Agile é uma metodologia de gerenciamento de projetos que se concentra em entregas incrementais e interativas, em vez de um planejamento detalhado e rígido do início ao fim. Isso significa que em vez de planejar todo o projeto de uma vez, o projeto é dividido em pequenos ciclos chamados de sprints, onde uma pequena parte do trabalho é realizado e entregue em um curto período de tempo. As equipes podem então avaliar o progresso e fazer ajustes para o próximo sprint, permitindo maior flexibilidade e adaptação a mudanças ao longo do caminho.

O **Scrum** é um dos frameworks mais populares dentro do Agile. É baseado em um processo iterativo e incremental, com uma equipe auto-organizada e multifuncional. O processo Scrum é dividido em três principais papéis: o Product Owner, o Scrum Master e a Equipe de Desenvolvimento. O Product Owner é responsável por definir os requisitos do projeto e priorizar o trabalho. O Scrum Master é responsável por garantir que a equipe esteja aderindo ao processo Scrum e removendo impedimentos. A Equipe de Desenvolvimento é responsável por realizar o trabalho real.

Kanban é outra metodologia ágil popular que se concentra em visualizar e limitar o trabalho em andamento para maximizar a eficiência da equipe. Ele utiliza um quadro Kanban, geralmente em um quadro branco ou software, para rastrear o fluxo de trabalho da equipe, permitindo que todos saibam o que está sendo feito, o que precisa ser feito e o que está esperando para ser iniciado.

O gráfico de **Burn-Down** é uma ferramenta de monitoramento de projeto comum no Agile, que mostra a quantidade de trabalho restante versus o tempo. Ele permite que a equipe visualize e rastreie seu progresso ao longo do tempo, ajudando a garantir que eles estejam no caminho certo para cumprir seus objetivos. À medida que o projeto avança, o gráfico Burn-Down mostra se a equipe está à frente ou atrás do cronograma planejado e se precisam ajustar a velocidade para atender aos objetivos do projeto.

BDD (Behavior-Driven Development) é uma metodologia de desenvolvimento de software que se concentra na comunicação e colaboração entre as partes

interessadas, como desenvolvedores, testadores e gerentes de projeto. Ele se concentra em descrever o comportamento esperado do software em termos de cenários de uso do usuário.

Gherkin é uma linguagem de especificação usada no BDD para escrever cenários de teste de aceitação. Ele permite que os usuários descrevam o comportamento esperado do software em uma linguagem simples e legível por humanos. Os cenários de teste escritos em Gherkin geralmente seguem uma estrutura padrão conhecida como Given-When-Then.

Given-When-Then é um formato comum usado para escrever cenários de teste em Gherkin. O "Given" descreve o estado inicial do sistema ou as condições necessárias para o cenário de teste. O "When" descreve a ação que está sendo executada, geralmente pelo usuário. O "Then" descreve o comportamento esperado do sistema após a ação ser executada. Juntos, essas três partes formam um cenário completo que pode ser testado para garantir que o software esteja funcionando conforme o esperado.

Em resumo, **BDD** é uma metodologia de desenvolvimento de software que se concentra em descrever o comportamento esperado do software em termos de cenários de uso do usuário, enquanto o **Gherkin** e o formato **Given-When-Then** são ferramentas usadas no BDD para especificar cenários de teste de aceitação de forma clara e legível por humanos.

Bug tracking é o processo de rastrear e gerenciar problemas ou defeitos (bugs) em um software. Ele é usado para garantir que todos os problemas identificados no software sejam registrados, monitorados e corrigidos, garantindo que o software seja entregue com o menor número possível de problemas.

Para rastrear bugs, geralmente é utilizado um sistema de gerenciamento de bugs, que é um software projetado especificamente para rastrear bugs. Existem muitas ferramentas disponíveis, incluindo o **Mantis, Jira e Azure DevOps**.

Essas ferramentas permitem que os desenvolvedores registrem os bugs que encontraram, incluindo informações sobre a gravidade, a etapa do processo em que foi encontrado, a data em que foi relatado, a pessoa que o relatou e outras informações relevantes. Os bugs podem ser atribuídos a um desenvolvedor específico para correção e o progresso pode ser monitorado ao longo do tempo.

A **rastreabilidade** é uma característica importante do gerenciamento de bugs, que é a capacidade de rastrear um bug do momento em que ele é relatado até a sua resolução final. Isso permite que os desenvolvedores e gerentes de projeto acompanhem o status de cada bug e forneçam atualizações aos usuários e partes interessadas.

Em resumo, o **bug tracking** é o processo de rastrear e gerenciar problemas ou defeitos em um software, utilizando ferramentas como **Mantis, Jira e Azure DevOps**. A **rastreabilidade** é uma característica importante dessas ferramentas, permitindo que os bugs sejam rastreados do momento em que são relatados até a sua resolução final.

Testes automatizados são testes de software que são executados automaticamente, em vez de serem executados manualmente por um testador humano. Esses testes podem incluir testes de unidade, testes de integração, testes de aceitação e outros tipos de testes que são executados repetidamente durante o ciclo de vida do desenvolvimento do software.

Para executar esses testes automatizados, são usadas ferramentas de teste automatizado, como o **Cucumber, Specflow, Selenium WebDriver e Appium**. Essas ferramentas permitem que os desenvolvedores escrevam scripts de teste que possam ser executados automaticamente em diferentes plataformas, sistemas operacionais e navegadores.

O **Cucumber** e o **Specflow** são ferramentas que permitem que os desenvolvedores escrevam testes automatizados em uma linguagem natural e legível por humanos. Esses testes podem ser escritos em inglês simples e incluem cenários de teste descritos em linguagem de negócios. Eles também permitem a colaboração entre os

desenvolvedores e as partes interessadas no processo de desenvolvimento de software.

O **Selenium WebDriver** e o **Appium** são ferramentas de teste automatizado de interface do usuário (UI) que permitem que os desenvolvedores testem a interface do usuário de seus aplicativos em diferentes navegadores e dispositivos. O Selenium WebDriver é usado principalmente para testes em navegadores da web, enquanto o Appium é usado para testes em aplicativos móveis em diferentes plataformas, como Android e iOS.

Essas ferramentas de teste automatizado são geralmente escritas em linguagens de programação, como Java e C#. O **Java** é uma linguagem de programação popular e amplamente utilizada em todo o mundo, enquanto o **C#** é uma linguagem de programação popular entre os desenvolvedores que trabalham em aplicativos para a plataforma Microsoft.

Em resumo, **testes automatizados** são testes de software que são executados automaticamente e usam ferramentas como **Cucumber**, **Specflow**, **Selenium WebDriver e Appium**. Essas ferramentas permitem que os desenvolvedores escrevam scripts de teste que possam ser executados automaticamente em diferentes plataformas e sistemas operacionais. Esses scripts são geralmente escritos em linguagens de programação, como **Java e C#**.

Controle de versão é o processo de gerenciar e acompanhar as mudanças feitas em um arquivo ou conjunto de arquivos ao longo do tempo. Ele permite que várias pessoas trabalhem no mesmo conjunto de arquivos, sem medo de sobrepor ou perder alterações.

O **Git** é um sistema de controle de versão distribuído que permite que os desenvolvedores trabalhem em projetos em equipe, rastreiem alterações em arquivos e revertam para versões anteriores se necessário. O Git é amplamente utilizado na indústria de desenvolvimento de software e é uma ferramenta importante para qualquer desenvolvedor.

O **Gitflow** é uma metodologia de desenvolvimento de software que utiliza o Git como sistema de controle de versão. Ele define um conjunto de regras e práticas para gerenciar o fluxo de trabalho em um projeto, desde o desenvolvimento de novas funcionalidades até a correção de bugs. O Gitflow permite que as equipes trabalhem em paralelo em diferentes recursos, reduzindo o risco de conflitos e aumentando a eficiência do desenvolvimento.

Em resumo, o c**ontrole de versão** é o processo de gerenciar e acompanhar as mudanças feitas em um conjunto de arquivos ao longo do tempo. O **Git** é um sistema de controle de versão distribuído amplamente utilizado, e o **Gitflow** é uma metodologia de desenvolvimento de software que utiliza o Git como sistema de controle de versão. Ambos são ferramentas importantes para desenvolvedores que trabalham em equipe.

Pipeline é um termo usado no desenvolvimento de software para descrever um processo automatizado de entrega de software. Ele é frequentemente usado em conjunto com o conceito de I**ntegração Contínua (CI)** e **Entrega Contínua/Implantação Contínua (CICD)**.

A **Integração Contínua** é uma prática que envolve a integração frequente do código-fonte de uma equipe de desenvolvimento em um repositório compartilhado. O objetivo é detectar problemas mais cedo no ciclo de desenvolvimento, identificando conflitos de integração, erros de compilação ou falhas em testes automatizados.

A **Entrega Contínua/Implantação** Contínua é uma prática que envolve a entrega automatizada de software em ambientes de produção. O objetivo é entregar software de alta qualidade com mais rapidez, segurança e eficiência, minimizando o tempo de espera entre a escrita do código e sua disponibilidade para uso em produção.

O **Jenkins** e o **GitLab** são ferramentas de automação de pipeline que permitem que os desenvolvedores construam, testem e implantem seu código de forma automatizada. Essas ferramentas fornecem uma interface visual para criar pipelines

personalizados que podem ser executados em resposta a alterações no código-fonte. Eles também oferecem recursos de rastreamento e relatório para monitorar o progresso do pipeline e identificar problemas.

Em resumo, **Pipeline** é um processo automatizado de entrega de software que pode incluir práticas de **Integração Contínua** e **Entrega Contínua/Implantação Contínua**. O **Jenkins** e o **GitLab** são ferramentas de automação de pipeline que permitem que os desenvolvedores construam, testem e implantem seu código de forma automatizada.