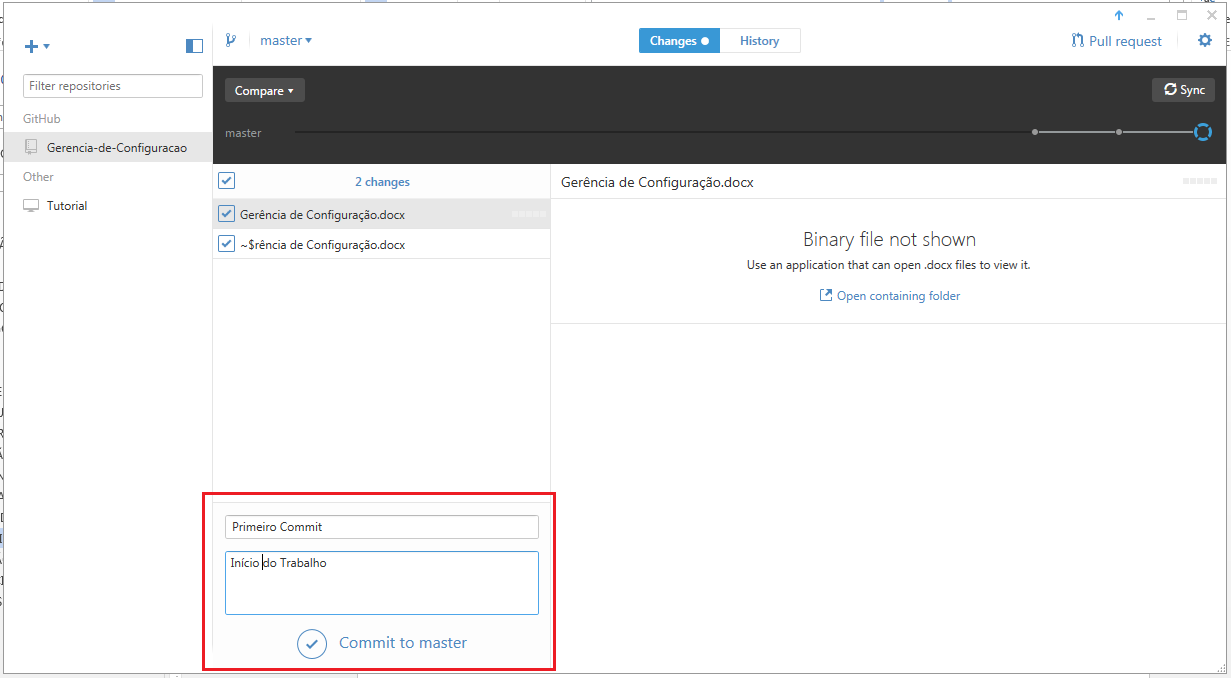


Neste exemplo o nome do diretório será Gerencia de Configuracao. Então criado seu diretório você terá esta tela:

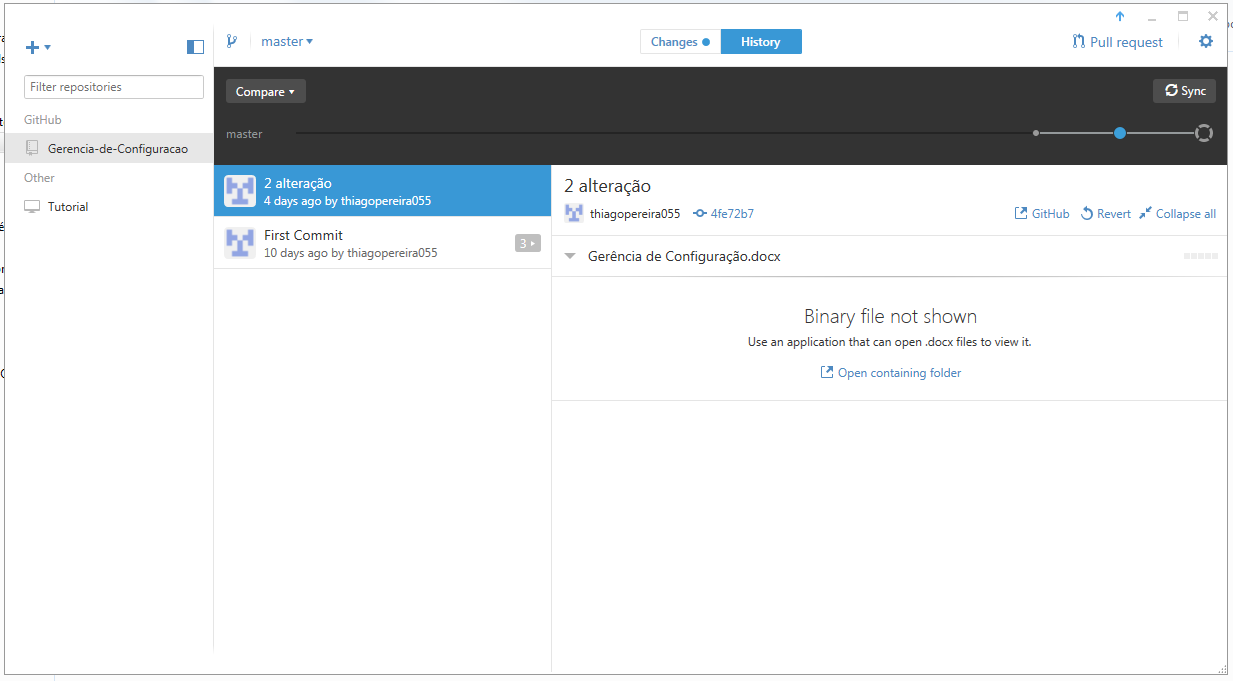


## PRIMEIRO COMMIT

Agora vamos fazer o nosso primeiro *commit*, ou seja, vamos registrar a nossa primeira alteração no repositório. É importante que todo *commit* seja acompanhado de um comentário explicando qual alteração foi feita.

Escreva “Primeiro Commit” em Summary e “Início do Trabalho” em *Description*. Agora clique em *Commit to Master*. 

Pronto! Agora na sua aba *History* já consta o seu primeiro *commit*.

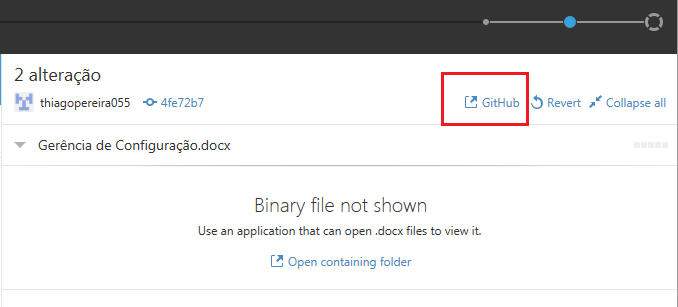


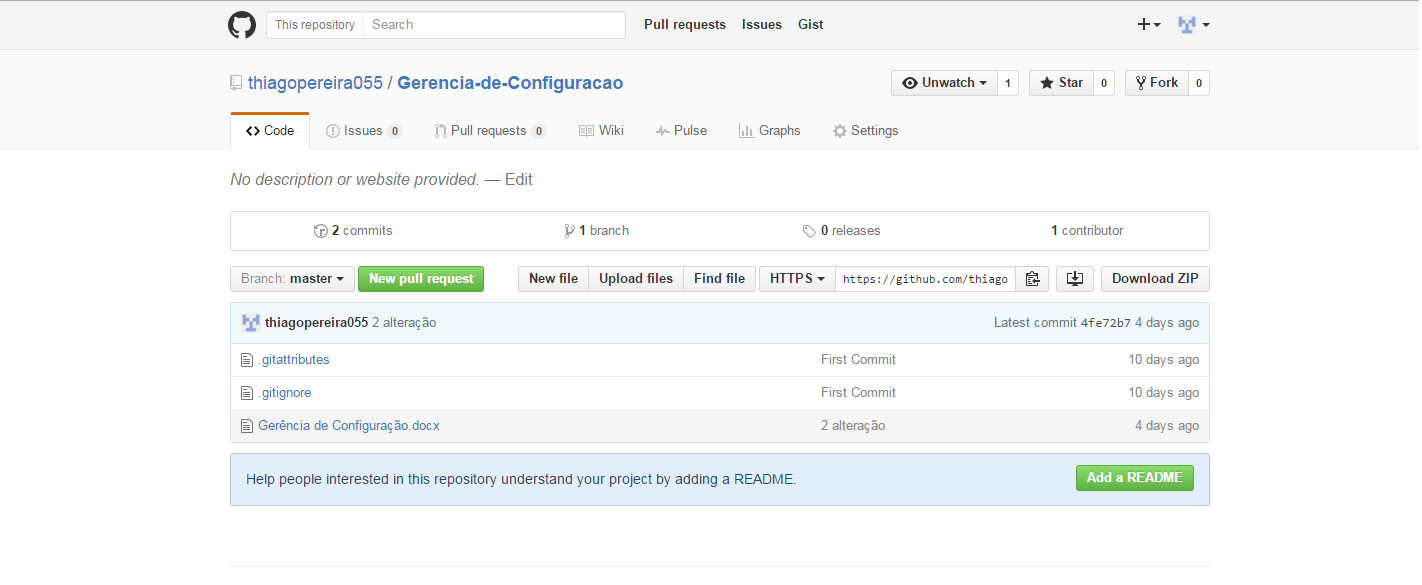
# PUBLICANDO NOSSO REPOSITÓRIO NO GITHUB

E por fim iremos subir nosso repositório criado localmente para o GitHub.

Para isso basta clicar em *Publish* no canto direito do cliente. Na tela que será aberta escreva “Meu primeiro repositório” em *Description* e verifique se a sua conta está selecionada. Agora basta clicar em *Publish Repository*.

Agora você pode visualizar sua publicação no seu diretório selecionando sua publicação e clicando em “GitHub”.





# CONCLUSÃO

Neste trabalho vimos como funciona um ambiente de desenvolvimento de software, a necessidades das organizações a melhorarem seus processos internos a fim de que ganhem mais espaço, o que as obrigam adotar metodologias de gerenciamento e alguns de seus desdobramentos e necessidades. Vimos também que a quantidade de informação gerada durante o desenvolvimento dos softwares é grande e, como consequência requer que sejam subdividas em *itens de configuração* que criam hierarquias que precisam serem organizadas, daí o surgimento de aplicações e metodologias como MPS.BR focadas neste aspecto.

Foram apresentadas 3 ferramentas de gerência de configuração SUBVERSION, TRAC e GITHUB, na qual foi apresentada sua utilização inicial.

# REFERÊNCIAS BIBLIOGRÁFICAS

PRESSMAN, Roger S. **Engenharia de Software Uma Abordagem Profissional.** AMGH Editora Ltda, 2011.

PARREIRAS, F. S., BAX, M. P. A gestão de conteúdos no apoio a engenharia de software. In: **KMBrasil**, 2003, São Paulo. Anais... São Paulo: SBGC - Sociedade Brasileira de Gestão do Conhecimento. 2003. CD-ROM. Disponível em <<http://www.fernando.parreiras.nom.br/publicacoes/pgct142.pdf>>. Acesso em 03 abril.2016.

## LINKS

<http://www.mandrado.com/gerenciamento-do-ciclo-de-vida-de-software/1-gerenciamento-de-configuracao-de-software>

<https://git-scm.com/book/pt-br/v1/Primeiros-passos-No%C3%A7%C3%B5es-B%C3%A1sicas-de-Git>

<http://mobgeek.com.br/blog/tutorial-git-iniciantes>