



Integração Contínua



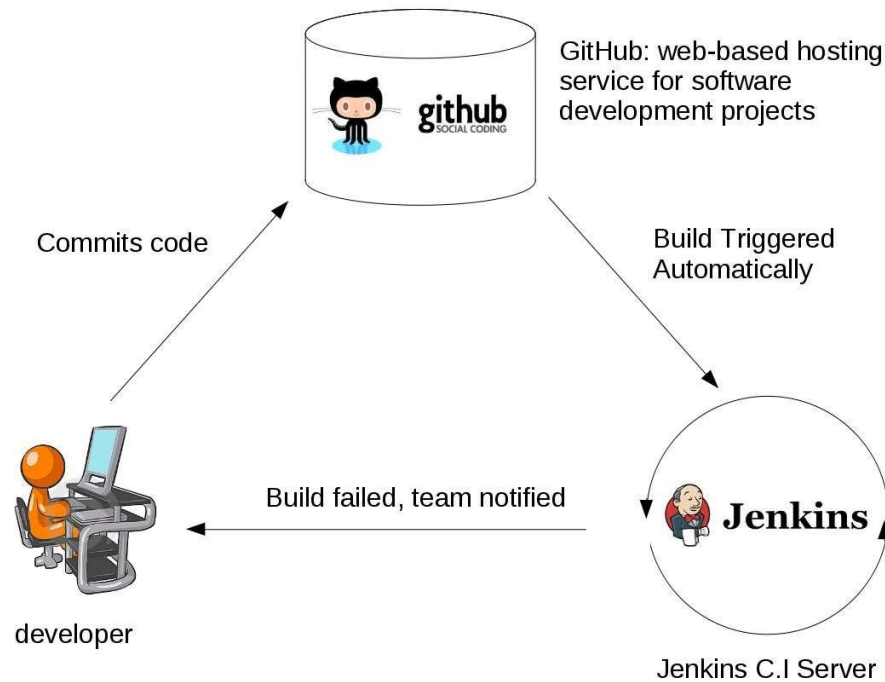
Jenkins

Integração Contínua

- O que é Integração Contínua?
- Por que precisamos disso?
- Diferentes fases de adoção da Integração Contínua



Jenkins



O que é Integração Contínua?

- Os desenvolvedores comitam o código em um repositório compartilhado regularmente.
- O sistema de controle de versão está sendo monitorado. Quando um commit é detectado, uma compilação será acionada automaticamente.
- Se a compilação não estiver ok, os desenvolvedores serão notificados imediatamente.

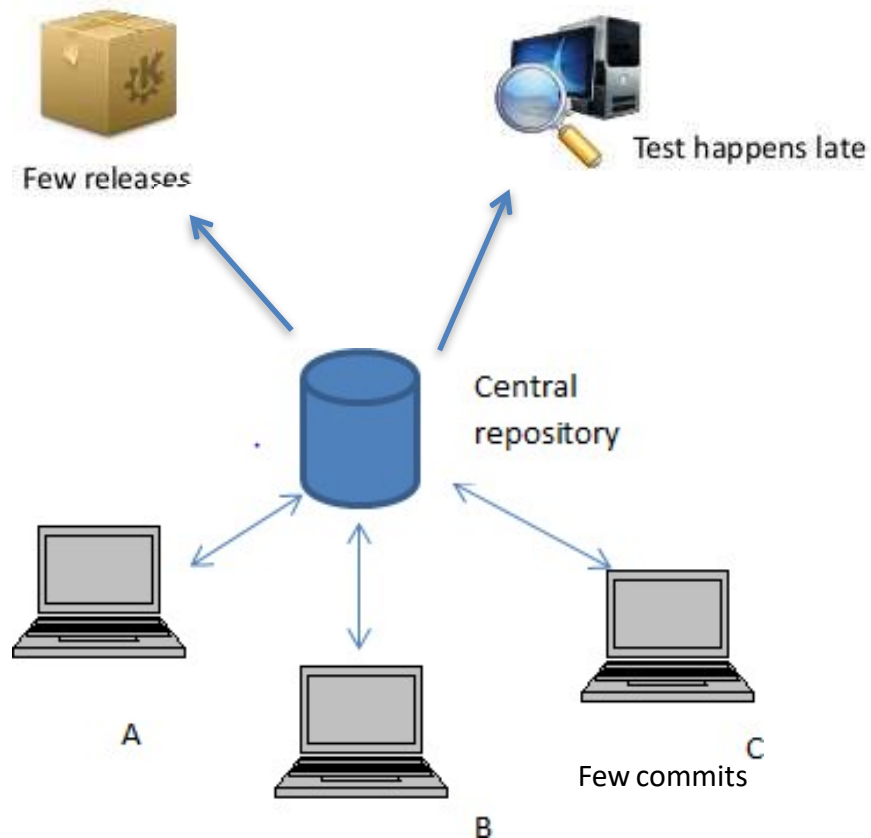
Por que precisamos de Integração Contínua?

- Detectar problemas ou bugs, o mais cedo possível, no ciclo de vida do desenvolvimento.
- Uma vez que toda a base de código é integrada, construída e testada constantemente, os potenciais bugs e erros são pegos no início do ciclo de vida, o que resulta em um software de melhor qualidade.

Diferentes etapas da adoção da Integração Contínua

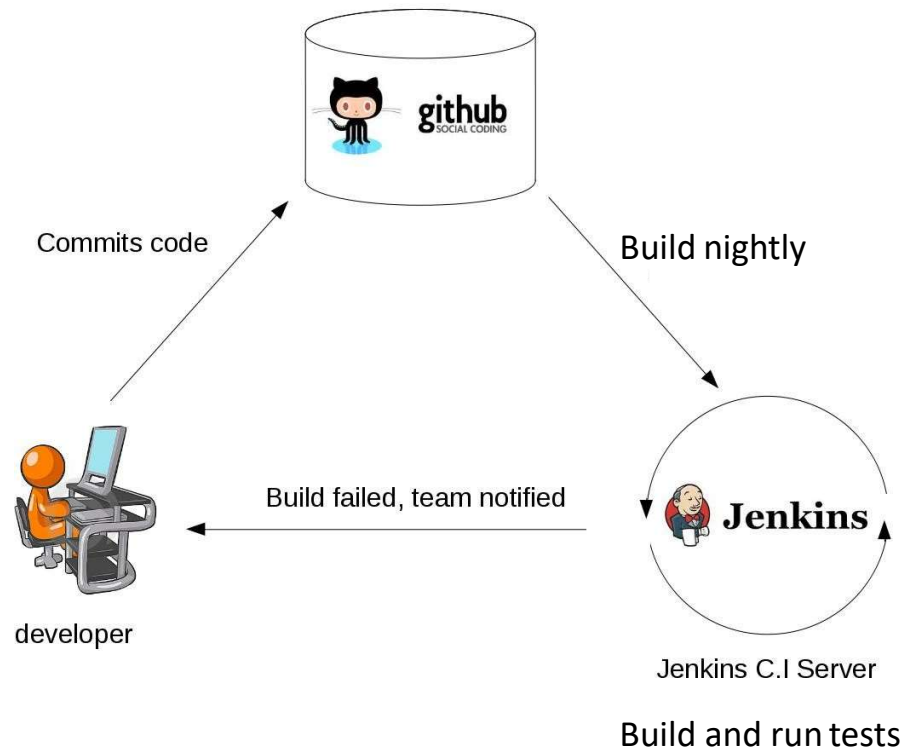


Jenkins



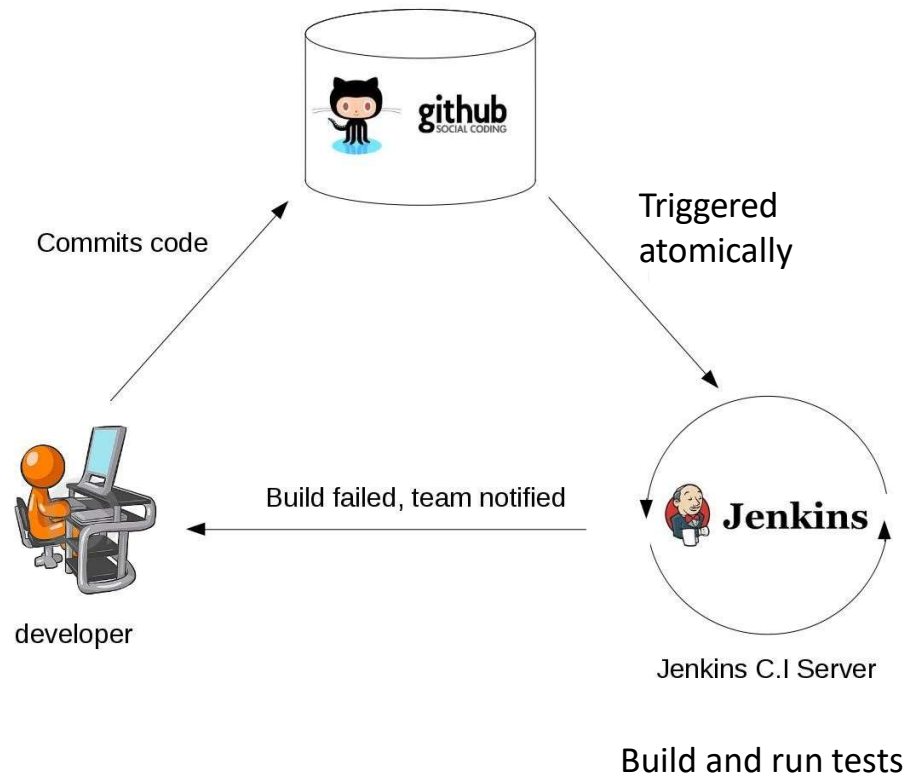
Etapa 1:

- Sem servidores de build.
- Os desenvolvedores NÃO comitam regularmente.
- As alterações são integradas e testadas manualmente.
- Poucas releases.



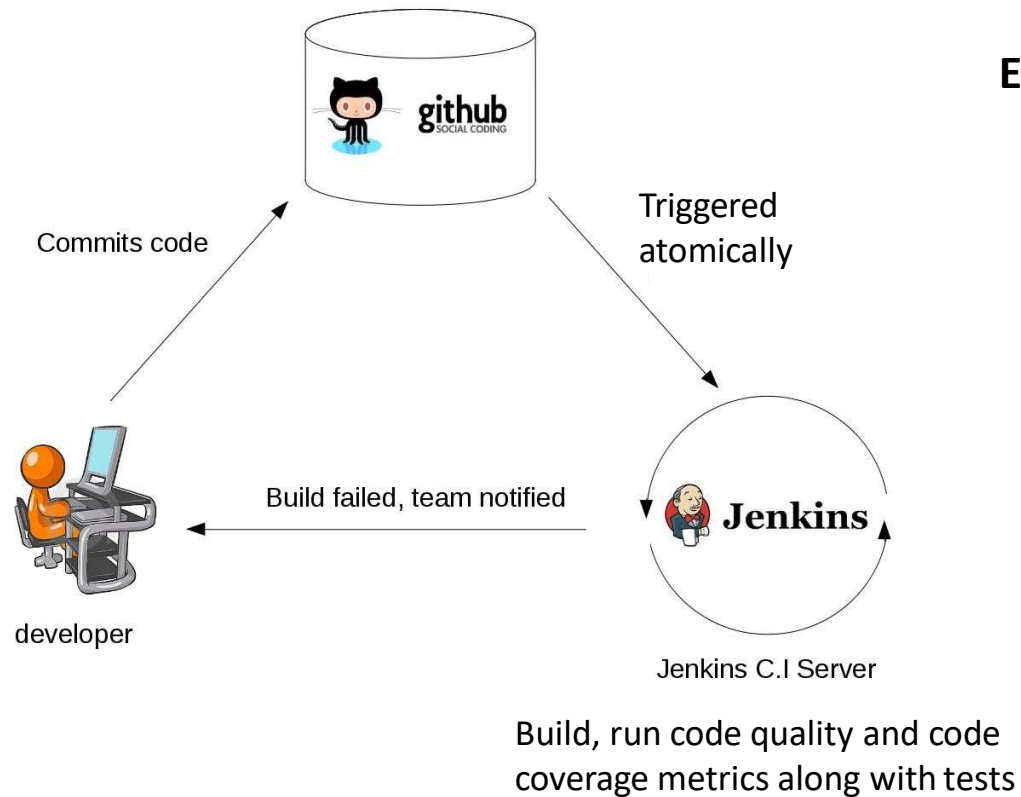
Estágio 2:

- As compilações automatizadas são programadas regularmente.
- Script de build compila a aplicação e executa um conjunto de testes automatizados.
- Desenvolvedores agora comitam suas alterações regularmente.
- Servidor de build alerta os membros da equipe em caso de falha de compilação.



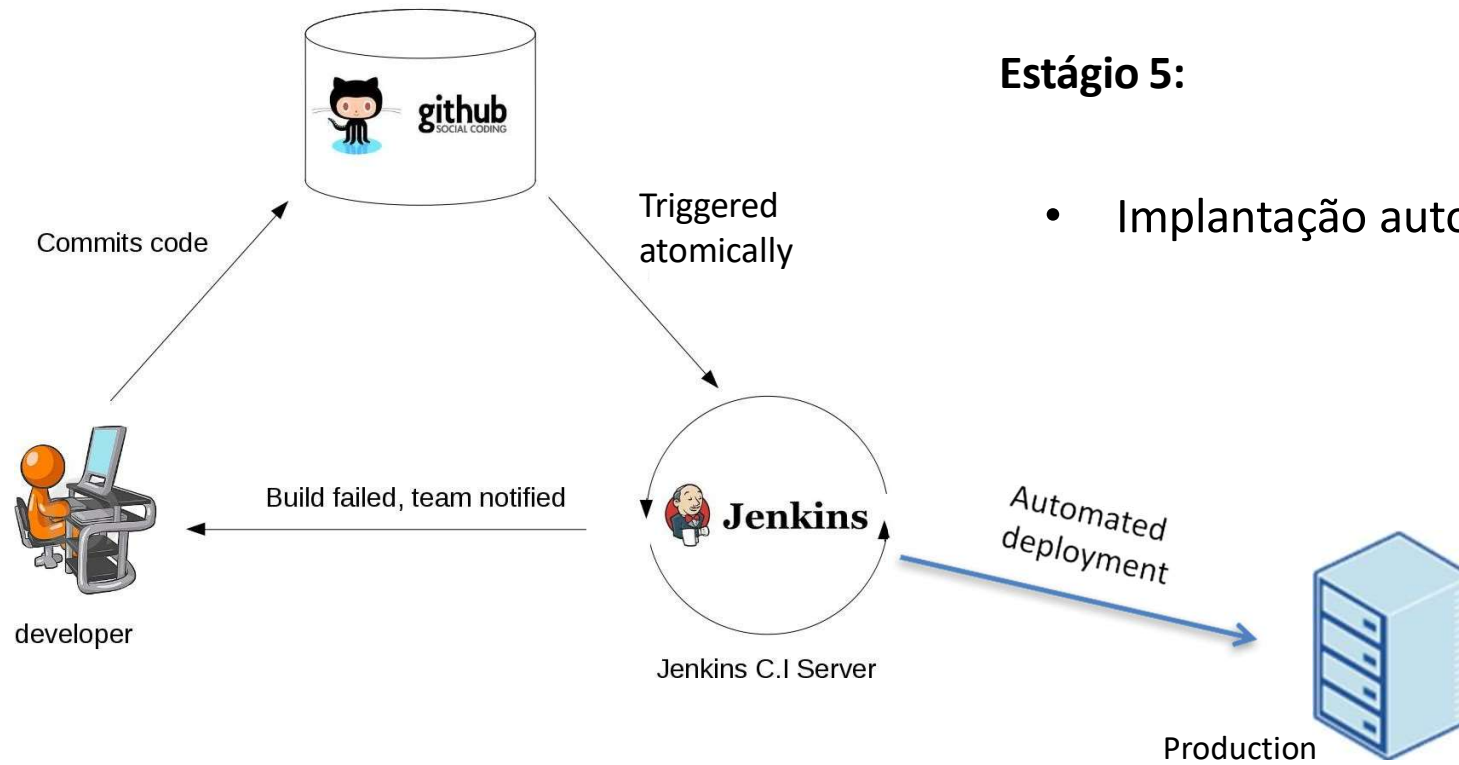
Estágio 3:

- Uma compilação é acionada sempre que novo código é alterado no repositório central.
- Builds quebradas são geralmente tratadas como um problema de alta prioridade e são corrigidas rapidamente.



Estágio 4:

- As métricas automatizadas de qualidade de código e cobertura de código são agora executadas juntamente com testes unitários para avaliar continuamente a qualidade do código.
- A cobertura de código está aumentando?
Temos cada vez menos falhas de construção?



Estágio 5:

- Implantação automatizada

Integração Contínua
Entrega Contínua
Implantação Contínua



Jenkins

- **Integração Contínua**

- A prática de realizar o merge do trabalho de desenvolvimento com o ramo principal constantemente.

- **Entrega Contínua**

- Entrega contínua de código para um ambiente destino uma vez que o código está pronto para ser enviado.
- Isso pode ser em teste ou produção. A ideia é que o produto seja entregue a uma base de usuários, que pode ser QAs ou clientes para revisão e inspeção.

- **Implantação Contínua**

- A implantação ou liberação do código para produção assim que estiver pronto.

Como implementar a Integração Contínua?



Soluções não hospedadas



Soluções hospedadas

Integração Contínua também é uma mentalidade

O conserto de builds quebradas deve ser tratada como um problema de alta prioridade para todos os membros da equipe.

O processo de implantação deve ser automatizado, sem etapas manuais envolvidas.

Todos os membros da equipe devem se concentrar em contribuir para testes de alta qualidade, pois a confiança do processo de IC depende muito da qualidade dos testes.

Uma breve introdução ao Jenkins e a história do Jenkins



Jenkins

O que é Jenkins?

- Jenkins é um servidor de integração e build contínua.
- É usado para fazer builds manualmente, periodicamente ou automaticamente de projetos de desenvolvimento de software.
- É uma ferramenta de integração contínua de código aberto escrita em Java.
- Jenkins é usado por equipes de todos os tamanhos diferentes, para projetos com várias línguas.

Por que Jenkins é popular

- Fácil de usar
- Grande extensibilidade
 - Suporte a diferentes sistemas de controle de versão
 - Métricas de qualidade de código
 - Construir notificadoros
 - Personalização da UI

-  New Item
-  People
-  Build History
-  Project Relationship
-  Check File Fingerprint
-  Manage Jenkins
-  My Views
-  Credentials

 [add description](#)

All build pipeline +

| S | W | Name ↓ | Last Success | Last Failure | Last Duration | |
|---|---|-----------------------------------|-------------------------------|------------------------------|---------------|---|
|  |  | first-jenkins-job | 11 days - #7 | 11 days - #6 | 31 ms |  |
|  |  | maven-abc | N/A | N/A | N/A |  |
|  |  | maven-project | 13 days - #26 | N/A | 4.3 sec |  |

Icon: [S](#) [M](#) [L](#)

[Legend](#)  [RSS for all](#)  [RSS for failures](#)  [RSS for just latest builds](#)

Build Queue

No builds in the queue.




















Build Executor Status

1 Idle
2 Idle

Plugins by topic

Source code management

Jenkins has native support for Subversion and CVS as well as the following plugins:

| | |
|---|---|
|  AccuRev Plugin | — This plugin allows you to use AccuRev as a SCM. |
|  Anchore Container Image Scanner Plugin | — Allows users to add a build step to run the Anchore container image scanner. |
|  archive-files-scm-plugin | — ArchiveFilesSCM - This plugin for Jenkins checkouts archive files and extracts to Jenkins job workspace |
|  AWS CodePipeline Plugin | — AWS CodePipeline is a continuous delivery service for fast and reliable application updates. |
|  Bazaar Plugin | — This plugin integrates Bazaar version control system to Jenkins. The plugin requires the Bazaar binary (bzzr) to be installed on the target machine. |
|  Bitbucket Branch Source Plugin | — Multibranch projects and Team/Project folders from Bitbucket Cloud and Server . Please note that this plugin requires a server running BitBucket 4.0 or later; Stash 3.x and earlier are not supported. |
|  BitKeeper Plugin | — Add BitKeeper support to Jenkins |
|  BlameSubversion | — This plug-in provides utilities for getting svn info from upstream job to downstream job |
|  ClearCase Plugin | — Integrates Jenkins with ClearCase . |
|  ClearCase UCM Baseline Plugin | — Allows using ClearCase UCM baselines as the input of builds: When using this SCM, users will be asked at build-time to select the baseline on which the job has to work. |
|  ClearCase UCM Plugin | — A Pragmatic integration to ClearCase UCM, simplifying continuous integration with Jenkins. |
|  Clone Workspace SCM Plugin | — This plugin makes it possible to archive the workspace from builds of one project and reuse them as the SCM source for another project. |
|  CMVC Plugin | — This plugin integrates CMVC to Hudson. |
|  Compuware Source Code Download for Endeavor, PDS, and ISPW Plugin | — The Compuware Source Code Download for Endeavor, PDS, and ISPW plugin allows Jenkins users to download Endeavor, PDS, or ISPW members from the mainframe to the PC. |
|  Config Rotator Plugin | |
|  CVS Plugin | — This bundled plugin integrates Jenkins with CVS version control system. |
|  Darcs Plugin | — This plugin integrates Darcs version control system to Jenkins. The plugin requires the Darcs binary (darcs) to be installed on the target machine. |
|  Dimensions Plugin | — This plugin integrates the Serena Dimensions CM SCM with Jenkins. |
|  File System SCM | — Use File System as SCM. |

A História de Jenkins

Hudson



Jenkins

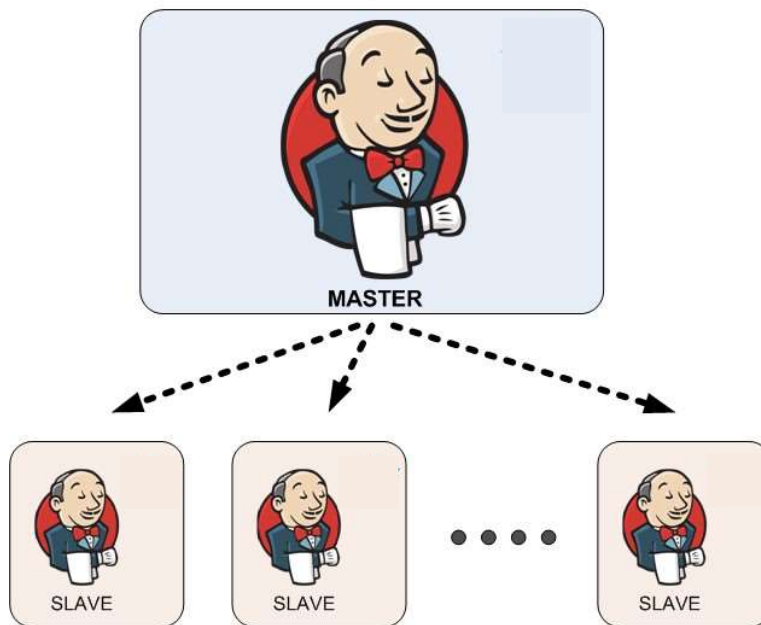
- Hudson foi iniciado em 2004 na Sun por Kohsuke Kawaguchi como um projeto de hobby.
- Primeiro lançamento em 2005.
- Kohsuke trabalhou em Hudson em tempo integral no início de 2008.
- Tornou-se a solução líder de Integração Contínua com uma participação de mercado de mais de 70% em 2010.
- Renomeado para Jenkins em 2011.

- **Arquitetura Mestre e Escravo de Jenkins**
- **Algumas terminologias importantes de Jenkins**



Jenkins

Arquitetura Mestre e Escravo de Jenkins



Mestre:

Agenda build jobs.

- Envia builds para os escravos para a execução do trabalho.
- Monitora os escravos e registra os resultados da compilação.
- Também pode executar trabalhos de build diretamente.

Escravo:

- Executar trabalhos de builds enviados pelo mestre

Trabalho / Projeto

- Esses dois termos são usados de forma intercambiável. Todos eles se referem a tarefas executáveis que são controladas / monitoradas por Jenkins.

Escravo / Nó

- Escravos são computadores que são criados para fazer build de projetos para um mestre.
- Jenkins dirige um programa separado chamado "agente escravo" em escravos.
- Quando os escravos são registrados para um mestre, um mestre começa a distribuir cargas para escravos.
- Nó é usado para se referir a todas as máquinas que fazem parte da grade jenkins, escravos e mestre.

Executor

- Executor é um fluxo separado de builds a serem executadas em um nó em paralelo.
- Um Nó pode ter um ou mais executores.

Build

- A build é resultado de um dos projetos.

Plugin

- A Plugin, como plugins em qualquer outro sistema, é um software que amplia a funcionalidade principal do servidor Jenkins núcleo.