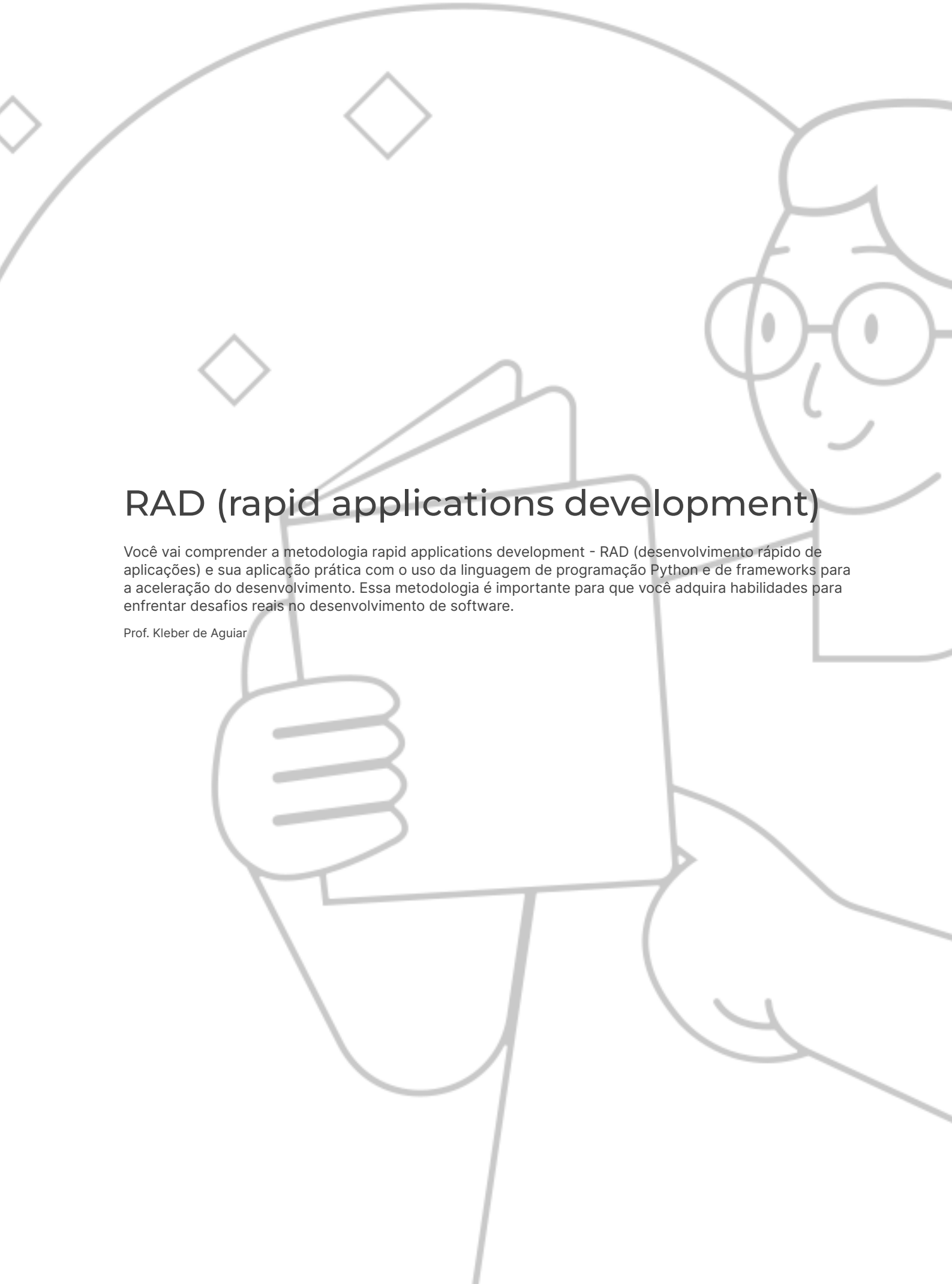


RAD (rapid applications development)

Você vai comprender a metodologia rapid applications development - RAD (desenvolvimento rápido de aplicações) e sua aplicação prática com o uso da linguagem de programação Python e de frameworks para a aceleração do desenvolvimento. Essa metodologia é importante para que você adquira habilidades para enfrentar desafios reais no desenvolvimento de software.

Prof. Kleber de Aguiar

Objetivos

- Descrever a contextualização, os conceitos, princípios, as ferramentas e técnicas da metodologia de desenvolvimento rápido de aplicações (RAD).
- Identificar as fases da RAD.
- Distinguir quando aplicar e quando não aplicar RAD.
- Justificar o Python e as ferramentas (framework) para o desenvolvimento RAD.

Introdução

Aqui, discutiremos a metodologia RAD (rapid applications development), ou desenvolvimento rápido de aplicações, uma abordagem ágil e iterativa para o desenvolvimento de software que enfatiza a prototipagem rápida e o feedback dos usuários. Descubra como a RAD facilita o desenvolvimento em ciclos curtos, permitindo ajustes contínuos por meio da colaboração ativa entre desenvolvedores e usuários. Essa abordagem é ideal para projetos que exigem entrega rápida e adaptabilidade. Assista ao vídeo para começar!

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Contextualização

Entender os fundamentos do desenvolvimento rápido de aplicações (RAD) é essencial para você, estudante e futuro profissional de tecnologia, pois oferece habilidades em metodologias ágeis de desenvolvimento. A RAD facilita a entrega rápida de protótipos funcionais, um requisito importante em um mercado competitivo que exige soluções consistentes e adaptáveis. Além disso, o conhecimento em RAD capacita os desenvolvedores a responder eficientemente às necessidades dos clientes e do mercado, utilizando um modelo iterativo e incremental que favorece a reutilização de protótipos e a integração contínua de feedback dos usuários.

Vamos explorar neste vídeo os conceitos básicos da RAD, explicando seu processo e destacando a importância da prototipagem rápida e das iterações curtas para ajustes contínuos com base no feedback dos usuários. Confira!

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

A RAD (Rapid Application Development) é uma metodologia de desenvolvimento de software focada na entrega em um período muito mais curto do que o ciclo tradicional. Não se trata da entrega final, mas de um modelo funcional que equivale funcionalmente a um componente do produto. Ou seja, simula apenas alguns aspectos do produto e é útil para o entendimento e a evolução do sistema final.

Para alcançar isso, é necessário um planejamento mínimo para obter um protótipo rapidamente. Na RAD, o foco está no desenvolvimento dos principais módulos funcionais do sistema. Essa versão inicial, limitada mas funcional, é chamada de protótipo.

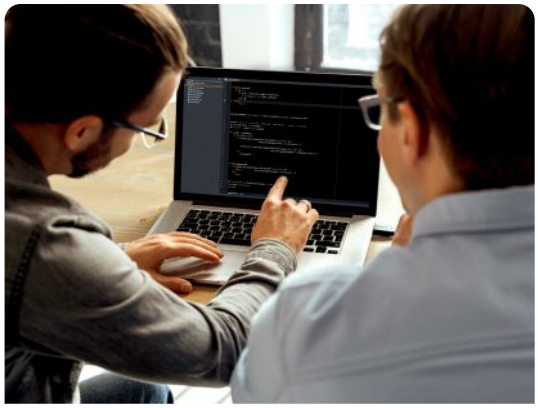
Protótipo

Conheça a seguir algumas das suas características:

- É muito útil para a compreensão do sistema.
- Serve de demonstração para os clientes.
- É mais flexível para mudanças.
- Quando está mais evoluído, pode ser integrado ao produto completo para uma entrega mais rápida da versão final.

O protótipo é fundamental no desenvolvimento de sistemas, oferecendo vantagens como melhor compreensão do sistema, facilidade de demonstração para clientes, flexibilidade para mudanças e integração gradual ao produto final, o que agiliza a entrega e aumenta a eficiência do desenvolvimento.

A RAD também pode ser aplicada para aperfeiçoar o seu treinamento prático como estudante de computação. Isso porque você pode aplicar o conhecimento adquirido para desenvolver sistemas em etapas, conforme é proposto pela RAD. Como será mostrado mais adiante, o fator humano é um importante requisito para a aplicação dessa metodologia, então, a sua aplicação para treinar recursos humanos pode acelerar a curva de aprendizado dentro de um curto período.



Aplicação de treinamento prático da RAD para estudante.

Os projetos RAD seguem o modelo iterativo e incremental. O desenvolvimento iterativo promove progressos sucessivos, em que o produto é refinado por etapas. No modelo incremental, o software é entregue em pedaços, que são chamados de incrementos. A ideia é que o software seja criado em ciclos curtos, com introdução de funcionalidades, coleta de feedback e revisão. As equipes de desenvolvimento são pequenas, compostas por desenvolvedores, analistas de negócio e representantes de clientes.

> Um dos aspectos mais importantes do modelo iterativo e incremental é garantir que os protótipos desenvolvidos sejam reutilizáveis para o projeto do sistema, ou seja, a ideia não é criar unidades descartáveis.

⚠

Atenção

A RAD foca o desenvolvimento rápido por meio de iterações frequentes e feedback contínuo.

O modelo RAD foi introduzido pelo consultor de tecnologia e autor James Martin em 1991. Surgiu como o reconhecimento da necessidade de atender o competitivo mercado de software, que tem uma demanda contínua por novas aplicações. Uma característica explorada para a formalização da RAD foi a flexibilidade do desenvolvimento de software para projetar modelos de desenvolvimento. Trata-se de uma combinação de sessões JAD, desenvolvimento de protótipos, equipes SWAT, entregas com prazo de entrega e ferramentas CASE.

RAD

Na sequência, observe algumas de suas vantagens. Acompanhe!

- É muito prático em diversos ambientes modernos de desenvolvimento.
- Apresenta uma abordagem útil para criar aplicações de comércio eletrônico e aplicativos de dispositivos móveis.
- Possui uma velocidade de entrega que pode determinar o posicionamento de uma empresa em um ambiente de mercado muito competitivo.

Trata-se de uma metodologia importante a ser empregada para que as empresas lancem suas aplicações antes de seus concorrentes.

Observe como iniciar um projeto RAD. Vamos lá!

Aplicação da metodologia JAD
Aplicar a metodologia Joint Application Development (JAD), na qual usuários e analistas projetam o sistema juntos, sob uma liderança em oficinas de trabalho.

Dinâmicas de grupo
Potencializar o resultado do desenvolvimento através de dinâmicas de grupo.

Objetivos e aplicações do sistema
Definir os objetivos e as aplicações do sistema, desde a geração de telas até a geração de relatórios.

Princípios
Estabelecer a dinâmica de grupo, os recursos audiovisuais, o processo organizado e racional, a escolha do local e documentação com a abordagem WYSIWIG – "O que você vê é o que você obtém".

A RAD foi a precursora do gerenciamento ágil de projetos. As características de prototipagem rápida e ciclos de liberação e iterações mais curtos fortaleceram o posicionamento da RAD como um método eficaz no desenvolvimento de software, tornando-se cada vez mais popular entre as empresas ágeis que procuram métodos que acompanhem o crescimento de suas necessidades comerciais e de clientes. Trata-se de uma metodologia orientada pelo feedback do usuário, e não por um planejamento detalhado e caro.

Os métodos tradicionais de desenvolvimento de software, como a metodologia de desenvolvimento **cascata**, seguem modelos rígidos de processo. Isso significa que, nesses modelos tradicionais, os clientes são pressionados a estabelecer os requisitos antes do início do projeto. A iteração ao longo do projeto é baixa, o que complica o processo de mudança para novos requisitos e ajustes de viabilidade.

Metodologia de desenvolvimento cascata.

Atividade 1

O desenvolvimento rápido de aplicações (RAD) é uma metodologia que enfatiza a rápida prototipagem e o feedback contínuo. Qual é o principal objetivo da RAD?

A

Substituir completamente métodos tradicionais sem feedback do usuário.

B

Priorizar planejamento detalhado e desenvolvimento de longo prazo.

C

Melhorar a eficiência no desenvolvimento, diminuindo o tempo e aumentando a satisfação dos clientes.

D

Eliminar a prototipagem e iterações, focando uma entrega única.

E

Usar grandes equipes para construir soluções complexas antes de qualquer feedback.

A alternativa C está correta.

A RAD tem como foco a entrega rápida de protótipos, permitindo ajustes rápidos e adaptação às mudanças do mercado, através de feedback contínuo dos usuários. Com isso, os produtos tendem a ter uma melhor aceitação, já que os clientes participam ativamente do processo de desenvolvimento.

Conceitos da RAD

A RAD se destaca por sua eficiência em produzir software de alta qualidade de forma ágil e econômica, sendo caracterizado por seu modelo adaptativo e iterativo. A correta aplicação dessa metodologia permite a mobilização de equipes mais enxutas e eficientes, o que consiste em um importante diferencial para organizações que necessitam entregar produtos em ciclos cada vez menores. Profissionais que conhecem e aplicam a RAD podem se destacar no mercado, propiciando às empresas um conceito mais ágil de desenvolvimento de software.

Neste vídeo, abordaremos os princípios fundamentais da metodologia de desenvolvimento rápido de aplicações (RAD), destacando seus quatro elementos essenciais e explorando seus dois principais tipos: faseado e intensivo. Assista!

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

A RAD é uma abordagem interativa com o objetivo de produzir o desenvolvimento de software de alta qualidade. O resultado da aplicação da RAD é um software com menor custo, menos erros e menor tempo de desenvolvimento.

> A RAD pode ser considerada um tipo de técnica ágil.
>
> —
>
> (Naz & Khan, 2015)

A metodologia RAD combina diversas técnicas para acelerar o desenvolvimento de aplicações de software. Outra forma pela qual a RAD é conhecida é como construção rápida de aplicações, do inglês rapid application building (RAB). Um dos principais elementos da RAD é o desenvolvimento de protótipos para chegar ao sistema final. Trata-se de um modelo adaptativo, uma vez que o desenvolvimento é feito em iterações em que mudanças podem ser realizadas a partir dos comentários do usuário. A ênfase está na criação rápida de um protótipo, em vez de um planejamento detalhado.

A metodologia RAD possui quatro elementos fundamentais. Vejamos!

Uso de ferramentas para dar suporte ao desenvolvimento

O uso de ferramentas CASE facilita a automação no desenvolvimento de sistemas. Isso é obtido através de recursos como geração de código e verificação automática de erros de consistência. As ferramentas CASE, portanto, são usadas para gerar protótipos, dando, assim, suporte ao desenvolvimento iterativo, possibilitando que os usuários finais acompanhem a evolução do sistema à medida que ele está sendo construído.

Metodologia bem definida

É seguido um processo formal de desenvolvimento com atividades em etapas e entregas intermediárias. As tarefas são organizadas de modo a não negligenciar nenhum dos aspectos pré-acordados, e as técnicas são documentadas para garantir que uma tarefa seja executada da maneira correta.

Pessoas

Deve haver treinamento das pessoas tanto na metodologia de trabalho como no uso das ferramentas. As tarefas devem ser distribuídas por pequenas equipes, que devem trabalhar bem juntas.

Gestão

O gerenciamento do projeto deve ser feito com rapidez. Isso é obtido através de oficinas de planejamento de requisitos e design de sistema para extrair rapidamente os requisitos dos usuários. Além disso, deve ser feita alocação de tempo fixo, que é chamado de Timebox para entregar iterativamente o sistema para os usuários.

Para entendermos melhor, Timebox é o tempo máximo estabelecido para atingir as metas, tomar uma decisão ou executar um conjunto de tarefas.

Além disso, existem dois tipos de projetos RAD. Confira!

Intensivo

Nesse projeto, uma equipe de desenvolvedores e usuários trabalham por um curto período (algumas semanas) e, ao final desse tempo, espera-se que produza um produto que seja utilizável.

Faseado

Projeto distribuído por um longo período. É normalmente iniciado por um workshop JAD. As fases subsequentes do projeto são geralmente organizadas em termos de entrega e demonstração de protótipos incrementais. O objetivo é refinar continuamente o protótipo, tornando-o algo que seja entregue no final do timebox.

A criação rápida de protótipo é a base da RAD. Nas situações em que os projetos são orientados por requisitos de interface do usuário, o desenvolvimento de protótipo é uma escolha muito adequada, pois é normal que o usuário crie a ideia de como a interface do sistema deve ficar ao longo do desenvolvimento do projeto. O desenvolvimento rápido de protótipos tem como pré-requisito o uso de ferramentas com suporte a componentes gráficos.

No mercado, desde a década de 1990, existiam diversas ferramentas para esse fim, em que os programadores simplesmente podem selecionar um componente gráfico e arrastá-lo para um formulário. Desse modo, as interações com os usuários finais são mais produtivas, pois, constantemente, recebem um software operacional.

Atividade 2

A metodologia de desenvolvimento rápido de aplicações (RAD) prioriza a agilidade e flexibilidade no desenvolvimento de software, com ênfase na integração entre desenvolvedores e clientes. Qual das seguintes características está associada à metodologia RAD?

A

Planejamento detalhado e extensivo antes do início do desenvolvimento.

B

Iterações rápidas baseadas no feedback dos usuários.

C

Grandes equipes trabalhando de forma independente.

D

Desenvolvimento linear sem iterações ou revisões.

E

Ignorar o feedback dos usuários durante o processo de desenvolvimento.

A alternativa B está correta.

A RAD enfoca o desenvolvimento iterativo e ágil, utilizando feedback dos usuários para fazer ajustes contínuos. As outras opções contradizem princípios fundamentais da RAD, que rejeita abordagens de planejamento extenso, trabalho em grandes equipes independentes e processos lineares sem feedback.

Princípios, ferramentas e técnicas

Para entender e aplicar corretamente a metodologia RAD, devemos observar seus princípios. Esses princípios facilitam a adaptação rápida às mudanças de requisitos e promovem uma colaboração efetiva entre desenvolvedores e usuários. A RAD reduz significativamente o tempo de desenvolvimento e aumenta a qualidade do software. Ao dominar esses conceitos, você ganhará habilidades para produzir soluções de software mais alinhadas às necessidades do usuário, melhorando sua capacidade de inovação e competitividade no mercado de tecnologia.

Neste vídeo, discutiremos a importância da colaboração entre desenvolvedores e usuários, o uso de ferramentas de automação e como metodologias ágeis ajudam pequenas equipes a entregar produtos rapidamente. Assista!

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Princípios

Constantemente, os programadores são pressionados a entregar as aplicações em prazos curtos e, muitas vezes, sabe-se com antecedência que o projeto terá de passar por modificações ao longo do desenvolvimento. Essas situações são exemplos em que o desenvolvimento rápido é bastante útil, pois ele está embasado exatamente na entrega rápida de protótipos que incorporam os comentários e as solicitações dos usuários a cada entrega. Para ser eficaz, no entanto, a RAD tem alguns requisitos que não são triviais. Alguns requisitos relacionados aos recursos humanos são os seguintes:

- Equipe de desenvolvedores qualificada e motivada.
- Usuários comprometidos com a participação ativa ao longo do projeto.
- Comprometimento para atingir o resultado satisfatório.

O desenvolvimento baseado na entrega de protótipos funcionais busca dar a oportunidade para que o usuário possa interagir com o projeto antes de receber o sistema final. Dessa forma, poderá fazer comentários e solicitações que guiarão os desenvolvedores na confecção do produto que atenda às suas expectativas sob o ponto de vista de funcionalidades, recursos, interatividade do sistema (experiência do usuário), relatórios, gráficos, entre outros.

Segundo Fitzgerald (1998), a RAD é baseado nos seguintes princípios básicos:

Envolvimento ativo dos usuários

A metodologia RAD reconhece que o envolvimento do usuário é necessário para reduzir problemas caros de obtenção de requisitos. Além disso, os usuários podem rejeitar completamente os sistemas se não estiverem suficientemente envolvidos no desenvolvimento. No centro da abordagem da RAD, estão as oficinas de design de aplicativos conjuntos (JAD) e planejamento de requisitos conjuntos.

Equipes pequenas com poder de decisão

As vantagens da elaboração de equipes pequenas estão na redução de ruídos de comunicação e na minimização de atrasos devido à burocracia que a hierarquia de uma metodologia tradicional impõe. Em relação aos ruídos de comunicação, os canais que tratam dessa área aumentam proporcionalmente ao tamanho da equipe, portanto, equipes pequenas evitam a distorção e o conflito na comunicação. A respeito da redução do tempo, empoderar a equipe aumenta as chances de cumprir os prazos por causa da responsabilidade de tomada de decisão. As equipes têm o poder de tomar decisões sobre o design (embora as mudanças sejam reversíveis).

Entrega frequente de produtos

Diferentemente das metodologias de desenvolvimento tradicionais, em que os projetos podem levar muito tempo para serem concluídos, a RAD procura reduzir o tempo de desenvolvimento. Portanto, prazos mais curtos para o desenvolvimento são uma característica importante. Em vez de se concentrar no processo, a RAD tem como premissa a entrega de produtos que satisfazem os requisitos funcionais.

Desenvolvimento incremental e iterativo

Na RAD, os sistemas evoluem de forma incremental em cada iteração. A cada nova iteração, surgem novos requisitos que são incorporados ao sistema. Desse modo, os sistemas evoluem através da prototipagem iterativa. Existe um entendimento na RAD que a especificação de requisitos é um processo não determinístico e que evolui à medida que desenvolvedores e usuários interagem com o protótipo do sistema.

Abordagem top-down

Uma vez que, na metodologia RAD, os requisitos não precisam ser completamente definidos logo no início do projeto, eles são especificados em um nível apropriado ao conhecimento disponível no momento. Esses são então elaborados através de prototipagem incremental. Os sistemas são elaborados e confeccionados à medida que o conhecimento cresce. Além disso, como se trata de uma abordagem de “cima para baixo” caracterizada por um curto período, todas as decisões são consideradas reversíveis rapidamente.

Utilização de ferramentas de automação (case)

Trata-se de usar programas que facilitem a automação de processos, criação de diagramas, realização de testes e quaisquer tarefas que facilitem as entregas dentro dos prazos pré-estabelecidos e, obviamente, com qualidade. Além disso, essas ferramentas facilitam a reutilização de componentes que podem ser usados ao longo do projeto.

O ponto fundamental na metodologia RAD é que se trata de uma abordagem colaborativa entre todas as partes interessadas, que são: patrocinadores, desenvolvedores e usuários ao longo da vida de um projeto.

Ferramentas e técnicas

A RAD precisa ser suportada por ferramentas que auxiliem no desenvolvimento das aplicações rapidamente. Entre as categorias de ferramentas que dão suporte à RAD para desenvolver projetos de software estão:

- Integração de dados
- Ambientes de desenvolvimento
- Ferramentas de coleta de requisitos
- Ferramentas de modelagem de dados
- Ferramentas de geração de código

Desde que a RAD foi formalizada, foram desenvolvidas muitas técnicas para a sua utilização. Cada uma das técnicas tem suas particularidades, mas mantém a essência da RAD. Aqui estão algumas dessas técnicas (Naz; Khan, 2015), veja!

Modelo CBD

Trata-se o método que descreve como componentes antigos podem ser reutilizados com os novos.

RepoGuard
É um framework para integração de ferramentas de desenvolvimento com repositórios de código-fonte.

Adição dinâmica ágil
São técnicas usadas para integração do ágil para tornar o projeto mais adaptável.

Método baseado em camadas para desenvolvimento rápido de software
É baseado em camadas e segue o XP (Extreme Programming), uma metodologia de desenvolvimento de software que visa maximizar a qualidade e responder rapidamente às mudanças nos requisitos do cliente.

Análise de projeto de sistema baseado em simulação
Refere-se ao desenvolvimento de ferramentas ágeis baseadas em simulação.

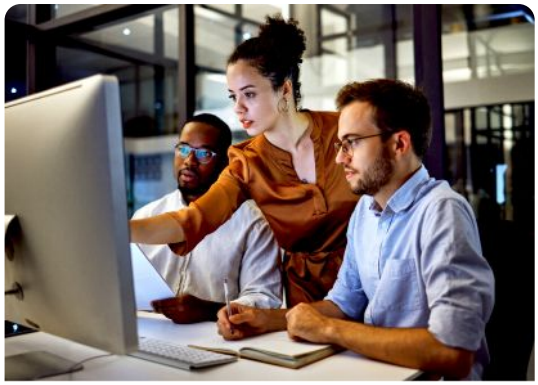
Uso de Ajax na RAD
Trata-se da prototipagem rápida em aplicativos e ferramentas da web.

Desenvolvimento de aplicativos multiusuário em ambiente distribuído rapidamente
É a Middleware de comunicação.

Programação extrema
Refere-se à adição de reutilização ao XP.

A ideia do uso das técnicas de RAD é de otimizar os resultados obtidos dentro do tempo estimado que, pela natureza da RAD, é curto. Essencialmente, um software é construído para atender a alguma demanda, ou seja, existe uma razão para que seja confeccionado.

A interação com os usuários auxilia o entendimento dos desenvolvedores para construir, agregar e incorporar esse entendimento em um protótipo através de técnicas e ferramentas que **acelerem** a entrega e **reduzam** os desvios de compreensão. A concordância sobre o propósito do sistema e a sua evolução é muito importante para o sucesso do projeto. Tanto desenvolvedores como clientes devem estar envolvidos em interações formais que fortaleçam o comprometimento de todos.



Equipe de desenvolvedores em interação com usuário.

A pressão por soluções de software confiáveis e em curtos prazos favoreceu a criação da metodologia de desenvolvimento rápido de aplicações (RAD). A ideia de entregar protótipos em um ciclo de desenvolvimento incremental e iterativo permite que o usuário possa ter rapidamente uma visão clara de como o sistema está progredindo e se existe alguma questão relacionada aos requisitos que precisa ser aperfeiçoada. Portanto, a colaboração entre desenvolvedores e usuários suporta o desenvolvimento de especificações mais precisas e validadas.

Atividade 3

No cenário atual de desenvolvimento ágil de software, a metodologia RAD é notável por sua capacidade de acelerar a entrega de projetos. Qual dos seguintes princípios é um princípio da RAD?

A

Rígida aderência a um planejamento em longo prazo.

B

Independência dos usuários no processo de desenvolvimento.

C

Desenvolvimento incremental e iterativo.

D

Grandes equipes centralizadas.

E

Ciclos de feedback longos e infrequentes.

A alternativa C está correta.

O princípio do desenvolvimento incremental e iterativo é fundamental para a RAD, permitindo que o software evolua por meio de prototipagem rápida e feedback constante, integrando novos requisitos em cada iteração.

Introdução às fases da RAD

Conhecer as fases da RAD é muito importante para aproveitar ao máximo suas vantagens, como a redução do tempo de desenvolvimento e o aumento da satisfação do cliente. Cada etapa (planejamento de requisitos, design do usuário, construção e transição) é projetada para envolver ativamente usuários e outras partes interessadas, garantindo que o feedback seja integrado rapidamente e que os requisitos do sistema sejam continuamente ajustados para melhor atender às necessidades do negócio. Assim, a familiaridade com essas fases permite que as equipes de desenvolvimento naveguem eficientemente pelo processo RAD, minimizando retrabalhos e maximizando a entrega de valor.

Exploraremos neste vídeo as quatro fases da metodologia RAD e como ela acelera o desenvolvimento de software e promove a colaboração entre equipes. Confira!

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

A metodologia RAD é caracterizada pelo desenvolvimento do projeto através de etapas iterativas e incrementais, em que um protótipo é entregue ao final de cada ciclo. A proposta é que haja redução nas atividades relacionadas ao planejamento em detrimento do processo de desenvolvimento através de um processo que se caracteriza por incrementos de funcionalidades a cada nova iteração.

Desse modo, a expectativa é que as equipes produzam mais em menos tempo, maximizando a satisfação do cliente, uma vez que ele é envolvido no processo. Isso ocorre porque a RAD é estruturada para que as partes interessadas interajam e possam detectar a necessidade de alterações do projeto em tempo real, sem a necessidade de completar longos ciclos de desenvolvimento, e os desenvolvedores possam realizar as implementações rapidamente ao longo das iterações.

Equipe de desenvolvedores em reunião com cliente.

O ciclo de vida da RAD foi projetado para direcionar os desenvolvedores na criação de soluções de software que atendam às necessidades dos usuários. Esse ciclo de vida trata das atividades que são necessárias para definir o escopo e os requisitos de negócios, além das atividades para projetar, desenvolver e implementar o sistema. Na abordagem de James Martin (1991), a metodologia RAD possui quatros fases distintas. Vamos conhecê-las!

Planejamento de requisitos

Nessa fase, os usuários, gerentes e desenvolvedores estudam as necessidades de negócios, o escopo do projeto, as restrições e os requisitos do sistema. A gerência só autoriza a continuidade do projeto depois que os membros das equipes concordam sobre o entendimento dos requisitos do sistema.

Design do usuário

São desenvolvidos modelos e protótipos, através da interação de usuários e desenvolvedores, para representar todos os processos, as entradas e saídas do sistema. Para isso, são usadas uma combinação de técnicas JAD (joint application development) e ferramentas CASE para representar as demandas do usuário em modelos de trabalho.

Construção

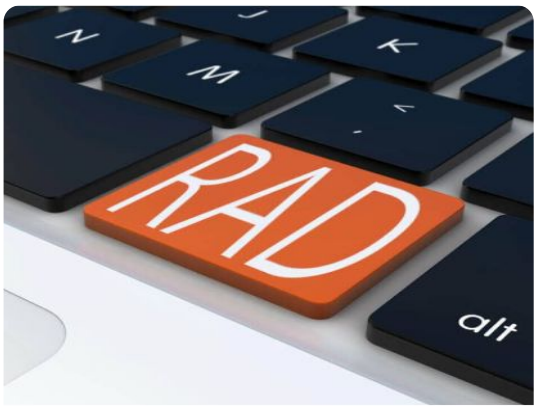
Nessa fase os protótipos são desenvolvidos. A interação entre usuários e desenvolvedores continua, para que haja sugestões sobre alterações, ajustes, ou melhorias, à medida que unidades do sistema, como telas ou relatórios reais, por exemplo, são desenvolvidas.

Transição

Aqui são realizados o processamento de dados, que envolve a coleta e organização de informações essenciais, a execução de testes minuciosos para garantir a funcionalidade e a segurança, a transição cuidadosa para o novo sistema, assegurando que todas as operações sejam transferidas corretamente, e o treinamento abrangente do usuário, preparando-o para utilizar o novo sistema.

O planejamento de requisitos está focado em determinar as metas e expectativas do projeto e quais são os potenciais problemas que podem ser impeditivos para o desenvolvimento do software. No caso da RAD, em que a entrega rápida de resultados é um dos objetivos principais, a identificação prévia dos requisitos funcionais é muito importante.

Na fase de design do usuário, a interação entre os desenvolvedores e os usuários é constante no desenvolvimento de modelos e protótipos que abordam todos os processos, entradas e saídas do sistema. Na fase de construção, converte o protótipo aprovado da fase de design do usuário em um modelo de trabalho. Como já houve bastante interação entre usuários e desenvolvedores na fase anterior, agora o foco dos desenvolvedores está na construção do modelo de trabalho final. Por fim, na fase de transição, o produto está pronto para ser lançado. Aqui, o usuário deve passar por treinamento para começar a usar o sistema.



Tecla RAD do teclado ativada.

Existem outras abordagens sobre a divisão das fases da RAD. Por exemplo, uma das mais conhecidas é a de James Kerr (Kerr &; Hunter, 1994), em que existem cinco fases distintas. Vamos conhecê-las!

Modelagem de negócios

As informações sobre os requisitos funcionais do sistema são coletadas de diversas fontes relacionadas aos negócios. O modelo de negócios do produto em desenvolvimento é definido em termos de fluxo de informações, obtidas através de canais como entrevistas com usuários do sistema e outras fontes relevantes. Essas informações são então combinadas para criar uma documentação que será usada para modelar o ciclo de vida dos dados: como são utilizados, quando são processados e como se transformam em informações úteis para áreas ou setores específicos do negócio. Assim, realiza-se uma análise com foco comercial para identificar dados essenciais para os negócios: como são obtidos, processados e convertidos em informações úteis.

Modelagem de dados

Todas as informações obtidas durante a fase de modelagem de negócios são analisadas para formar conjuntos de objetos de dados essenciais para a empresa. A análise agrupa essas informações de maneira útil, determinando, por exemplo, as entidades principais que o sistema irá tratar. A qualidade de cada grupo de dados é então examinada e descrita com precisão. Em seguida, é feito um mapeamento que relaciona esses grupos e define o significado desses relacionamentos, conforme estabelecido na modelagem de negócios. Avançando mais, é necessário identificar e definir os atributos de todos os conjuntos de dados, bem como estabelecer e detalhar as relações entre esses objetos de dados, conforme sua relevância para o modelo de negócios.

Modelagem de processos

Todos os grupos de dados coletados durante a etapa de modelagem de dados são analisados do ponto de vista do processamento, ou seja, como os dados são convertidos em informações úteis. Durante a fase de modelagem de processos, podem ser feitas mudanças e otimizações, além de definir os conjuntos de dados com mais detalhes. Descrições para adicionar, remover ou alterar objetos de dados também são criadas nesta fase. A ideia é que os conjuntos de objetos de dados definidos anteriormente sejam convertidos para estabelecer o fluxo de informações necessário para atingir objetivos específicos de negócios, conforme o modelo de negócios. A modelagem do processamento dos dados, para alterá-los ou utilizá-los em outras operações, é definida nessa fase. Aqui, também são descritos os processos para adicionar, excluir, recuperar ou modificar um objeto de dados.

Geração da aplicação

Todas as informações coletadas são codificadas e é construído o sistema que será usado para criar o protótipo. Os modelos de dados criados são transformados em protótipos reais que podem ser testados na próxima fase. O sistema real é construído, e a codificação é feita usando ferramentas de automação para converter modelos de processo e dados em protótipos reais.

Teste e modificação

São feitos testes dos protótipos criados. Cada módulo é testado de modo a identificar e adaptar os componentes para criar o produto mais eficaz. Como a maioria dos elementos já foi examinada anteriormente, a expectativa é que haja grandes problemas com o protótipo. O tempo total de teste é reduzido no modelo RAD, pois os protótipos são testados independentemente durante cada iteração. No entanto, o fluxo de dados e as interfaces entre todos os componentes precisam ser exaustivamente testados com uma cobertura de teste completa. Como a maioria dos componentes de programação já foi testada, o risco de problemas importantes é minimizado.

O princípio-chave do processo RAD é a redução de atividades burocráticas para se concentrar em um processo iterativo de design e construção, permitindo que as equipes realizem mais em menos tempo, sem

afetar a satisfação do cliente. As fases de prototipagem e construção rápida podem ser repetidas, até que o proprietário e os usuários do produto se sintam seguros de que o protótipo e a forma como foi construído atendem aos requisitos do projeto.

Nos métodos tradicionais, como a metodologia **cascata**, por exemplo, demora bastante até que os desenvolvedores recebam comentários dos usuários, aumentando, assim, as chances de ser necessário refazer partes do sistema, ou seja, retrabalho.

> Um dos maiores benefícios da RAD são os comentários dos usuários através da constante interação.

O ponto do projeto em que fica mais evidente essa interação está nos componentes de UI/UX do sistema. Com os protótipos, os usuários podem ter mais clareza sobre a forma como o projeto está avançando, e os desenvolvedores podem medir riscos na escolha de tecnologias que venham prejudicar a entrega do projeto, podendo, assim, fazer escolhas em tempo hábil.

Atividade 1

No contexto da metodologia de desenvolvimento rápido de aplicações (RAD), é essencial compreender a sequência e o propósito de cada fase para garantir o sucesso do projeto. Qual das seguintes opções melhor descreve a fase de construção nessa metodologia?

A

É a fase inicial, em que os requisitos do negócio e do sistema são definidos com a participação de todas as partes interessadas.

B

São criados modelos e protótipos detalhados que representam processos, entradas e saídas do sistema.

C

A construção é o momento em que os protótipos são transformados em modelos operacionais do sistema, com contínua interação entre usuários e desenvolvedores para refinamentos.

D

Fase em que ocorrem a implementação do sistema em ambiente de produção e o treinamento dos usuários finais.

E

Consiste na análise e modelagem dos processos de negócios que irão orientar o desenvolvimento dos sistemas de informação.

A alternativa C está correta.

A fase de construção na RAD envolve transformar protótipos em modelos operacionais, ajustando-os conforme o feedback dos usuários para refinar o sistema final. Essa etapa garante o funcionamento do produto em ambientes reais, cujas necessidades são diferentes da fase de prototipagem.

Ciclo de desenvolvimento

É iniciado com um planejamento conciso em que as necessidades e os requisitos são definidos em colaboração com usuários finais. Esse processo é marcado pela construção de protótipos funcionais em estágios iniciais, permitindo ajustes contínuos. Ao reduzir a formalidade na documentação inicial e enfatizar a entrega rápida de versões testáveis, a RAD acelera significativamente o ciclo de desenvolvimento, aumentando a satisfação do cliente e a eficácia do produto. Em um cenário marcado por constante mudança e prazos cada vez menores, os profissionais capazes de lidar com esse ciclo de iterações costumam ter mais possibilidades de atuação no mercado.

Neste vídeo, abordaremos o ciclo de desenvolvimento da metodologia RAD, explorando cada etapa e compreendendo como ele permite a entrega rápida de aplicações de alta qualidade. Assista!

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Os requisitos, definidos em parceria com os usuários finais, incluem:

- Metas
- Expectativas
- Cronogramas
- Orçamento

O cliente deve fornecer a visão do produto e, junto com outras partes interessadas, realizar pesquisas para finalizar os requisitos, garantindo a aprovação de todos. É essencial que todos os envolvidos compreendam claramente o projeto desde o início do ciclo de desenvolvimento, ajudando as equipes a evitar falhas de comunicação e erros custosos.

Outro ponto importante a ser destacado é que um dos princípios fundamentais da RAD é a capacidade de alterar os requisitos em qualquer momento do ciclo de desenvolvimento. Acompanhe!

Levantamento de requisitos

Logo no início do levantamento de requisitos, é feita uma pesquisa do ambiente interno para compreender de que forma atender o projeto que será iniciado. Logo depois, é desenvolvido o escopo do sistema proposto. Os processos de negócios e os dados com os quais o sistema trabalhará são usados para definir as suas respectivas funcionalidades. Características como benefícios, custos e riscos potenciais do sistema são identificadas. Passa-se, então, para a documentação de possíveis problemas de gerenciamento, e, por fim, formaliza-se o escopo do sistema. Faz-se também uma estimativa dos recursos e do tempo de implementação. Se o custo e a duração do desenvolvimento já forem definidos, é necessário analisar com mais cuidado o escopo do projeto para verificar se é viável.

Oficinas JAD

Para completar a análise de negócios e de dados do sistema, são feitas as oficinas JAD, em que são realizados revisões e detalhamento do escopo do sistema para garantir que as entregas ocorram dentro do prazo. Um exemplo de resultado obtido dessa fase são a definição das regras de negócios a serem aplicadas em cada atividade e os atributos de cada entidade. Aqui, elementos de UI/UX, tais como layouts provisórios de telas e dos principais relatórios, são desenvolvidos. Além disso, um esboço do projeto do sistema é desenvolvido. Com a conclusão do design, passa-se a mapear as funcionalidades do sistema com seus componentes de interação.

Validações dos protótipos

A consistência do projeto é confirmada através das sucessivas iterações, em que são apresentados protótipos para os usuários validarem. Através de testes, são identificados erros, ou a necessidade de se fazer ajustes no sistema. Ainda nesta fase, são iniciadas estratégias para a implementação do sistema. Em determinado momento, deve-se finalizar o escopo do projeto e o plano de implementação. Por fim, os resultados das interações entre desenvolvedores e usuários são incorporados ao projeto e ao plano de transição. Se tudo for aprovado, passa-se para a fase de construção rápida.

Banco de dados

Agora, com o ambiente de desenvolvimento concluído, a próxima etapa é o projeto de banco de dados, que deve ser construído conforme a estrutura de dados desenvolvida na fase de design do usuário. O sistema passa a integrar as funcionalidades de banco de dados com os componentes de interação visual. Aplicam-se, agora, os testes, que devem ser executados com dados que simulem situações reais e auxiliem na detecção de possíveis falhas, para que sejam devidamente tratadas, e na documentação do sistema. Cada componente do sistema e as funcionalidades são verificadas de acordo com os requisitos do usuário. Por fim, são iniciados os preparos para a fase de transição através dos planos de trabalho e de contingência.

Transição

Nesta última fase, a de transição, são feitos treinamentos para o usuário utilizar o sistema antes que seja colocado em produção. Em seguida, o sistema é colocado em produção e, por fim, instalado no cliente. Configurações de hardware, instalações de bibliotecas e demais configurações são concluídas nesta etapa. A última tarefa a ser terminada é a aceitação do sistema por parte do usuário, conforme o que foi estabelecido na etapa inicial do projeto.

Após a definição do escopo do projeto, as equipes iniciam a construção dos modelos e protótipos. O objetivo é demonstrar para o cliente um design funcional o mais rápido possível.

Observe os tópicos adiante:

- Desenvolvedores e designers – que desenvolvem telas e componentes interativos – trabalham em colaboração com os clientes até que o produto esteja pronto, para que os requisitos funcionais sejam atendidos. Essa etapa é repetida à medida que o projeto é construído.

- Durante a fase inicial de prototipagem, os desenvolvedores focam seus esforços em elementos essenciais do sistema para produzir um produto que seja aceitável pelo proprietário do produto.

- O uso de protótipos construídos rapidamente incentiva o envolvimento do usuário nos testes do sistema e, como consequência, são obtidos comentários que podem ser utilizados para aperfeiçoar o trabalho que está sendo executado, em vez de tentar fazer avaliações abstratas de um documento de design.

- Com esses comentários, os desenvolvedores podem ajustar os modelos de forma incremental, até que se atenda aos requisitos do projeto. A experiência compartilhada entre as partes interessadas é obtida através da boa comunicação. O aprendizado habilita a identificação rápida do que funciona e o que não funciona.

- A liberação rápida de protótipos aumenta as chances de descobrir erros precocemente, o que leva ao aumento da confiabilidade do sistema. Através da criação de protótipos, a equipe de desenvolvimento pode avaliar a viabilidade de componentes complexos. Consequentemente, aumenta-se a robustez do software, além de facilitar adições de design futuras.

O desenvolvimento rápido torna possível que protótipos evoluam para a versão comercial do sistema, ou seja, que as versões parciais progridam para o software que vai atender às expectativas do cliente. Através de incrementos ao longo das iterações, novos componentes e novas alterações são feitas. As equipes de desenvolvimento usam ferramentas computacionais que viabilizam o progresso rápido para a versão final do sistema. Na metodologia RAD, a maioria dos problemas e ajustes são tratados durante a fase de prototipagem iterativa.

Nas metodologias tradicionais, os desenvolvedores consomem muito tempo com atividades que não estão diretamente ligadas ao desenvolvimento do projeto. No caso da RAD, são feitos muitos testes, aumentando, assim, as chances de que o resultado satisfaça as expectativas do cliente. A colaboração entre desenvolvedores e usuários finais auxilia na construção de interfaces e funcionalidades que melhorarão todos os aspectos do produto. Os clientes fornecem informações detalhadas com sugestões de alterações – ajustes, ou novas ideias – que resolvem os problemas à medida que são descobertos.

Equipe de testes em reunião.

A metodologia RAD se baseia no desenvolvimento iterativo e incremental. Ela é dividida em fases, caracterizando-a, assim, como um método com definição de procedimentos que devem ser seguidos para se atingir a meta do projeto: atender às necessidades do cliente dentro de um prazo curto e sem erros, ou, pelo menos, com o mínimo possível de erros. A existência de mais de uma abordagem para tratar as suas fases não muda a essência do processo, que se caracteriza pela interação constante entre usuários e desenvolvedores.

Atividade 2

Considerando a importância de testar protótipos em cenários que simulam condições reais de uso, como descrito no ciclo de desenvolvimento RAD, por que é essencial realizar esses testes antes da implementação final do software?

A

Garantir que o software possa ser usado em diferentes sistemas operacionais.

B

Retardar a execução de custos de desenvolvimento, deixando os problemas para depois.

C

Aumentar a velocidade de desenvolvimento do software.

D

Atender às regulamentações de segurança de software específicas da indústria.

E

Avaliar a performance e a usabilidade do produto em condições reais, evitando falhas críticas após a implementação.

A alternativa E está correta.

Testar em condições reais ajuda a detectar falhas antes que o software seja amplamente distribuído, assegurando que o produto seja robusto e confiável no seu ambiente de uso pretendido.

Introdução à metodologia RAD

A eficiência no desenvolvimento de software é potencializada pela metodologia RAD, que exige entregas rápidas em um mercado tecnológico em constante evolução. O sucesso da RAD depende de sinergia entre profissionais qualificados, gerenciamento ágil e emprego de ferramentas computacionais. Nesse contexto, cresce a importância da qualificação e adaptabilidade dos profissionais, da autonomia decisória das equipes e do emprego de ferramentas CASE para otimizar o desenvolvimento de software. Adotar essas práticas assegura a conclusão de projetos com a agilidade e qualidade necessárias para atender às demandas do setor tecnológico.

Neste vídeo, exploramos os conceitos fundamentais de pessoas, gerenciamento e utilização de ferramentas CASE na aplicação da metodologia RAD. Confira!

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

A metodologia RAD tem por objetivo fazer a entrega dos sistemas em menos tempo e com menos erros do que os métodos tradicionais de desenvolvimento. No entanto, para implementar essa metodologia, as empresas precisam satisfazer algumas condições (Berger &; Beynon-Davies, 2009). Vejamos!

Pessoas

Profissionais qualificados e com rápida adaptação, trabalhando bem em equipe.

Gerenciamento

Equipes com poder de decisão para evitar perda de tempo, o que é comum nos modelos tradicionais.

Uso de ferramentas computacionais (CASE)

Programas que facilitem a criação de diagramas e interface com usuário, componentes reutilizáveis (APIs, frameworks e templates, por exemplo) e de fácil manutenção.

A aplicação da metodologia RAD gera sistemas com telas e componentes padronizados devido ao uso de ferramentas que utilizam bibliotecas e templates reutilizáveis. No entanto, aspectos como desempenho do sistema e análise de risco são menos tratados, pois são atividades que demandam tempo em qualquer projeto.

> A RAD é mais adequada para softwares de baixa complexidade.

É natural associar a metodologia RAD a uma metodologia ágil, e alguns autores fazem essa associação devido às muitas semelhanças entre as duas. Portanto, as vantagens e desvantagens de ambas seriam, em grande parte, idênticas. No entanto, existem algumas diferenças entre elas. Na RAD, há uma limitação em trabalhar com várias equipes simultaneamente, enquanto no desenvolvimento ágil isso é algo comum. Outro ponto é o cumprimento dos prazos. Na RAD, o comprometimento é com a rapidez e a qualidade das entregas nas iterações, na expectativa de que isso se reflita no projeto como um todo. Já nos métodos ágeis, há prazos a cumprir do ponto de vista global do projeto.

Atenção

A RAD nem sempre é adequada para ser aplicada a um projeto. Existem casos em que métodos tradicionais são mais pertinentes. Trata-se de uma metodologia que funciona muito bem sob certas circunstâncias e disponibilidade de recursos e que, em outros casos, não é recomendada.

Atividade 1

A utilização de ferramentas que auxiliem as equipes pode aumentar a qualidade e a velocidade do desenvolvimento de software, diminuindo a possibilidade de erros e a necessidade de mão de obra. Nesse contexto, qual é o papel principal das ferramentas CASE na implementação da metodologia RAD?

A

Monitorar o desempenho do sistema.

B

Facilitar a criação de diagramas e interfaces com o usuário.

C

Conduzir análises de risco detalhadas.

D

Gerenciar as comunicações entre equipes.

E

Automatizar testes de usabilidade.

A alternativa B está correta.

As ferramentas CASE são destacadas por simplificar a criação de diagramas e interfaces, o que é fundamental para a agilidade no desenvolvimento pela RAD.

Vantagens e desvantagens da RAD

A metodologia RAD oferece vantagens significativas para o desenvolvimento de software, como entregas rápidas e interação constante com o usuário, permitindo adaptações frequentes e feedback imediato. No entanto, também apresenta desafios, como a necessidade de equipes altamente qualificadas e a dificuldade em escalar projetos grandes.

Compreender essas vantagens e desvantagens é fundamental para avaliar se a RAD é apropriada para um projeto específico, garantindo que as decisões de desenvolvimento se alinhem às necessidades organizacionais e aos recursos disponíveis.

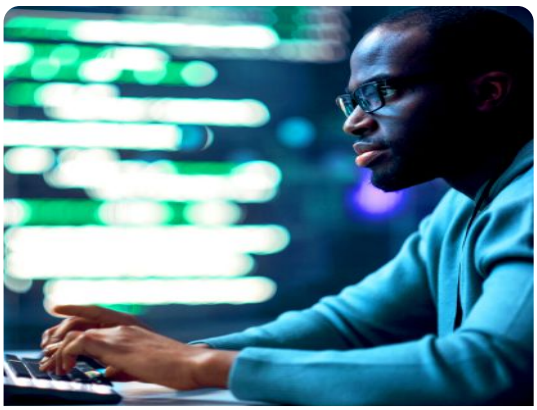
Este vídeo analisa as vantagens e desvantagens da metodologia RAD, destacando que seu desenvolvimento rápido e iterativo pode aumentar a eficiência, mas enfrenta desafios de escalabilidade e requer equipes muito qualificadas. Não deixe de conferir!

> Conteúdo interativo
>
> Acesse a versão digital para assistir ao vídeo.

O modelo RAD tem por objetivo a entrega rápida, pois o tempo total de desenvolvimento é reduzido devido à reutilização dos componentes e ao desenvolvimento paralelo. Para que funcione bem, a RAD precisa de profissionais qualificados e que o cliente também se comprometa a colaborar a fim de que os protótipos evoluam para o sistema desejado no prazo determinado. Caso não haja esse compromisso de ambos os lados, a metodologia poderá falhar.

Apesar da rapidez e qualidade da entrega serem as principais vantagens das RAD, também há desvantagens em relação à escalabilidade dos projetos e à demanda por recursos.

A rapidez da entrega é obtida através do uso de ferramentas que auxiliam a conversão de requisitos em código, permitindo que os desenvolvedores e usuários possam interagir através de protótipos que já são funcionais. Em relação à melhoria da qualidade, com a metodologia RAD, a qualidade será tão alta quanto a capacidade do software entregue de atender às necessidades dos usuários, necessitando de poucas intervenções para correção de erros, o que resulta em baixos custos de manutenção.



Desenvolvedor utilizando computador.

Portanto, a aplicação da RAD tem muitos benefícios para o desenvolvimento do projeto, em especial por integrar as equipes de software e seus clientes. Os times de desenvolvimento têm um aumento da sua produtividade, uma vez que rapidez e agilidade são prioridades. Isso melhora os resultados do projeto e torna possível que as entregas ocorram nos prazos estimados.

Principais vantagens

Conheça algumas das principais vantagens da RAD (Berger &; Beynon-Davies, 2009).

Integração antecipada do sistema e redução de riscos

A metodologia RAD permite a entrega rápida de protótipos funcionais, o que viabiliza revisões constantes das funcionalidades do projeto desde o início do ciclo de vida do software. Devido às iterações frequentes e ao feedback dos usuários, todos os aspectos do sistema podem ser reavaliados continuamente. Isso permite uma análise mensurável do progresso, garantindo que os cronogramas e orçamentos estejam no caminho certo. Enquanto nas metodologias tradicionais a integração com outros sistemas e serviços ocorre no final do ciclo de desenvolvimento, na RAD, os testes são realizados durante cada iteração. Isso permite que as partes interessadas identifiquem e resolvam rapidamente os erros e vulnerabilidades, evitando impactos no progresso do desenvolvimento. A redução de riscos resulta em uma diminuição de custos.

Adaptabilidade e compartimentação dos componentes do sistema

Durante o desenvolvimento, é mais fácil fazer modificações no software, tornando-o maleável enquanto o projeto ainda não está concluído. No entanto, essas alterações devem ser realizadas com cuidado, pois até mesmo pequenas mudanças podem afetar todo o sistema. Em geral, os desenvolvedores podem aproveitar essa flexibilidade para criar protótipos ao longo do processo. Devido à natureza iterativa da RAD, designers e desenvolvedores são incentivados a criar componentes funcionais, independentes e reutilizáveis.

Versões iterativas e menor tempo de colocação no mercado

O uso de ferramentas que suportam o desenvolvimento fortalece as equipes de desenvolvimento a entregar protótipos prontos, ou seja, utilizáveis, para produção mais rápida do que o desenvolvimento feito pelas metodologias tradicionais. As equipes devem ser pequenas, pois a ideia é que o trabalho “repetitivo” seja feito por essas ferramentas, pois, assim, ocorre o aumento da produtividade. As iterações frequentes incentivam a quebra das tarefas, o que é conhecido como “granularização”. Essas tarefas são atribuídas aos membros da equipe, conforme a especialidade e experiência de cada um.

Feedback constante do usuário

Trata-se de uma das principais características da RAD. A eficiência e a qualidade do projeto aumentam com a comunicação regular e o feedback constante dos usuários finais. A estrutura iterativa e o acesso aos componentes de UI / UX de um sistema aumentam ainda mais a importância do feedback dos usuários. O fato de os desenvolvedores terem a oportunidade de apresentar para os usuários os protótipos que construíram faz com que fiquem mais confiantes de que estão no caminho certo para satisfazer o cliente quando o produto final é entregue.

Uma das principais características da metodologia RAD é a entrega de protótipos funcionais. Exatamente por isso, a escalabilidade dos projetos é reduzida, pois a aplicação da metodologia sem adaptações inviabiliza a interação com o usuário para um sistema complexo.

A limitação do tempo de desenvolvimento das iterações é uma característica muito importante para fazer as entregas rápidas, porém é um limitador para a implementação de recursos mais avançados. Embora a RAD tenha diversos benefícios, há situações para as quais a metodologia não é adequada, como em projetos com risco técnico alto.

Principais desvantagens

Conheça agora algumas das desvantagens da RAD (Berger &; Beynon-Davies, 2009). Vamos lá!

Necessidade de equipes tecnicamente muito qualificadas

A metodologia RAD depende fortemente das habilidades de modelagem de dados e processos. Além disso, é necessário que os profissionais se adaptem rapidamente, pois a natureza da metodologia permite mudanças

significativas ao longo das iterações. Portanto, uma equipe bem qualificada é essencial para identificar e atender aos requisitos de negócios.

Foco exigente na interface

Os clientes avaliam a qualidade de uma solução com base na interação com o protótipo. A metodologia RAD é caracterizada por iterações nas quais novas funcionalidades são implementadas – incrementos – e, ao final de cada iteração, protótipos são entregues. Os clientes podem avaliar o progresso a cada novo lançamento. Devido à rapidez com que os protótipos são desenvolvidos, os desenvolvedores às vezes não aplicam as melhores práticas de projeto, focando principalmente atender às necessidades imediatas dos usuários. Isso pode gerar dívidas técnicas, o que pode causar problemas na entrega da versão final do sistema.

Alto nível de comprometimento das partes interessadas

Nos métodos de desenvolvimento tradicionais, os requisitos funcionais são definidos no início do projeto e não são revisados posteriormente, o que significa que clientes e equipes de desenvolvimento não interagem durante o projeto. Em contraste, na metodologia RAD, o entendimento do projeto evolui à medida que ele é desenvolvido, tornando a colaboração entre as partes interessadas essencial. Se essa colaboração não ocorrer, a qualidade do projeto pode ser seriamente comprometida.

Sistemas modulares para projetos de grande escala

A natureza da RAD, com suas entregas rápidas ao longo das iterações, exige foco nas partes essenciais do sistema. Como resultado, outros elementos importantes, como segurança, desempenho e tratamento de erros, podem ser comprometidos. Além disso, a necessidade de ferramentas padronizadas para entregas rápidas limita a flexibilidade do desenvolvedor. Embora essa característica seja vantajosa para projetos de baixa complexidade, para projetos de grande escala, que envolvem diversas tecnologias, a metodologia RAD não é adequada.

Comparação de RAD com outras metodologias

A metodologia RAD e as metodologias ágeis compartilham muitos pontos em comum. De fato, a RAD é uma das precursoras das metodologias ágeis, embora estas sejam mais abrangentes do que uma simples metodologia de desenvolvimento. No entanto, também existem diferenças, como podemos ver a seguir.

RAD	Outras metodologias
• Foco no desenvolvimento rápido com entregas frequentes de protótipos. • Trabalho feito com uma única equipe de desenvolvimento, com poucos membros. A ideia é melhorar a qualidade da comunicação e a transferência rápida de informações. • Envolvimento do usuário em todo o processo.	• Foca na entrega de um produto funcional ao cliente. • Trabalho feito com equipes maiores, divididas em diferentes especializações. Por exemplo, no modelo ágil, há equipes com papéis específicos. • Envolvimento dos usuários apenas no início e no final do ciclo de desenvolvimento do projeto.

Suas principais diferenças estão no estabelecimento de um cronograma de entregas, na importância dos comentários dos usuários para o projeto e no desenvolvimento focado em desenvolver protótipos do projeto – que é o caso da RAD – em detrimento do desenvolvimento de características do projeto – que é o caso das metodologias ágeis.

Atividade 2

Considerando a necessidade de adaptabilidade e velocidade no mercado de tecnologia atual, qual das seguintes opções melhor descreve uma das principais vantagens da metodologia RAD conforme discutido no documento?

A

Permite um desenvolvimento sem necessidade de feedback do usuário.

B

Reduz a necessidade de equipes tecnicamente qualificadas.

C

Minimiza a interação e a comunicação entre as partes interessadas.

D

Oferece entregas rápidas através de iterações frequentes e protótipos funcionais.

E

Elimina completamente o risco de erros no desenvolvimento.

A alternativa D está correta.

A RAD possibilita a entrega rápida de protótipos funcionais, permitindo revisões e ajustes contínuos, essenciais para acelerar o ciclo de desenvolvimento.

Aplicabilidade da metodologia RAD

Entender quando e como aplicar a metodologia RAD é essencial para maximizar a eficácia dos projetos de software. A RAD é ideal para situações que exigem um desenvolvimento rápido e iterativo, sendo especialmente vantajosa em projetos de pequena escala que podem ser modularizados e nos quais a interatividade do usuário é prioritária. No entanto, essa metodologia pode não ser adequada para sistemas complexos e não deve ser aplicada quando for necessária a integração com sistemas existentes.

Entenda neste vídeo quando usar e evitar a metodologia RAD no desenvolvimento de software. Confira as condições ideais, como projetos de pequena escala e alta interatividade com o usuário.

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Metodologia RAD – quando aplicar e quando não aplicar

A metodologia RAD é adequada para determinados tipos de desenvolvimento, por exemplo, quando a interatividade do front-end dos sistemas é uma característica muito importante em detrimento da complexidade do back-end (Berger &; Beynon-Davies, 2009). Observe!

Quando aplicar

A metodologia RAD é adequada para projetos de pequena escala com equipes otimizadas de quatro a oito pessoas. Para projetos de grande escala, a RAD pode ser aplicada, desde que sejam divididos em projetos menores e mais gerenciáveis.

Quando não aplicar

A RAD não é adequada para desenvolvimento de sistemas críticos em tempo real; sistemas de infraestrutura muito grandes e quando os requisitos funcionais precisam ser especificados detalhadamente ainda no início do projeto. Também não é adequada quando o sistema deve interagir com outros sistemas já existentes.

É essencial o envolvimento das partes interessadas para maximizar o entendimento do sistema. Para isso, as equipes precisam ter poderes para tomar decisões. A capacidade de tomar decisões reduz o tempo para realizar modificações sem que haja a necessidade de aprovação por outros níveis hierárquicos dentro da organização. Portanto, a comunicação eficaz no desenvolvimento entre as partes interessadas é bastante importante na RAD.

Nas oficinas de trabalho, em que ocorrem as reuniões entre usuários e desenvolvedores, são usadas ferramentas e técnicas de modelagem para confirmar e documentar o entendimento dos requisitos. Além disso, o desenvolvimento de protótipos auxilia de modo concreto a possibilidade de avaliar se o caminho escolhido para o desenvolvimento do sistema está correto através dos comentários dos usuários.

Condições para aplicação

De forma resumida, a metodologia RAD é adequada desde que as seguintes condições sejam satisfeitas. Vamos lá!

- Disponibilidade de profissionais experientes e comprometidos com um processo de desenvolvimento intensivo e contínuo.
- Comprometimento do cliente para participar efetivamente do desenvolvimento do projeto avaliando os protótipos e contribuindo com comentários que deem suporte para que os desenvolvedores avancem no desenvolvimento com aperfeiçoamentos e ajustes.
- Seu cliente está disposto a seguir cronogramas do projeto e um cronograma para conclusão do modelo? Todas as partes interessadas precisam estar presentes para aplicar efetivamente essa metodologia.
- É necessário que o sistema possa ser dividido em módulos.
- Também é necessário ter à disposição programas e infraestrutura adequada para aplicar a RAD.

Para a aplicação da RAD, o projeto, a estrutura tecnológica da organização e a própria cultura da empresa precisam estar adequados para que o desenvolvimento do sistema seja bem-sucedido.

A RAD funciona perfeitamente para sistemas que podem ser divididos em módulos. Em especial, esses são alguns exemplos para os quais a RAD se encaixa bem:

- Sistemas que podem ser modularizados.
- Aspectos de interatividade com o usuário UI/UX são muito importantes no projeto.
- Ter à disposição profissionais qualificados no uso de ferramentas adequadas ao desenvolvimento rápido; em especial, no uso de frameworks.

- Os clientes entendem a importância da interatividade com os desenvolvedores e têm a expectativa de receber protótipos ao longo do projeto.

- Desde o início do projeto, já é conhecido que haverá mudanças durante o processo de desenvolvimento.

Essas são algumas situações em que é esperado que a RAD funcionará bem. A RAD é uma metodologia de desenvolvimento com processos bem definidos, em que a colaboração entre usuários e desenvolvedores é fundamental para que os projetos tenham sucesso. A fim de que possa ser aplicada, algumas condições precisam ser satisfeitas. Além de possuir vantagens, a RAD possui desvantagens no sentido de que não é adequada para sistemas complexos e de grande escala.

Atividade 3

A metodologia de desenvolvimento rápido de aplicações (RAD) surgiu para suprir a necessidade de reduzir o tempo de entrega dos sistemas e a quantidade de erros. No entanto, para ser usada, ela deve satisfazer alguns critérios. Nesse sentido, selecione a opção correta sobre os critérios que devem ser satisfeitos para aplicar a metodologia RAD.

A

É adequada para projetos de infraestrutura de larga escala.

B

A complexidade dos projetos não é um fator impeditivo.

C

A RAD não é adequada para projetos de grande escala.

D

Projetos de banco de dados distribuídos satisfazem aos critérios para aplicabilidade da metodologia RAD.

E

A RAD é ideal para sistemas que necessitam de especificações detalhadas e imutáveis desde o início do projeto.

A alternativa C está correta.

A metodologia RAD é adequada para projetos de pequena e média escalas, nos quais as interações entre desenvolvedores e usuários são viáveis e auxiliam o entendimento e o desenvolvimento do sistema.

Introdução ao desenvolvimento com RAD

A utilização de ferramentas de desenvolvimento rápido é essencial para a eficácia da metodologia RAD, facilitando a interação entre desenvolvedores e usuários por meio de protótipos funcionais. Essas ferramentas permitem que os usuários testem e forneçam feedback em tempo real, enquanto os desenvolvedores adaptam o software para melhor atender às expectativas dos usuários.

Além disso, a capacidade de desenvolver e iterar rapidamente permite uma redução significativa no tempo de lançamento no mercado, uma vantagem fundamental em um ambiente de mercado altamente competitivo. Assim, ferramentas que suportam desenvolvimento ágil e reutilização de componentes são fundamentais para maximizar a produtividade e a satisfação do cliente.

Confira no vídeo como as ferramentas de desenvolvimento rápido fortalecem a metodologia RAD, permitindo prototipagem ágil e feedback imediato dos usuários. Veja como essas ferramentas reduzem o tempo de mercado e melhoram a interação entre usuários e desenvolvedores.

> Conteúdo interativo
>
> Acesse a versão digital para assistir ao vídeo.

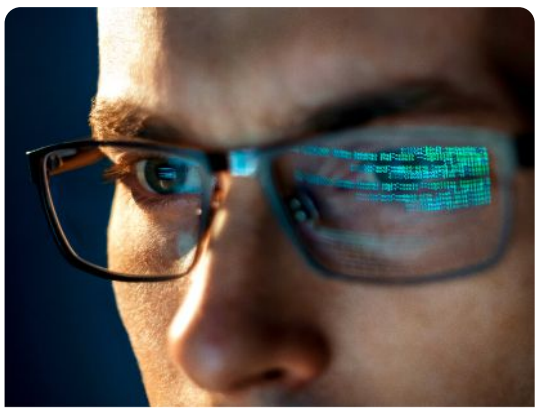
Uma das principais características da RAD é o desenvolvimento de protótipos, de modo que desenvolvedores e usuários possam interagir em um sistema funcional, acompanhe a seguir suas peculiaridades:

- Os usuários podem usar o software, ainda que tenha limitações, e fazer comentários e sugestões.
- Os comentários auxiliam os desenvolvedores a tomar decisões que maximizem a satisfação dos usuários.
- Com a validação do protótipo, os desenvolvedores montam um plano de trabalho para atender aos requisitos definidos pelos usuários.

Nesse sentido, fazer uso de ferramentas de desenvolvimento rápido é fundamental para que haja boa tradução entre ideias e representações visuais e funcionais.

A utilização de ferramentas que acelerem o desenvolvimento auxilia na redução do tempo de lançamento no mercado. No ambiente competitivo do mercado de software, trata-se de uma característica muito importante. Frameworks facilitam a codificação e padronizam o uso de recursos; desse modo, a “reusabilidade” de componentes é um dos benefícios que são obtidos e que podem ajudar no desenvolvimento de novos módulos do sistema.

A seguir, serão apresentadas características da linguagem Python, porém já pode ser adiantado que a sintaxe dela é bem mais simples do que muitas outras linguagens do mercado, como o Java, por exemplo. Portanto, escolher o Pyhton como linguagem de desenvolvimento de um projeto pode reduzir consideravelmente o tempo de produção. A ideia é simples: supondo que o número de linhas que um desenvolvedor pode produzir em determinado intervalo é limitado, escolher uma linguagem de sintaxe mais simples ajuda a aumentar a produtividade.



Desenvolvedor trabalhando com reflexo da tela em seu óculos.

Atividade 1

Considerando as características de entrega e feedback contínuos da metodologia RAD, qual é o impacto da escolha de uma linguagem de programação como Python para o processo de desenvolvimento?

A

Aumenta a complexidade do desenvolvimento devido à sua sintaxe complicada.

B

Reduz o tempo de desenvolvimento por ter uma sintaxe mais simples.

C

Não afeta o desenvolvimento, pois todas as linguagens são equivalentes na RAD.

D

Diminui a qualidade dos protótipos gerados.

E

Exige maior número de desenvolvedores para a implementação devido à sua popularidade.

A alternativa B está correta.

Escolher uma linguagem com sintaxe simples, como o Python, aumenta a produtividade ao permitir que desenvolvedores produzam mais em menos tempo.

Python

A linguagem Python, com sua sintaxe simples e vasta de pacotes e frameworks, possui uma excelente sinergia com os aspectos da metodologia RAD. Sua facilidade de uso acelera o desenvolvimento e a iteração de protótipos, permitindo que equipes respondam rapidamente às necessidades do mercado e aos feedbacks dos usuários. Python também oferece uma excelente integração com várias plataformas e sistemas, tornando-se uma escolha versátil para projetos de software que necessitam de agilidade e eficiência na entrega.

Este vídeo destaca a importância da linguagem Python para a RAD, enfocando sua sintaxe simples e eficácia em acelerar o desenvolvimento e a iteração de protótipos. Você verá como Python facilita a adaptabilidade e a colaboração entre desenvolvedores e usuários. Assista!

Conteúdo interativo

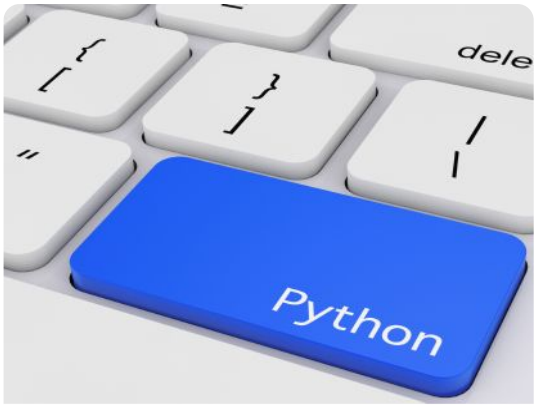
Acesse a versão digital para assistir ao vídeo.

Python é uma linguagem de programação muito usada para diversos fins. Ela possui muitos pacotes para diferentes tipos de aplicações, bibliotecas de GUI, além de ser uma linguagem de programação “enxuta”, no sentido de que sua sintaxe é mais simples do que muitas outras linguagens.

Dadas essas características, entre outras, o Python é uma linguagem adequada para a aplicação da metodologia RAD.

Outro ponto que fortalece a escolha do Python para projetos RAD é que muitos frameworks são baseados nela com o objetivo de acelerar o desenvolvimento e garantir desempenho e extensibilidade do código, pois é uma linguagem de tipo dinâmico, interpretada e de alto nível. Nela, é possível desenvolver programas estruturados, orientados a objetos e, até mesmo, aplicar conceitos de linguagens funcionais. Possui uma sintaxe que a diferencia de outras linguagens de programação, como Java, C ++ e Java Script.

Para fazer o desenvolvimento de um sistema em Pyhton, é necessário utilizar um ambiente de desenvolvimento, e um dos mais usados é o Spyder (Spyder, 2020), mas existem muitos outros.



Tecla Python ativada do teclado.

Instalação

A instalação dos pacotes no Pyhton também é bem simples, de modo geral. Normalmente, basta digitar: “pip install nome-do-pacote”.

Em algumas situações, pode-se ter um pouco mais de trabalho por causa da compatibilidade de versões. Essa questão de configurações de versões, inclusive, é um problema que abrange os projetos de desenvolvimento de modo geral e, sem o apoio de uma equipe focada em questões que incluam esse tipo de problema, perde-se bastante tempo no projeto.

Aqui estão os motivos para escolher o Python em projetos RAD, veja!

Portabilidade

Pode-se trabalhar com Python em diversos sistemas operacionais.

Licença de código aberto

As distribuições do Python são gratuitas e de código aberto sem restrições do uso da licença.

Integração com outros sistemas

Já existem muitos pacotes desenvolvidos para Python que abrangem aplicações de bancos de dados, web, interfaces gráficas, ciências de dados, entre muitos outros.

Desenvolvimento rápido

Por possuir uma sintaxe mais simples do que muitas outras linguagens de programação e, além disso, ter diversos pacotes e frameworks, o Python é uma linguagem muito apropriada para projetos RAD, em que a velocidade do desenvolvimento é considerada um fator fundamental.

Otimização de desempenho

O Python possui tratamentos específicos para manipulação de listas e de dados de grande porte que otimizam o desempenho do sistema, inclusive para aplicações de big data.

Linguagem interpretada

Torna mais simples fazer o uso interativo com o programa. No caso de desenvolvimento de protótipos, trata-se de uma característica prática.

Outra vantagem de usar o Python é que a comunidade de usuários é ampla, formada por pessoas com bastante conhecimento e prestativas. Uma das grandes preocupações de qualquer projeto é sobre o suporte que será fornecido ao longo do desenvolvimento; no caso do Python, a comunidade é ativa, ou seja, as pessoas realmente se esforçam para responder a perguntas de diversos níveis de conhecimento. O nível de suporte técnico disponível, gratuitamente, é considerável.

Dica

Para obter mais detalhes sobre a linguagem Python, visite a página Python 3.8.5 documentation (2020).

Atividade 2

Para que a implementação da RAD seja bem-sucedida, é necessário selecionar criteriosamente as ferramentas empregadas no projeto; em especial, a linguagem de programação utilizada. Nesse contexto, qual das seguintes características do Python é destacada como particularmente vantajosa para a metodologia RAD?

A

Python é uma linguagem de programação exclusiva para sistemas operacionais Windows.

B

Python limita desenvolvimento a aplicações web exclusivamente.

C

A comunidade Python é pequena e pouco ativa, oferecendo suporte limitado.

D

Python possui uma sintaxe simples que acelera o desenvolvimento e facilita a iteração de protótipos.

E

Python requer uso intensivo de ferramentas externas para gerenciamento de pacotes.

A alternativa D está correta.

A simplicidade da sintaxe do Python é ressaltada como uma vantagem para a RAD, pois permite um desenvolvimento mais rápido e eficiente, facilitando a criação e modificação de protótipos.

Ferramentas para o desenvolvimento RAD

A utilização de ferramentas (frameworks) Python é indispensável para profissionais de desenvolvimento que empregam a metodologia RAD, pois eles aceleram significativamente o tempo de entrega e padronizam o desenvolvimento dos sistemas. Frameworks como Django, Flask e PyQt, além de facilitarem a criação de protótipos e interfaces gráficas, melhoram a comunicação e interação com os usuários ao longo do processo de desenvolvimento. Essas ferramentas permitem uma construção rápida e eficiente, fundamental em um ambiente de desenvolvimento ágil, em que o tempo até o mercado e a capacidade de resposta rápida às mudanças são imprescindíveis.

Este vídeo mostra a importância dos frameworks Python para projetos RAD, destacando como aceleram o desenvolvimento e melhoram a interatividade com o usuário. Confira!

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

A linguagem Python em si – com seus diversos pacotes disponíveis para instalação gratuita – é uma escolha muito apropriada para desenvolver projetos RAD, mas, além disso, também possui diversos frameworks que auxiliam no ganho de velocidade do tempo de entrega, além de padronizar o desenvolvimento dos sistemas. Com uma interface gráfica obtida rapidamente, o usuário pode interagir com o desenvolvedor com mais qualidade, ou seja, facilita a comunicação entre as partes interessadas, além de manter a motivação no projeto.

Observe os tipos de frameworks suportados pelo **Python**.

Full-Stack
Fornece uma solução completa de todos os requisitos para o desenvolvedor, desde a geração e validação de formulários até a disponibilização de modelos layouts.

Microframework
Oferecem uma quantidade mínima de serviços. Normalmente, é usado para o recebimento de solicitações, roteamento, despacho e o retorno de uma resposta HTTP.

Framework assíncronos
Trata-se de um tipo de microframework que permite lidar com grande conjunto de conexões simultâneas. A estrutura assíncrona é usada para gerenciar operações de longa duração, como transformações de dados.

Ao longo dos últimos anos, o Python tem atraído mais desenvolvedores devido à facilidade de aprendizado e diversidade de aplicações que abrange. A combinação de Python e RAD permite desenvolver rapidamente protótipos que abrangem diversos tipos de aplicações, além do fato de que a aplicação poderá operar em múltiplas plataformas. A seguir, serão tratados alguns exemplos de frameworks em Python para desenvolvimento de GUIs e de aplicações para web.

Frameworks GUI para python

A interface gráfica do usuário (GUI) é um aspecto essencial para que possa haver interação com o sistema; portanto, o desenvolvimento de recursos interativos, ainda nos protótipos, possibilita que o usuário tenha uma percepção mais clara de como o projeto está progredindo. Existem diversos frameworks para Python que tratam desses tipos de recursos, tais como botões, ícones, campos de texto, gráficos, e assim por diante.

Veja a seguir alguns frameworks GUI para Python.

Tkinter

É uma biblioteca que já está embutida na instalação padrão do Python que permite desenvolver interfaces gráficas (TCL DEVELOPER XCHANGE, 2020).

PyQt

É uma biblioteca de componentes gráficos do Python. Ela não é instalada com o Python, além disso tem diferentes tipos de licença (PYQT, 2020).

PySide

É um software livre que executa nos sistemas operacionais Windows, Mac e Linux (PYSIDE, 2020).

Kivy

É um framework de código aberto, em que é possível desenvolver aplicativos “multi-touch” para dispositivos móveis e computadores (KIVY, 2020).

wxPython

É um kit de ferramentas de código aberto que possui suporte para as plataformas Windows de 32 bits, Unix e Mac OS X (WXPYTHON).

Cada um desses frameworks possui particularidades que podem torná-los mais adequados para determinados projetos. Em especial, o framework Tkinter é o framework GUI mais usado do Python.

Tkinter

O framework GUI mais usado do Python, possui componentes que podem ser organizados para facilitar a interatividade com os usuários. O Tkinter possui três classes para gerenciar a organização dos componentes. Vamos conhecê-las!

Método pack ( )

Organiza os componentes em blocos antes de posicioná-los no componente pai.

Método grid ( )
Faz a configuração dos componentes em forma de tabela.

Método place ( )
O posicionamento do componente é feito em um local específico.

Um dos motivos para o Tkinter ser muito usado é que ele já faz parte da biblioteca padrão do Python e é utilizado pelo framework web Django (framework web mais usado) para fazer páginas web. Portanto, não há necessidade de instalá-lo e, por ser usado pelo Django, torna mais fácil encontrar documentação e exemplos de como usá-lo.

Conheça agora os principais componentes do Tkinter.

Button

Trata-se de um componente botão. Para adicioná-lo no sistema, basta escrever:

```
w=Button(master,option=value)
```

Canvas

É usado para aplicar design e layouts complexos. Para adicioná-lo no sistema, basta escrever:

```
w=Canvas(master, option=value)
```

CheckButton

É aplicado para selecionar uma opção. Para adicioná-lo no sistema, basta escrever:

```
w=CheckButton(master, option=value)
```

Entry

Usado para entrar texto. Para adicioná-lo no sistema, basta escrever:

```
w=Entry(master, option=value)
```

Frame

É usado para agrupar e organizar componentes. Para adicioná-lo no sistema, basta escrever:

```
w=Frame(master, option=value)
```

Label

Exibe uma caixa onde se pode inserir texto. Para adicioná-lo no sistema, basta escrever:

```
w=Label(Master, option=value)
```

MenuButton

É usado para criar “menus de topo”. Para adicioná-lo no sistema, basta escrever:

```
w=MenuButton(Master, option=value)
```

Menu

Cria menu. Para adicioná-lo no sistema, basta escrever:

```
w=Menu(master, option=value)
```

Message

Usado para exibir múltiplas linhas sem texto editável. Para adicioná-lo no sistema, basta escrever:

```
w=Message(master, option=value)
```

Scale

É um componente deslizante que permite selecionar valores de uma escala específica. Para adicioná-lo no sistema, basta escrever:

```
w=Scale(master, option=value)
```

Todos esses componentes estão disponíveis no Tkinter.

Demais frameworks também possuem diversos recursos, portanto a escolha de um framework é uma decisão que deve ser feita no início do projeto.

Frameworks web para Python

É um conjunto de pacotes que habilitam os desenvolvedores a desenvolver aplicações para web, ou serviços, sem ter de implementar excesso de detalhes, como protocolos, soquetes ou gerenciamento de processos/ threads. A maioria dos frameworks para web focam o desenvolvimento de aplicações do lado do servidor. A responsabilidade pelas comunicações e pela infraestrutura ficam sob a responsabilidade do framework, permitindo que o desenvolvedor possa se concentrar na lógica do sistema.

Homem criando frameworks para web.

Os frameworks dão suporte para diversas atividades, como interpretar solicitações – obtenção de parâmetros de formulário, gerenciamento de cookies e de sessões –, produzir respostas (apresentação de dados, como o formato JSon, por exemplo) e fazer armazenamento persistente de dados. É característico de aplicações web que haja vários tipos diferentes de camadas de programação, geralmente empilhadas umas sobre as outras. Nesse sentido, os frameworks facilitam o rápido desenvolvimento de protótipos.

Em geral, um framework web consiste em um conjunto de bibliotecas e um arquivo principal utilizado para a programação. A maioria dos frameworks para aplicações web incluem padrões que devem ter os seguintes elementos. Vejamos!

Roteamento de URL

Trata de solicitações HTTP recebidas por parte específica do código Python.

Objetos de solicitação e resposta

Agrupa as informações recebidas, ou enviadas, para o navegador de um usuário.

Template Engine

São normalmente usados como um formato intermediário escrito por desenvolvedores para produzir um ou mais formatos de saída, normalmente, HTML, XML ou PDF.

Servidor web de desenvolvimento

Executa um servidor HTTP em máquinas de desenvolvimento. Quando os arquivos são atualizados, recarrega automaticamente o código do lado do servidor.

Existem frameworks web para Python diversos. Confira!

Django

É um framework de código aberto muito bem-sucedido para criar sites complexos baseados em dados. Possui modelos, bibliotecas e APIs que dão suporte na criação de projetos de desenvolvimento da web escaláveis (Django, 2020).

TurboGears

É um framework que usa a arquitetura MVC (Model View Control), que ajuda no rápido desenvolvimento de aplicações web (Turbogears, 2020).

Flask

É um framework que suporta o envio de solicitações REST (Flask, 2020).

Bottle

É um framework distribuído como um módulo de arquivo único, sem dependências além da biblioteca padrão do Python. Suporta o envio de solicitações com suporte a URL, bancos de dados de chave/valor e modelos e um servidor HTTP interno (Bottle, 2020).

CherryPy

É um framework que fornece as funcionalidades CRUD (Criar, Recuperar, Atualizar e Excluir) (CherryPy, 2020).

Falcon

É um framework para o desenvolvimento de programas de pequena escala; usa a arquitetura REST e tem disponível vários pacotes complementares para facilitar o desenvolvimento.

O Python possui muitos outros pacotes que servem para operações numéricas, manipulações de listas e impressão de gráficos que são usados, por exemplo, em aplicações de ciências de dados. Vamos conhecê-los!

Numpy

Com funções para operações matemáticas e lógicas em matrizes (NumPy, 2020).

Pandas

Aplicada, especialmente, na análise de dados (Pandas, 2020).

Matplotlib

É uma biblioteca de plotagem que auxilia na criação de gráficos e plotagens 2D, que incluem gráficos de barras, histogramas e muitos outros usando scripts Python.

A abrangência do Python é extensa com frameworks e bibliotecas que podem ser usadas para diversos tipos de aplicações.

A linguagem Python tem características que a tornam uma escolha adequada para aplicação em projetos RAD. Além de ter uma biblioteca e pacotes que cobrem diversos tipos de aplicações, também possui frameworks que são bem documentados e que facilitam a criação de protótipos.

Atividade 3

Dado o papel fundamental dos frameworks no desenvolvimento ágil e interativo de software utilizando a metodologia RAD, qual das seguintes opções contém um framework de desenvolvimento web que pode auxiliar na implementação da RAD?

A

Django

B

Numpy

C

Pandas

D

Kivy

E

PyQt

A alternativa A está correta.

Apesar de as alternativas citarem diversos frameworks Python que podem auxiliar projetos RAD, Django é o único framework web entre as alternativas.

Considerações finais

- Os conceitos e princípios da metodologia de desenvolvimento rápido (RAD).
- As principais ferramentas e técnicas da RAD.
- As fases da RAD.
- As situações em que se deve aplicar a RAD.
- O uso do Python em RAD.
- Os principais frameworks da RAD.

Explore +

Acesse o site oficial do Python para ler sobre conceitos da linguagem e de aplicações GUI, além de poder fazer downloads de diversos pacotes para desenvolvimento rápido.

Acesse o site oficial da IEEE e procure artigos que tratam do desenvolvimento rápido de software aplicado para web e internet das coisas.

Referências

BERGER, H.; BEYNON-DAVIES, P.: **The utility of rapid application development in large-scale, complex projects**. Information Systems Journal, v.19, n.6, p. 549-570, 2009.

bottlepy. Consultado na internet em: 4 ago. 2020.

cherrypy. Consultado na internet em: 4 ago. 2020.

Django. Consultado na internet em: 4 ago. 2020.

flask. Consultado na internet em: 4 ago. 2020.

Fitzgerald, B. **A Preliminary Investigation of Rapid Application Development in Practice, Proceedings of 6th International Conference on Information Systems Methodologies**, editors Wood-Harper AT, Jayarantna N., Wood J R G, p. 77-87, 1998.

Kerr, J.; Hunter, R. **Inside RAD**: How to Build Fully Functional Computer Systems in 90 Days or Less. New York: McGraw-Hill, 1994.

Kivy. Consultado na internet em: 4 ago. 2020.

Martin, J. **Rapid Application Development**, Macmillan, USA, 1991.

matplotlib. Consultado na internet em: 4 ago. 2020.

NAZ, R.; KHAN, M. N. A. **Rapid applications development techniques**: A critical review, Int. J. Softw. Eng. its Appl., v. 9, n 11, p. 163-176, 2015.

NumPy. Consultado na internet em: 4 ago. 2020.

Pandas. Consultado em meio eletrônico em: 4 ago. 2020.

PyQt. Consultado na internet em: 4 ago. 2020.

PySide. Consultado na internet em: 4 ago. 2020.

Python 3.8.5 documentation. Consultado na internet em: 4 ago. 2020.

Spyder. Consultado na internet em: 4 ago. 2020.

Tcl Developer Xchange. Consultado na internet em: 4 ago. 2020.

The Falcon Web Framework. Consultado na internet em: 4 ago. 2020.

turbogears. Consultado em meio eletrônico em: 4 ago. 2020.

wxPython. Consultado na internet em: 4 ago. 2020.