RAD (RAPID APPLICATION DEVELOPMENT)_anotations

quarta-feira, 28 de agosto de 2024 14:36

RAD (RAPID APPLICATI...

DEFINIÇÃO

Apresentação da metodologia de desenvolvimento rápido de software (RAD) e sua aplicação prática com o uso da linguagem de programação Python e frameworks para a aceleração do desenvolvimento.

PROPÓSITO

Compreender os conceitos, as fases e vantagens e desvantagens da RAD e apresentar a linguagem de programação Python e frameworks para RAD.

OBJETIVOS

MÓDULO 1

Descrever a contextualização, os conceitos, princípios, as ferramentas e técnicas da metodologia de desenvolvimento rápido de software (RAD)

MÓDULO 2

Identificar as fases da RAD

MÓDULO 3

Distinguir quando aplicar e quando não aplicar RAD

MÓDULO 4

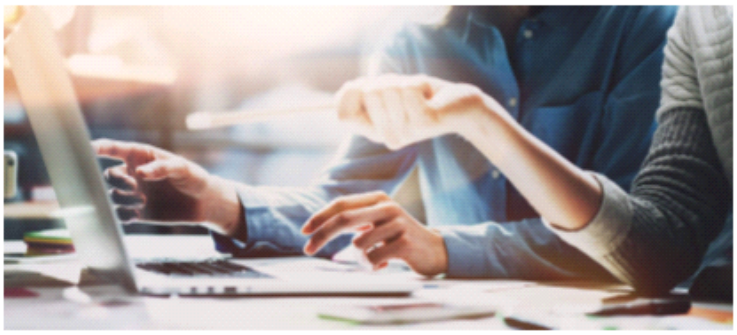
Justificar o Python e as ferramentas (framework) para o desenvolvimento RAD

INTRODUÇÃO

O desenvolvimento de software tem como objetivo atender as demandas da sociedade, cada vez mais complexas e com abrangência em diversas áreas. Logo nos primórdios da indústria de software, aplicavam-se metodologias que seguiam etapas que não eram revistas e, no final dos projetos, muitas vezes, desenvolvedores e clientes ficavam frustrados com o resultado obtido. Nesse sentido, a necessidade de criar formas mais eficazes de desenvolver sistemas levou à criação da metodologia rápida de desenvolvimento de software, mais conhecida pela sigla em inglês: RAD (Rapid Application Development). Baseia-se na entrega de protótipos para os clientes, a fim de que possam ter uma noção mais clara do progresso do desenvolvimento do software e que também possam colaborar com comentários que permitam aos desenvolvedores fazer alterações para atender as expectativas do cliente.

MÓDULO 1

⊙ Descrever a contextualização, os conceitos, princípios, as ferramentas e técnicas da metodologia de desenvolvimento rápido de software (RAD)



Autor: SFIO CRACHO / Fonte: Shutterstock

CONTEXTUALIZAÇÃO

O desenvolvimento rápido de aplicações RAD (Rapid Application Development) é uma metodologia de desenvolvimento de software com foco na entrega em um período muito inferior ao do ciclo de desenvolvimento tradicional de software. Não se trata de uma entrega final, mas, sim, de um **protótipo do software**. Para que isso seja possível, é feito um planejamento mínimo para obter um protótipo rápido.

Na metodologia RAD, existe uma concentração no desenvolvimento dos principais módulos funcionais do sistema. Essa versão inicial, que, apesar de limitada, já é funcional, é chamada de protótipo.

PROTÓTIPO DO SOFTWARE

Um protótipo de software é 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.

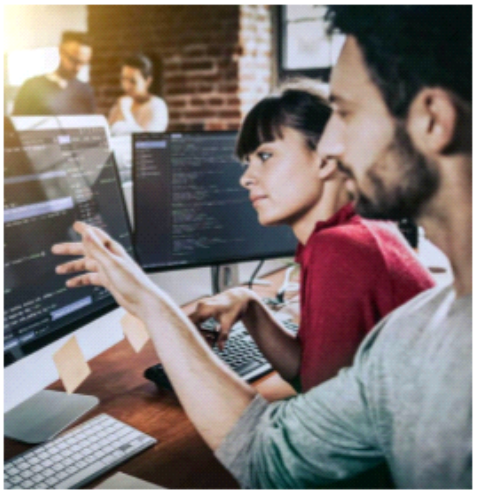
Na metodologia RAD, existe uma concentração no desenvolvimento dos principais módulos funcionais do sistema. Essa versão inicial, que, apesar de limitada, já é funcional, é chamada de protótipo.

PROTÓTIPO	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

Atenção! Para visualização completa da tabela utilize a rolagem horizontal

A RAD também pode ser aplicada para aperfeiçoar o treinamento prático de estudantes de computação, auxiliando-os em seus futuros empregos. Isso porque os estudantes podem aplicar o conhecimento adquirido nas aulas 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.



Autor: aurielaki / Fonte: Shutterstock

Os projetos RAD seguem o **modelo iterativo e incremental**. As equipes de desenvolvimento são pequenas, compostas por desenvolvedores, analistas de negócio e representantes de clientes. **Um dos aspectos mais importantes deste modelo é garantir que os protótipos desenvolvidos sejam reutilizáveis para o projeto do sistema, ou seja, a ideia não é criar unidades descartáveis**. Isso não contradiz o fato de o protótipo ser flexível.

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.

O RAD FOCA NO DESENVOLVIMENTO RÁPIDO POR MEIO DE ITERAÇÕES FREQUENTES E FEEDBACK CONTÍNUO.

VOCÊ SABIA

O modelo RAD foi introduzido pelo consultor de tecnologia e autor James Martin em 1991 (MARTIN, 1991). Surgiu como o reconhecimento da necessidade de atender o competitivo mercado de software, que tem uma demanda contínua por novas aplicações. Uma característica que foi 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	É muito prática 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.

Atenção! Para visualização completa da tabela utilize a rolagem horizontal

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

OBSERVE NO FLUXO COMO INICIAR UM PROJETO RAD:

UMA DAS FORMAS DE INICIAR O PROJETO RAD É ATRAVÉS DA APLICAÇÃO DA METODOLOGIA JOINT APPLICATION DEVELOPMENT (JAD).

TRATA-SE DE UMA METODOLOGIA NA QUAL USUÁRIOS E ANALISTAS PROJETAM O SISTEMA JUNTOS, SOB UMA LIDERANÇA EM OFICINAS DE TRABALHO.

A IDEIA É POTENCIALIZAR O RESULTADO DO DESENVOLVIMENTO ATRAVÉS DE DINÂMICAS DE GRUPO.

OU SEJA, DEFINIR OS OBJETIVOS E AS APLICAÇÕES DO SISTEMA, DESDE A GERAÇÃO DE TELAS ATÉ A GERAÇÃO DE RELATÓRIOS.

TEM COMO PRINCÍPIOS: DINÂMICA DE GRUPO; RECURSOS AUDIOVISUAIS; PROCESSO ORGANIZADO E RACIONAL; A ESCOLHA DO LOCAL; 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, por exemplo, 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.

Autor: astel design / Fonte: Shutterstock

CONCEITOS

Trata-se de 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:

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 (Timebox) para entregar iterativamente o sistema para os usuários.

TIMEBOX

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:

INTENSIVO

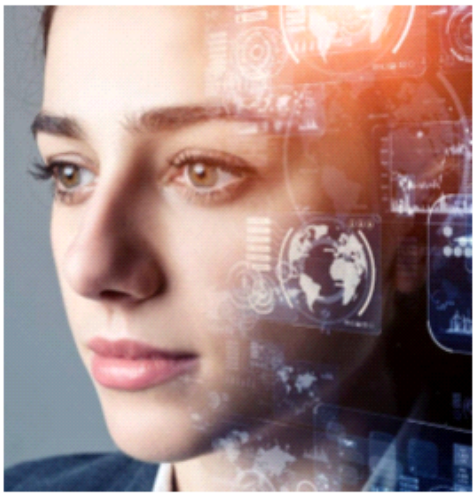
FASEADO

Como visto até aqui, está claro que 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.

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:

Quando usar o RAD é Depende da Natureza do Projeto, dimensão e TimeBox.



Autor: metamorworks / Fonte: Shutterstock

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.

O RAD É BASEADO EM ALGUNS PRINCÍPIOS BÁSICOS, QUE SÃO (FITZGERALD, 1998):

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

Outro princípio fundamental do RAD é que 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 no 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. Estes 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.

Os Requisitos podem mudar sem impactar o desenvolvimento

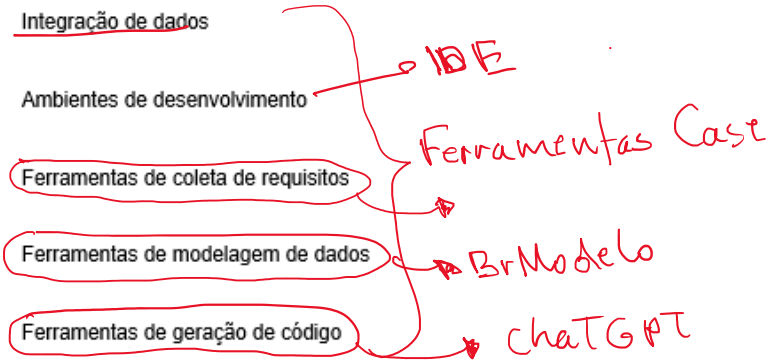
UTILIZAÇÃO DE FERRAMENTAS DE AUTOMAÇÃO (CASE)

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:



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. No quadro a seguir, conheça algumas dessas técnicas (Naz; Khan, 2015):

TÉCNICA	PARTICULARIDADE
Modelo CBD	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	Técnicas usadas para integração do ágil para tornar o projeto mais adaptável.
Método baseado em camadas para desenvolvimento rápido de software	Baseado em camadas que segue o **XP.**

Reaproveitamento de código

Análise de projeto de sistema baseado em simulação	Desenvolvimento de ferramentas ágeis baseadas em simulação.
Uso de Ajax no RAD	Prototipagem rápida em aplicativos e ferramentas da Web.
Desenvolvimento de aplicativos multiusuário em ambiente distribuído rapidamente.	Middleware de comunicação.
Programação extrema	Adição de reutilização ao XP.

Atenção! Para visualização completa da tabela utilize a rolagem horizontal

XP

Extreme Programming (XP) consiste em uma metodologia de desenvolvimento de software que tem como objetivo maximizar a qualidade do software e responder mais rapidamente às mudanças nos requisitos do cliente

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. Portanto, 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.

A pressão por soluções de software confiáveis e em curtos prazos favoreceu a criação da metodologia de desenvolvimento rápido de software (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.

VÍDEO COM AVALIAÇÃO

Assista no vídeo a abordagem do especialista sobre a aplicação do RAD em comparação com as metodologias ágeis existentes, diferenças e motivos para a escolha.

VERIFICANDO O APRENDIZADO

1. A JOINT APPLICATION DEVELOPMENT (JAD) É UMA METODOLOGIA DE DESENVOLVIMENTO QUE TEM COMO OBJETIVO MELHORAR O ENTENDIMENTO DO SISTEMA AINDA NO INÍCIO DO PROJETO. EM RELAÇÃO À METODOLOGIA JAD, SELECIONE A OPÇÃO QUE DESCREVE CORRETAMENTE UMA CARACTERÍSTICA DELA:

A) Desenvolvedores e usuários do sistema colaboram em oficinas de trabalho.

B) A documentação do sistema é feita detalhadamente, de modo a evitar ambiguidade do entendimento.

C) Desenvolve diversos protótipos em paralelo, de modo que o usuário possa selecionar entre eles qual o mais próximo dos requisitos funcionais.

D) Trabalha com componentes reutilizáveis de outras versões, de modo a minimizar o tempo de entrega dos protótipos.

2. A METODOLOGIA RAD TEM COMO OBJETIVO ENTREGAR O SISTEMA RAPIDAMENTE E COM QUALIDADE. DESDE QUE SURGIU, OUTRAS METODOLOGIAS EVOLUÍRAM A PARTIR DELA, MAS TODAS TÊM EM COMUM OS PRINCÍPIOS DO RAD. EM RELAÇÃO AOS TIPOS DE PROJETO RAD, SELECIONE A OPÇÃO QUE DESCREVE CORRETAMENTE UMA CARACTERÍSTICA DELA:

A) Existem dois tipos bem caracterizados de RAD: com e sem participação do usuário ao longo das iterações do projeto.

B) Em todos os tipos de RAD, é necessário que haja reuniões entre desenvolvedores e usuários com dedicação intensiva por curtos períodos, de modo a produzir unidades de sistema utilizáveis.

C) Como o RAD é uma metodologia de desenvolvimento flexível, o mais importante é que o desenvolvedor termine o software rapidamente.

D) O projeto é desenvolvido em etapas e com a inclusão de novas funcionalidades.

GABARITO

1. A Joint Application Development (JAD) é uma metodologia de desenvolvimento que tem como objetivo melhorar o entendimento do sistema ainda no início do projeto. Em relação à metodologia JAD, selecione a opção que descreve CORRETAMENTE uma característica dela:

A alternativa **"A "** está correta.

A principal característica da metodologia JAD são as oficinas de trabalho, em que desenvolvedores e usuários interagem e colaboram para o entendimento dos requisitos do sistema.

2. A metodologia RAD tem como objetivo entregar o sistema rapidamente e com qualidade. Desde que surgiu, outras metodologias evoluíram a partir dela, mas todas têm em comum os princípios do RAD. Em relação aos tipos de projeto RAD, selecione a opção que descreve CORRETAMENTE uma característica dela:

A alternativa **"D "** está correta.

A colaboração entre usuários e desenvolvedores ao longo do projeto é uma característica fundamental da RAD. O projeto é desenvolvido com interações e incrementos de funcionalidades. A operação dessa metodologia pode ocorrer em fases, ou de modo intensivo.

MÓDULO 2

- Identificar as fases da RAD

Autor: mrmohock / Fonte: Shutterstock

CONCEITOS

A metodologia RAD é caracterizada pelo desenvolvimento do projeto através de etapas iterativas e incrementais, onde 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.

Autor: fotoinfot / Fonte: Shutterstock

DESVENDANDO AS 4 FASES DA METODOLOGIA RAD

Neste vídeo, vamos explorar em detalhes as quatro fases essenciais da Metodologia RAD (Rapid Application Development). Você aprenderá como essa abordagem ágil transformou o desenvolvimento de software, acelerando a criação de aplicações e promovendo a colaboração eficaz entre equipes de projeto.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.

O ciclo de vida da RAD foi projetado para direcionar os desenvolvedores na criação de soluções de software que atendam às necessidades dos usuários. Este 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 (MARTIN, 1991), a metodologia RAD possui quatros fases distintas, que são:

FASE 1

FASE 2

FASE 3

FASE 4

PLANEJAMENTO DE REQUISITOS

As partes interessadas – usuários, gerentes e desenvolvedores – estudam as necessidades de negócios, escopo do projeto, restrições e 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, 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

É nesta fase que 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

São feitos processamento de dados, testes, mudança para o novo sistema e treinamento do usuário.

RESUMINDO

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, onde 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.

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, que são:

FASE 1

FASE 2

FASE 3

FASE 4

FASE 5

MODELAGEM DE NEGÓCIOS

Nesta fase, é feita a obtenção de informações a respeito dos requisitos funcionais do sistema reunidas por várias fontes relacionadas aos negócios, ou seja, o modelo de negócios do produto em desenvolvimento é tratado em termos de fluxo de informações, que são obtidas a partir de vários canais de negócios, tais como entrevistas com usuários do sistema e outras fontes de informação que auxiliem no entendimento do negócio que será tratado. Essas informações são então combinadas, para criar uma documentação que será utilizada para modelar o ciclo de vida dos dados: como os dados podem ser usados, quando são processados e a sua transformação em informações que serão utilizadas por alguma área, ou por algum setor específico do negócio. Portanto, é feita uma análise com viés comercial a fim de encontrar os dados vitais para os negócios: como podem ser obtidos, como e quando são processados e quais são os fatores que os transformam em informações úteis.

MODELAGEM DE DADOS

Todas as informações que foram obtidas durante a fase modelagem de negócios são analisadas para formar conjuntos de objetos de dados essenciais para os negócios. Através da análise, as informações são agrupadas, de modo que sejam úteis para a empresa, como, por exemplo, na determinação de quais são as entidades principais que serão tratadas pelo sistema. A qualidade de cada grupo de dados, então, é examinada e recebe uma descrição precisa. Em seguida, é feito mapeamento que relaciona esses grupos entre si e qual o significado desses relacionamentos, conforme definido na etapa modelagem de negócios. Avançando um pouco mais, agora deve-se identificar e definir os atributos de todos os conjuntos de dados. Também é estabelecida e definida em detalhes de relevância para o modelo de negócios a relação entre esses objetos de dados.

MODELAGEM DE PROCESSOS

Esta fase é a etapa da metodologia RAD em que todos os grupos de dados coletados durante a etapa modelagem de dados são analisados sob o ponto de vista de processamento, ou seja, como os dados são convertidos em informações úteis. Durante a fase de modelagem de processos, mudanças e otimizações podem ser feitas, e os conjuntos de dados podem ser definidos com mais detalhes. Quaisquer descrições para adicionar, remover ou alterar os objetos de dados também são criadas durante esta fase. A ideia é a seguinte: os conjuntos de objetos de dados definidos na fase de modelagem de dados são convertidos para estabelecer o fluxo de informações de negócios necessário para atingir objetivos de negócios específicos conforme o modelo de negócios. A modelagem de processamento dos dados para alterá-los, ou utilizá-los para fazer quaisquer outras operações que os transformem, é definida nesta fase. Aqui, também são feitas descrições dos processos para adicionar, excluir, recuperar ou modificar um objeto de dados.

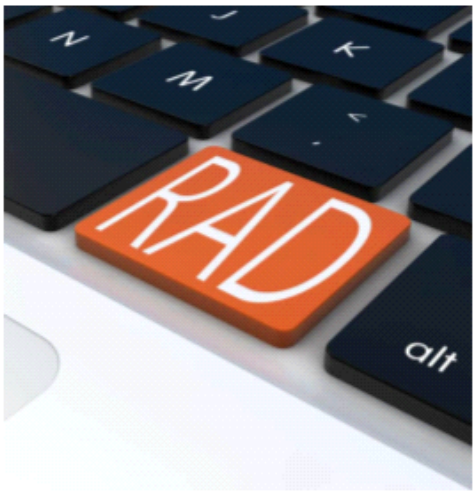
GERAÇÃO DA APLICAÇÃO

Nesta fase, 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

Neste momento, 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.



Autor: McLittle Stock / Fonte: Shutterstock

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. **Nesse sentido, 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.

UI/UX

O termo UI faz referência a “interface com o usuário” no sentido de telas e demais componentes interativos. Já o termo UX é usado para se referir à “experiência do usuário” com o sistema, ou seja, a melhor forma do usuário interagir com o sistema no aspecto ergonômico de como entrar com dados e obter informações.

CICLO DE DESENVOLVIMENTO

EXPLORANDO O CICLO DE DESENVOLVIMENTO DA METODOLOGIA RAD

Neste vídeo, adentramos no fascinante mundo do Ciclo de Desenvolvimento da Metodologia RAD (Rapid Application Development). Descubra como essa abordagem ágil revolucionou o desenvolvimento de software, promovendo velocidade e flexibilidade. Vamos explorar cada etapa desse ciclo, compreendendo como ele permite a entrega rápida de aplicações de alta qualidade.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.

O ciclo de desenvolvimento da metodologia RAD começa com as partes interessadas – usuários e desenvolvedores – definindo um conjunto de requisitos do projeto, que é equivalente ao que seria feito durante o escopo do projeto nos ciclos de desenvolvimento tradicionais. Esse estágio de planejamento é “rápido”, pois a ênfase da metodologia RAD está nas iterações de construção de protótipos que são

mais importantes para o sucesso final de um projeto. Todos os envolvidos no projeto – desenvolvedores, clientes, usuários e equipes de software – são levados a interagir entre si, de modo a determinar os requisitos do projeto e, em caso de problemas, quais estratégias serão aplicadas durante o desenvolvimento.

OS REQUISITOS INCLUEM:

- Metas
- Expectativas
- Cronogramas
- Orçamento

Espera-se que o cliente forneça a visão para o produto, e, em colaboração com outras partes interessadas, são realizadas pesquisas para finalizar os requisitos com a aprovação de cada parte interessada. É necessário que todos os interessados fiquem convencidos de que o entendimento do projeto está claro desde o início do ciclo de desenvolvimento, pois isso ajuda as equipes a evitar falhas de comunicação e erros dispendiosos.

📢 ATENÇÃO

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.

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**.

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.

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.

VÍDEO COM AVALIAÇÃO

Assista no vídeo a apresentação das fases do RAD exemplificando com um caso prático.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.

VERIFICANDO O APRENDIZADO

1. AS FASES DA METODOLOGIA RAD AUXILIAM NO ENTENDIMENTO DO PROJETO E, PORTANTO, NA CONSTRUÇÃO DE UM SISTEMA COM MENOS ERROS. NESSE SENTIDO, SELECIONE A OPÇÃO CORRETA SOBRE AS FASES DA METODOLOGIA RAD:

A) A coleta de dados pode ser feita em qualquer uma das fases da metodologia RAD, desde que seja necessário complementar alguma informação.

B) A fase de modelagem de processos inicia mediamente após a estruturação dos dados em objetos de negócio.

C) A modelagem de negócios está focada nos negócios da empresa cliente. A partir de uma visão holística, é feito um refinamento do entendimento até chegar aos requisitos do sistema.

D) Apesar de a RAD ser uma metodologia flexível, todas as fases devem ser bem documentadas; em especial, as transições entre elas.

2. INDEPENDENTEMENTE DA METODOLOGIA QUE SEJA UTILIZADA PARA DESENVOLVER UM SOFTWARE, SEMPRE SERÁ NECESSÁRIO EXECUTAR TESTES PARA VERIFICAR SE HÁ ALGUM ERRO E, DESSE MODO, TOMAR AS AÇÕES MAIS ADEQUADAS. EM RELAÇÃO À METODOLOGIA RAD, SELECIONE A OPÇÃO CORRETA SOBRE OS TESTES DE SOFTWARE:

A) Os testes sempre devem ser feitos no final do ciclo de uma iteração.

B) Os testes devem ser focados exclusivamente nas funcionalidades que foram acrescentadas na iteração.

C) Dada a flexibilidade da metodologia RAD, os testes podem ser executados em qualquer momento do ciclo de vida do desenvolvimento.

D) Testes de software envolvem a alocação de pessoas treinadas e outros recursos computacionais, além de consumirem bastante tempo, portanto só devem ser realizados quando houver um consenso entre desenvolvedores e usuários.

GABARITO

1. As fases da metodologia RAD auxiliam no entendimento do projeto e, portanto, na construção de um sistema com menos erros. Nesse sentido, selecione a opção CORRETA sobre as fases da metodologia RAD:

A alternativa **"B "** está correta.

As fases da metodologia RAD devem ser executadas em sequência, pois são complementares. No caso da modelagem de processos, é necessário que os dados já tenham sido organizados em objetos de negócio na fase de modelagem de dados.

2. Independentemente da metodologia que seja utilizada para desenvolver um software, sempre será necessário executar testes para verificar se há algum erro e, desse modo, tomar as ações mais adequadas. Em relação à metodologia RAD, selecione a opção CORRETA sobre os testes de software:

A alternativa **"A "** está correta.

Os testes de software são uma etapa essencial no ciclo de vida de desenvolvimento do software. No caso da metodologia RAD, os testes são executados ao final de cada ciclo iterativo. É fato que, de modo geral, os testes são caros, mas o que a metodologia RAD faz é reduzir o custo deles, uma vez que uma versão só começará se a anterior tiver sido aprovada; logo, existe a expectativa de que o sistema tenha poucos problemas para serem verificados e corrigidos.

MÓDULO 3

⦿ Distinguir quando aplicar e quando não aplicar RAD

Autor: Friends Stock / Fonte: Shutterstock

CONCEITOS

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 esta metodologia, as empresas precisam satisfazer algumas condições, que são (BERGER & BEYNON-DAVIES, 2009):

Autor: 4 PM production / Fonte: Shutterstock

PESSOAS

Profissionais qualificados e com rápida adaptação, além de trabalhar bem em equipe.

Autor: SFIO CRACHO / Fonte: Shutterstock

GERENCIAMENTO

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

Autor: REDPIXEL.PL / Fonte: Shutterstock

USO DE FERRAMENTAS COMPUTACIONAIS (CASE)

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

A aplicação da RAD gera sistemas cujas telas e demais componentes são padronizados, devido ao uso de ferramentas com bibliotecas e templates reutilizáveis. No entanto, algumas características, como desempenho do sistema e análise de risco, são menos tratadas, pois são atividades que demandam tempo em qualquer projeto. Portanto, **a RAD é mais adequada para softwares de baixa complexidade.**

Nesta parte do texto, é natural associar a RAD a uma metodologia ágil, inclusive alguns autores fazem essa associação, pois, de fato, há muitas semelhanças entre as duas metodologias e, portanto, as vantagens e desvantagens de ambas seriam idênticas. Entretanto, existem algumas diferenças entre elas. Na RAD, há uma limitação de trabalhar com várias equipes, enquanto no desenvolvimento ágil isso ocorre de modo normal. Outro ponto a ser considerado é o cumprimento dos prazos. Na RAD, o comprometimento é com a rapidez e qualidade das entregas nas iterações, na expectativa, é claro, de que isso se reflita para o projeto como um todo. No caso dos métodos ágeis, existem prazos a cumprir do ponto de vista global do projeto.

📢 ATENÇÃO

A RAD nem sempre é adequada para ser aplicada a um projeto, como será discutido mais adiante. Existem casos, inclusive, 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, como veremos.

METODOLOGIA RAD – VANTAGENS E DESVANTAGENS

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.**

Autor: G-Stock Studio / Fonte: Shutterstock

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. Sobre a rapidez da entrega, isso é obtido através do uso de ferramentas que auxiliam a conversão de requisitos em código, permitindo, assim, que os desenvolvedores e usuários possam interagir através de protótipos que já são funcionais. A respeito da melhoria da qualidade, com a RAD, será tão melhor quanto um software entregue que atende às necessidades dos usuários, onde são necessárias poucas intervenções para correção de erros, implicando, assim, em baixos custos de manutenção.

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. Agora, clique e saiba algumas das principais vantagens do uso de um método RAD (Berger & Beynon-Davies, 2009):

PRINCIPAIS VANTAGENS:

INTEGRAÇÃO ANTECIPADA DO SISTEMA E REDUÇÃO DE RISCOS

A entrega rápida de protótipos que funcionam por parte das equipes da RAD viabiliza que as empresas façam revisões das funcionalidades do projeto desde o início do ciclo de vida do software. Todos os aspectos que se referem ao sistema podem ser reavaliados, devido às iterações frequentes e pelos comentários e sugestões dos usuários, permitindo a análise do progresso dos trabalhos de forma mensurável, que determina se os cronogramas e orçamentos estão no caminho certo. A integração com outros sistemas e serviços é feita no final de um ciclo de vida de desenvolvimento, como ocorre nas metodologias tradicionais. Os testes são realizados durante cada iteração, de modo que as partes interessadas identifiquem e analisem os erros e tratem as vulnerabilidades, resolvendo-os rapidamente, para que não impactem no progresso do desenvolvimento. Redução no risco significa redução nos custos.

ADAPTABILIDADE E COMPARTIMENTAÇÃO DOS COMPONENTES DO SISTEMA

Durante o desenvolvimento, é mais fácil fazer modificações no software, ou seja, enquanto o projeto não é concluído, o software é maleável. É claro que alterações devem ser realizadas com cuidado, pois, por menor que possam parecer, podem afetar todo o sistema. De modo geral, os desenvolvedores podem usar essa flexibilidade para criar protótipos ao longo do processo. Dada essa característica iterativa, designers e desenvolvedores são incentivados a criar componentes funcionais e independentes que sejam 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. Este assunto será tratado um pouco mais adiante neste texto. 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. Portanto, embora a RAD tenha diversos benefícios, há situações para as quais a metodologia não é adequada. Por exemplo, a RAD não funciona bem em projetos onde o risco técnico é alto. A seguir, clique e saiba sobre algumas das desvantagens da RAD (BERGER & BEYNON-DAVIES, 2009):

PRINCIPAIS DESVANTAGENS:

NECESSIDADE DE EQUIPES TECNICAMENTE MUITO QUALIFICADAS

A RAD depende fortemente das habilidades de modelagem – de dados e processos –, e, além disso, é necessário que os profissionais tenham rápida adaptação, pois, devido à natureza da metodologia, é possível haver mudanças significativas ao longo das iterações. Portanto, uma equipe bem qualificada é essencial para identificar e entregar os requisitos de negócios.

FOCO EXIGENTE NA INTERFACE

Os clientes fazem a avaliação da qualidade de uma solução com base no que eles podem interagir em um protótipo. A RAD é caracterizada por iterações nas quais são implementadas novas funcionalidades – incrementos – e, ao final delas, são entregues protótipos. Os clientes podem avaliar os progressos a cada novo lançamento. Devido à rapidez com que os protótipos são implementados, às vezes, os desenvolvedores não aplicam as melhores práticas no projeto, preocupando-se, essencialmente, com o que deve ser entregue para satisfazer as necessidades imediatas do usuário. Isso pode gerar dívidas técnicas, o que pode implicar em problemas quando for entregar a versão final do sistema.

REQUER ALTO NÍVEL DE COMPROMETIMENTO DE TODAS AS PARTES INTERESSADAS

Nos métodos de desenvolvimento tradicionais, os requisitos funcionais são definidos no início do projeto e não são revisitados posteriormente; portanto, clientes e equipes de desenvolvimento não interagem ao longo do projeto. Já na RAD, o entendimento do projeto evolui à medida que é feito; logo, a colaboração entre as partes interessadas é essencial. Caso isso não ocorra, pode haver um comprometimento muito sério da qualidade do projeto.

REQUER SISTEMAS MODULARES E É DIFÍCIL PARA

PROJETOS DE GRANDE ESCALA

A natureza da RAD em fazer entregas rápidas ao longo das iterações demanda foco nas partes essenciais do sistema. Desse modo, outros elementos importantes, como questões de segurança, desempenho e tratamento de erros, por exemplo, podem ficar comprometidos. Outro ponto que deve ser considerado é que, para fazer entregas "rápidas" de protótipo, a RAD demanda ferramentas que são padronizadas, ou seja, o desenvolvedor tem pouca flexibilidade no desenvolvimento. Para projetos de baixa complexidade, essa é uma característica muito interessante, mas, para projetos de grande escala, em que há muitas outras tecnologias envolvidas, a RAD não é adequada.

COMPARAÇÃO DE RAD COM OUTRAS METODOLOGIAS

RAD

A primeira e mais importante diferença entre a RAD e as demais metodologias de desenvolvimento é a concentração dos esforços pelo desenvolvimento rápido com entregas constantes de protótipos.

Na RAD trabalha-se com uma única equipe de desenvolvimento, com poucos membros. A ideia é melhorar a qualidade da comunicação e a transferência rápida de informações.

Na RAD, é fundamental envolver o usuário em todo o processo de desenvolvimento.

OUTRAS METODOLOGIAS

Nos outros modelos, de modo geral, a preocupação principal é entregar um produto funcional para o cliente.

Enquanto em outras metodologias trabalha-se com equipes maiores divididas em diferentes especializações, como, por exemplo, o modelo para equipe ágil,

Nas metodologias tradicionais, usuários são envolvidos apenas no início e no final do ciclo de desenvolvimento do projeto.

Como foi apresentado no começo deste módulo, a RAD e as metodologias ágeis têm muitos pontos em comuns. De fato, a RAD é um predecessor das metodologias ágeis, embora essas últimas sejam mais abrangentes do que uma metodologia de desenvolvimento. 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.

METODOLOGIA RAD – QUANDO APLICAR E QUANDO NÃO APLICAR

A metodologia RAD é adequada para determinados tipos de desenvolvimento, como, 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).

QUANDO APLICAR

QUANDO NÃO APLICAR

QUANDO APLICAR

A RAD não é adequada para as seguintes situações: 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. Para complementar, a aplicação da RAD não é adequada quando o sistema deve interagir com outros sistemas já existentes.

QUANDO NÃO APLICAR

A RAD é adequada para projetos de pequena escala com equipes também otimizadas de quatro a oito pessoas. No caso de projetos de grande escala, caso seja aplicada a RAD, é necessário que sejam divididos em projetos menores e mais facilmente gerenciáveis. Além disso, é preciso que haja o

envolvimento das partes interessadas, de modo a maximizar o entendimento do sistema. Para isso, as **equipes precisam ter poderes para tomar decisões**.

EQUIPES PRECISAM TER PODERES PARA TOMAR DECISÕES

A capacidade de poder 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.

De forma resumida, a metodologia RAD é adequada desde que as seguintes condições sejam satisfeitas, clique para conhecê-las:

CONDIÇÕES PARA APLICAÇÃO

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 que não é adequada para sistemas complexos e de grande escala.

VÍDEO COM AVALIAÇÃO

Assista no vídeo a abordagem de casos práticos, exemplificando a aplicabilidade ou não da metodologia RAD.

VERIFICANDO O APRENDIZADO

1. A METODOLOGIA DE DESENVOLVIMENTO RÁPIDO DE SOFTWARE (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 os critérios para aplicabilidade da metodologia RAD.

2. A METODOLOGIA RAD POSSUI CARACTERÍSTICAS QUE SÃO CONSIDERADAS VANTAGENS EM RELAÇÃO AOS MÉTODOS TRADICIONAIS, COMO, POR EXEMPLO, O MÉTODO CASCATA. NESTE SENTIDO, SELECIONE A OPÇÃO CORRETA SOBRE AS VANTAGENS DA METODOLOGIA RAD:

A) Necessidade de equipe altamente qualificada.

B) Desenvolver componentes reutilizáveis.

C) Deve haver um esforço inicial da modelagem de dados que não pode ser modificada ao longo do projeto.

D) É adequada para sistemas complexos.

GABARITO

1. A metodologia de desenvolvimento rápido de software (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 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 desenvolvimento do sistema.

2. A metodologia RAD possui características que são consideradas vantagens em relação aos métodos tradicionais, como, por exemplo, o método cascata. Neste sentido, selecione a opção CORRETA sobre as vantagens da metodologia RAD:

A alternativa "B " está correta.

A colaboração entre usuários e desenvolvedores ao longo do projeto é uma característica fundamental da RAD. O projeto é desenvolvido com interações e incrementos de funcionalidades. A operação dessa metodologia pode ocorrer em fases, ou de modo intensivo.

MÓDULO 4

- Justificar o Python e as ferramentas (framework) para o desenvolvimento RAD

Autor: Rawpixel.com / Fonte: Shutterstock

CONTEXTUALIZAÇÃO

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

DESSA FORMA, 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.

E 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.

Autor: dencg / Fonte: Shutterstock

QUE OUTRA RAZÃO VOCÊ APRESENTA PARA A ESCOLHA DA LINGUAGEM PYTHON?

RESPOSTA

PYTHON

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 que serão tratadas um pouco mais adiante neste texto, o Python é uma linguagem adequada para a aplicação da metodologia RAD.

Linguagem Python	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.
	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.

Atenção! Para visualização completa da tabela utilize a rolagem horizontal

SAIBA MAIS

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.

ENTRE OS MOTIVOS PARA ESCOLHER O PYTHON EM PROJETOS RAD, ESTÃO:

PORTABILIDADE
LICENÇA DE CÓDIGO ABERTO
INTEGRAÇÃO COM OUTROS SISTEMAS
DESENVOLVIMENTO RÁPIDO
OTIMIZAÇÃO DE DESEMPENHO
LINGUAGEM INTERPRETADA

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.

SAIBA MAIS

Para obter mais detalhes sobre a linguagem Python, sugere-se visitar sua página eletrônica (Python 3.8.5 documentation, 2020).

FERRAMENTAS (FRAMEWORK) PARA O DESENVOLVIMENTO RAD

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 ABAIXO OS TIPOS DE FRAMEWORKS SUPORTADOS PELO PYTHON.

FRAMEWORKS (PYTHON)

1

FULL-STACK:

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

MICROFRAMEWORK:

São frameworks que oferecem uma quantidade mínima de serviços. Normalmente, são usados para o recebimento de solicitações, roteamento, despacho e o retorno de uma resposta HTTP.

2

3

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.

Autor: mrmohock / Fonte: Shutterstock

CLIQUE E VEJA A SEGUIR, ALGUNS FRAMEWORKS GUI PARA PYTHON:

TKINTER

PYQT

PYSIDE

KIVY

WXPYTHON

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. São elas:

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 – que é o 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. Entre os principais componentes do Tkinter, estão:

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.

🖌 RECOMENDAÇÃO

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

Um framework Web 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 no 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.

Autor: Rawpixel.com / Fonte: Shutterstock

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.

De modo geral, um framework Web é formado por um conjunto de bibliotecas e um arquivo principal, onde é feita, de fato, a programação. A maioria dos frameworks para aplicações Web incluem padrões que devem ter:

ROTEAMENTO DE URL

OBJETOS DE SOLICITAÇÃO E RESPOSTA

TEMPLATE ENGINE

SERVIDOR WEB DE DESENVOLVIMENTO

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.

SAIBA MAIS

Existem frameworks Web para Python diversos. Dentre eles, estão:

> Django: é um framework de código aberto que é 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, como:

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.

VÍDEO COM AVALIAÇÃO

O vídeo apresenta frameworks em Python e como utilizá-los no desenvolvimento de sistemas empregando a metodologia RAD.

Para assistir a um vídeo sobre o assunto, acesse a versão online deste conteúdo.

VERIFICANDO O APRENDIZADO

1. A METODOLOGIA RAD TEM POR OBJETIVO O DESENVOLVIMENTO RÁPIDO DE SOFTWARE. SELECIONE A OPÇÃO CORRETA QUE JUSTIFICA A ESCOLHA DO PYTHON PARA PROJETOS RAD:

A) Suporta os conceitos de orientação a objetos e, como se sabe, é o paradigma de programação mais adequado para desenvolvimento rápido de aplicações.

B) A capacidade de integrar-se com linguagens muito eficientes, como C, garante a velocidade do desenvolvimento.

C) Possui uma sintaxe enxuta com pacotes e frameworks que padronizam o desenvolvimento.

D) Sua integração com pacotes de segurança aumentam a proteção do sistema.

2. A ESCOLHA DE UM FRAMEWORK PARA UM PROJETO RAD DEVE LEVAR EM CONSIDERAÇÃO DIVERSAS CARACTERÍSTICAS. A MAIS IMPORTANTE DELAS É A CAPACIDADE DE ENTREGAR PROTÓTIPOS AO FINAL DE CADA ITERAÇÃO. NESSE SENTIDO, SELECIONE A OPÇÃO CORRETA QUE JUSTIFICA A ESCOLHA DE DETERMINADO FRAMEWORK PARA UM PROJETO RAD:

A) As funções que são disponibilizadas por suas bibliotecas, a documentação disponível e a linguagem de programação que é utilizada.

B) O uso das linguagens de programação mais modernas por serem tecnologicamente mais avançadas.

C) O tipo de licenciamento do framework e de suas bibliotecas e seus pacotes.

D) Possui componentes GUIs mais agradáveis para o usuário, estimulando, assim, a motivação e colaboração no projeto.

GABARITO

1. A metodologia RAD tem por objetivo o desenvolvimento rápido de software. Selecione a opção CORRETA que justifica a escolha do Python para projetos RAD:

A alternativa **"C "** está correta.

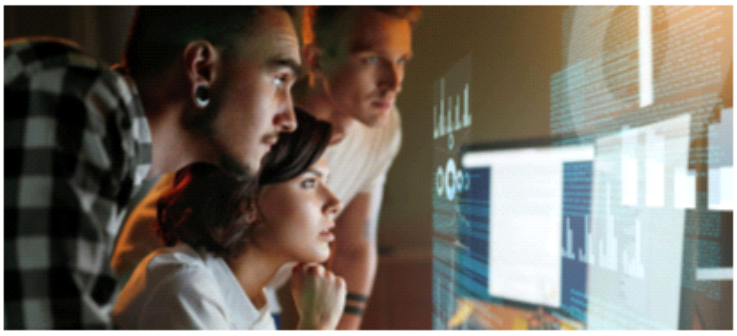
A linguagem Python possui uma sintaxe mais simples do que muitas outras linguagens de programação; além disso, dispõe de bibliotecas e frameworks que são bem documentados e facilitam o desenvolvimento.

2. A escolha de um framework para um projeto RAD deve levar em consideração diversas características. A mais importante delas é a capacidade de entregar protótipos ao final de cada iteração. Nesse sentido, selecione a opção CORRETA que justifica a escolha de determinado framework para um projeto RAD:

A alternativa **"A "** está correta.

Existem muitos frameworks disponíveis no mercado, e cada um tem características que os tornam mais adequados para um projeto RAD. A documentação certamente é fundamental para a escolha com exemplos de como usar o framework. Além disso, ter uma linguagem de programação com uma sintaxe mais simples facilita o desenvolvimento rápido de aplicações.

CONCLUSÃO



Autor: puhhha / Fonte: Shutterstock

CONSIDERAÇÕES FINAIS

Ao longo do texto, foram apresentados os conceitos da metodologia RAD, suas fases, vantagens e desvantagens e ferramentas – mais especificamente, frameworks – para o desenvolvimento rápido de software.

Como foi apresentado no tema, a RAD precisa que alguns requisitos sejam atendidos para que ela possa ser aplicada de maneira eficaz. O mais importante desses requisitos é a colaboração intensa do usuário, a fim de avaliar a qualidade do desenvolvimento do projeto através de comentários para os desenvolvedores.

Além disso, a linguagem Python foi descrita como uma escolha adequada para o desenvolvimento de projetos RAD, além de diversos frameworks que abrangem desde o desenvolvimento de interface gráfica com o usuário, aplicações Web, até aplicações específicas para Ciências de Dados.

Para ouvir um *podcast* sobre o assunto, acesse a versão online deste conteúdo.

REFERÊNCIAS

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

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

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

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

flask. Consultado em meio eletrônico 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, pp. 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 em meio eletrônico em: 4 ago. 2020.

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

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

NAZ, R.; KHAN, M. N. A. **Rapid applications development techniques**: A critical review, Int. J. Softw. Eng. its Appl., vol. 9, nº. 11, pp. 163–176, 2015.

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

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

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

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

Python 3.8.5 documentation. Consultado em meio eletrônico em: 4 ago. 2020.

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

Tcl Developer Xchange. Consultado em meio eletrônico em: 4 ago. 2020.

The Falcon Web Framework. Consultado em meio eletrônico em: 4 ago. 2020.

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

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

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.

CONTEUDISTA

Sérgio Assunção Monteiro

CURRÍCULO LATTES