Desenvolvimento Rápido de Aplicações em Python

Teresina, 06 de Novembro de 2022

Nome: Luan Felipe Silva Alves de Alencar

Matrícula: 202203241151

Resumo de Desenvolvimento Rápido de Aplicações em Python

Desenvolvimento Rápido de Aplicações ou Rapid Application Development (RAD) – o que é isso?

É um modelo de processo de desenvolvimento de software incremental.

Dentro da área de Engenharia de Software, RAD se encaixa nas metodologias ágeis.

Seu foco está na prototipação, para que haja rápido e constante feedback dos clientes/usuários → o que contribui para incrementar a aplicação.

Diferente do clássico Modelo em Cascata, o RAD se propõe a ser bem mais flexível.

A origem desse termo se deu em 1991, a partir de um livro publicado por James Martin.

Existem duas principais abordagens para o RAD. Uma proposta por James Martin e a outra proposta por James Kerr.

Fases por James Martin:

•FASE1/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.

•FASE2/DESIGN DO USUÁRIO; Nesta fase os desenvolvedores começam trabalhando com um protótipo. O objetivo é mostrar alguma coisa (funcionalidade) para o cliente o mais rápido e barato possível.

É importante lembrar que o protótipo não vai ter incluso todos os requisitos e cenários possíveis.

Assim que estiver pronto, o protótipo é apresentado ao(s) cliente(s)/usuário(s). Os feedback são coletados e então os requisitos podem ser ajustados.

A equipe de desenvolvimento continua nessa etapa de prototipagem até que o(s) cliente(s)/usuário(s) fiquem satisfeitos.

•FASE3/CONSTRUÇÃO; Nesta fase o time de desenvolvimento já sabe o que precisa ser feito (após os vários feedbacks do(s) cliente(s)/usuário(s) na fase anterior).

O sistema é desenvolvido com o foco em qualidade, escalabilidade e manutenabilidade. O(s) cliente(s)/usuário(s) ainda continuam participando, dando feedback à medida em que as features são implementadas.

É possível ter pequenos ajustes (fine tuning) nesta fase.

•FASE4/TRANSIÇÃO; Fase final, o que inclui testes (como o de aceitação), treinamento do usuário, deploy, etc.

Fases do RAD por James Kerr:

•FASE1/MODELAGEM DE NEGÓCIOS; “Nesta fase, serão definidos quais as características do negócio, isto é, define-se um fluxo de informações, que contém dados sobre: o que é gerado, quem gerou, para onde vão os dados e como é feito o processamento.”

•FASE2/MODELAGEM DE DADOS ; Nesta fase, o foco é sobre as funcionalidades. As informações extraídas na fase anterior são refinadas e, a partir disso, são extraídos os principais objetos de dados (e suas características) a serem processados pelo sistema.

•FASE3/MODELAGEM DE PROCESSOS; “Os objetos de dados definidos na modelagem de dados são transformados para com seguir o fluxo necessário para implementar uma função do negócio. Descrições do pro cessamento são criadas para adicionar, modificar, descartar ou recuperar um objeto de dados.”

•FASE4/GERAÇÃO DA APLICAÇÃO; Fase em que o time de programação utilizará de ferramentas que ajudem a acelerar o processo de desenvolvimento. Isto inclui reusabilidade de componentes, ferramentas automatizadas, etc.

•FASE5/TESTE E MODIFICICAÇÃO; Fase onde a aplicação passa pelos devidos testes.

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.

Devo usar RAD no meu projeto?

Sim, se

A aplicação possuir requisito muito bem definidos;

O sistema pode ser modularizado (reuso de código):

Pode-se fazer o uso de APIs;

O desempenho não é o mais importante;

O time de desenvolvimento é relativamente pequeno.

Não, se

O desempenho é essencial;

O produto for de larga escala e/ou são muitos os envolvidos;

O sistema não pode ser modularizado;

Não houver acesso regular aos usuários.

Frameworks (Python)

Manipulação de Arquivos em Python

#1 Abrindo arquivo existente

A função open(path, mode) recebe dois parâmetros: path e mode. O primeiro se refere ao caminho de onde está o arquivo a ser aberto. Já o segundo, se refere ao modo de acesso. A seguir, alguns modos de acesso:

- r : read / leitura

- w : write / escrita -> sobrescreve ou cria um novo

- r+ : read and write / leitura + escrita

- w+ : write and read / escrita + leitura

- a : append / acrescenta no fim

- a+ : append / acrescento no início

‘after seek’: significa que você pode definir o byte onde vai começar a operação. Por exemplo: arquivo.write(2).

‘truncate’: significa que o arquivo vai ser sobrescrito.

A função open() retorna o arquivo pedido, com o respectivo modo de acesso. Por isso é necessário uma variável para ‘receber’ esse arquivo. No nosso caso, o nome da variável é ‘arquivo’, mas poderia ser qualquer outra palavra, desde que não seja alguma já reservada (exemplo de palavra reservada: if). A partir de então, é possível manipular o arquivo a partir da variável.

É interessante sempre escrever logo o ‘.close()’, para evitar problemas. Caso a execução seja interrompida, ou finalizada normalmente, sem que o arquivo seja fechado, você terá problemas para acessá-lo, mesmo com outro programa, como o bloco de notas.

#2 Abrindo um arquivo existente com o modo de acesso ‘append’. A partir disso é possível acrescentar conteúdo ao arquivo.

#3 Caso o modo de escrita seja ‘selecionado’, e o arquivo pedido não exista, então ele será criado. É importan te lembrar também que, caso o arquivo já exista, com este modo de acesso, o coonteúdo dele será sobrescrito.

#4 A remoção de arquivos pode ser feita com a importação do módulo ‘os’ (de operating system, ou sistema operacional). Esse módulo provê uma interface para funcionalidades dependentes do sistema operacional.

#5 Da mesma forma, ou seja, com o uso do módulo ‘os’, é possível criar e remover pastas. Entretanto, a remoção só é possível se a pasta estiver vazia.

PYTHON COM BANCO DE DADOS:

DESCRIÇÃO; Desenvolvimento de aplicações para armazenamento e recuperação de informação em banco de dados utilizando a linguagem de programação Python.

CONCEITOS; Conectores para banco de dados; Quando estamos trabalhando com banco de dados, precisamos pesquisar por bibliotecas que possibilitem a conexão da aplicação com o banco de dados que iremos utilizar.

Na área de banco de dados, essas bibliotecas se chamam conectores ou adaptadores.

Como exemplo de conectores que implementam a DB-API 2.0, temos:

Psycopg2: Conector mais utilizado para o banco de dados PostgreSQL.

Mysqlclient, PyMySQL e mysql-connector-python: Conectores para o banco de dados MySQL.

Sqlite3: Adaptador padrão do Python para o banco de dados SQLite.

FLUXO BÁSICO

1/Criar uma conexão com o banco de dados com a função connect.

2/Utilizar a conexão para criar um cursor, o qual será utilizado para enviar comandos ao banco de dados.

3/Utilizar o cursor para enviar comandos ao banco de dados.

4/Efetivar as mudanças utilizando o método commit da conexão.

5/Fechar o cursor e a conexão.

PRINCIPAIS MÉTODOS DOS CONECTORES EM PYTHON:

CONNECT; Função global do conector para criar uma conexão com o banco de dados.

Retorna um objeto do tipo Connection.

CONNECTION; Classe utilizada para gerenciar todas as operações no banco de dados.

•Principais métodos:commit: Confirma todas as operações pendentes;

°rollback: Desfaz todas as operações pendentes;

°cursor: Retorna um objeto do tipo Cursor;

°close: Encerra a conexão com o banco de dados.

Principais Exceções;

Error : classe base para as exceções.

DatabaseError : exceção para tratar erros relacionados ao banco de dados. Subclasse de Error.

OperationalError: exceção para tratar erros relacionados a operações no banco de dados, mas que não estão sob controle do programador, como desconexão inesperada. É uma subclasse de DaDatabaseError

IntegrityError : exceção para tratar erros relacionados à integridade do banco de dados, como falha na checagem de chave estrangeira e falha na chegagem de valores únicos. É uma subclasse de DatabaseError.

ProgrammingError : exceção para tratar erros relacionados à programação, como sintaxe incorreta do comando SQL, tabela não encontrada, etc. É uma subclasse de DatabaseError.

NotSupportedError : exceção para tratar erros relacionados a operações não suportadas pelo banco de dados, como chamar o método rollback em um banco de dados que não suporta transações. É uma subclasse de DatabaseError.TIPOSS DE DADOS;

Cada dado armazenado em um banco de dados contém um tipo, como inteiro, texto, ponto flutuante, entre outros.

NULL: Para valores nulos.

INTEGER: Para valores que são números inteiros, com sinal.

REAL: Para valores que são números de ponto flutuante.

TEXT: Para valores que são texto (string).

BLOB: Para armazenar valores exatamente como foram inseridos, ex. bytes.

TIPOS DE DADOS:

Numeric, Monetary, Character, Binary, Date/Time, Boolean, Enumerated, Geometric, Network Address, Bit String, Text Search, UUID, XML, JSON, Arrays, Composite, Range, Domain, Object Identifier, Pg_lsn, Pseudo.

Alguns tipos de dados nativos de propósito geral

Nome	Aliases	Descrição
bigint	int8	inteiro de 8 bytes com sinal
boolean	bool	valor lógico (true/false)
Character [(n)]	char [(n)]	string de tamanho fixo
Character variando [(n)]	varchar [(n)]	string de tamanho variável

date		data de calendário (ano, mês, dia)

CONECTANDO A UM BANCO DE DADOS:

Como o SQLite trabalha com arquivo e não tem suporte à autenticação, para se conectar a um banco de dados SQLite, basta chamar a função connect do módulo sqlite3, passando como argumento o caminho para o arquivo que contém o banco de dados.

Veja a sintaxe a seguir

>>> import sqlite3

>>> conexao =

sqlite3.connect(‘meu_banco.db’

)

Pronto! Isso é o suficiente para termos uma conexão com o banco de dados meu_banco.db e iniciar o envio de comandos SQL para criar tabelas e inserir registros.

Caso o arquivo não exista, ele será criado automaticamente! O arquivo criado pode ser copiado e compartilhado.

Se quisermos criar um banco de dados em memória, que será criado para toda execução do programa, basta utilizar o comando conexao = sqlite3.connect(‘:memory:’).

INTERFACE GRÁFICA COM PYTHON:

CCONCEITO; A linguagem Python possui muitos frameworks para desenvolvimento de aplicações de interface gráfica para interação com o usuário, chamadas, comumente, de GUI (Graphical User Interface).

TKINTER; É o framework GUI padrão do Python. Sua sintaxe é simples, possui muitos componentes para interação com o usuário. Além disso, seu código é aberto e é disponível sob a licença Python. Caso ela não esteja instalada na sua versão do Python, basta digitar o comando:

pip install tkinter

Para quem usa a IDE Spyder, ver Spyder (2020), é necessário colocar uma exclamação antes do comando “pip”, ou seja, o comando, nesse caso, ficaria:

!pip install tkinter

Para testar a instalação, basta escrever na linha de comando o código abaixo:

import tkinter

tkinter._test()

Widgets Tkinter

São vários os widgets (ou componentes) do Tkinter. Os principais são:

- Botão (Button): É usado para exibir os botões na aplicação. São usados, por exemplo, para confirmar uma ação de salvar os dados.
- Telas (Canvas): É usado para desenhar formas, como linhas, ovais, polígonos e retângulos.
- Botão de Verificação (Checkbutton): É usado para exibir várias opções como caixas de seleção. O usuário pode selecionar várias opções ao mesmo tempo.
- Entrada de texto (Entry): É usado para exibir uma caixa de texto de linha única para que o usuário digite valores de entrada.
- Rótulo (Label): É usado para fornecer uma legenda de linha única para outros widgets. Também pode conter imagens.
- Caixa de listagem (Listbox): É usado para fornecer uma lista de opções para um usuário.
- Menubutton: É usado para exibir opções no menu.
- Menu: É usado para fornecer várias possibilidades de comandos a um usuário. Esses comandos estão contidos no Menubutton.
- Mensagem (Message): É usado para exibir uma mensagem de texto e um botão para o usuário confirmar uma ação.
- Botão de rádio (Radiobutton): É usado para exibir várias opções, como botões de rádio. O usuário pode selecionar apenas uma opção por vez.
- Escala (Scale): É usado para fornecer um widget de controle deslizante.
- Barra de rolagem (Scrollbar): É usado para adicionar capacidade de rolagem a vários widgets.
- Texto (Text): É usado para exibir texto em várias linhas.
- Toplevel: É usado para fornecer um contêiner de janela separado.
- Spinbox: É uma variante do widget Entry padrão. Ele é usado para selecionar um número fixo de valores.
- PanedWindow: É um widget de contêiner que pode conter qualquer número de painéis, organizados horizontalmente ou verticalmente.
- LabelFrame: É um widget de contêiner simples. Seu objetivo principal é atuar como um espaçador, ou contêiner para layouts de janela.
- tkMessageBox: Este módulo é usado para exibir caixas de mensagens.

APLICANDO RAD:

DESCRIÇÃO; Aplicação da metodologia de desenvolvimento rápido de software (RAD) com descrição do levantamento de requisitos, modelagem de negócios e de dados, Design de interface com o usuário e uma aplicação prática com o uso da linguagem de programação Python.

PROPÓSITO

Compreender a aplicação da metodologia RAD para o desenvolvimento de um sistema.

INTRODUÇÃO

A metodologia de desenvolvimento rápido de software (RAD) é caracterizada por um processo evolutivo em que usuários e desenvolvedores são engajados através de entregas de versões funcionais do sistema que, apesar de incompletas, permitem que a colaboração seja otimizada.

Os usuários podem operar essas versões e fazer comentários, críticas e sugestões que auxiliam bastante os desenvolvedores a direcionar seus esforços para atender, de fato, às necessidades do cliente. Essas versões parciais são chamadas de protótipos e a ideia é que, ao longo do processo, avancem no sentido de se tornarem a versão final do sistema.

Aspectos como custo-benefício, uso efetivo de tempo, alcançar a satisfação do cliente e controlar riscos são abordados pela RAD através da rápida verificação de inconsistências e alinhamento de entendimento.

Ao longo do texto, abordaremos as etapas de requisitos de um sistema, as modelagens de negócios e de dados e o design da interface com o usuário. No último módulo, apresentaremos uma aplicação RAD simples desenvolvida com o uso da linguagem de programação Python.

FASE DE PLANEJAMENTO DE REQUISITOS;

- Elicitação: Identifica os requisitos do sistema.
- Análise: Elenca os elementos necessários para tratar os requisitos
- Documentação: Comunica as partes envolvidas sobre o entendimento dos requisitos.
- Validação: O que foi implementado deve ser o que foi solicitado.
- Gerenciamento: Consiste na priorização dos requisitos.

ELICITAÇÃO DE REQUISITOS;

A elicitação de requisitos, ou ainda, levantamento de requisitos, tem por objetivo identificar os requisitos do sistema e em qual contexto será feita a operação deste com o apoio das partes interessadas. Por contexto, entende-se: Qual é o domínio da aplicação, quais as necessidades do negócio, quais são as restrições do sistema, a quem se destina e quais os pormenores da operação, a fim de obter uma melhor compreensão do sistema a ser desenvolvido.

Entrevistar as partes interessadas é um método para descobrir fatos e opiniões de usuários em potencial e evitar mal-entendidos que podem comprometer o desenvolvimento do sistema. As entrevistas podem ser de dois tipos:

- Fechadas: As partes interessadas respondem um conjunto predefinido de perguntas.

- Abertas: O engenheiro de requisitos e as partes interessadas discutem o que esperam do sistema.

MODELAGEM (NEGÓCIOS E DADOS);

A modelagem de sistema formaliza como os sistemas são definidos. É comum que, durante o desenvolvimento, sejam usadas imagens para ajudar a visualizar alguns aspectos do sistema. Através de representações por meio de diagramas, a modelagem fornece um meio para compreender e comunicar as ideias associadas ao desenvolvimento do sistema.

A RAD é uma metodologia de desenvolvimento de software evolutiva que, além do levantamento de requisitos, visto no Módulo 1, possui outras fases. De modo resumido, segundo Kerr e Hunter (1994), elas podem ser descritas como:

- Modelagem de negócios: Nessa fase, é feita a obtenção de informações a respeito dos requisitos funcionais do sistema, reunidas por várias fontes relacionadas aos negócios. 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 setor específico do negócio. Trata-se, portanto, de uma análise com viés comercial.
- Modelagem de dados: Nessa fase, todas as informações que foram obtidas durante a fase anterior 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. 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 precedente. 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.

  Ps: A modelagem não é um método exato e exige muita atenção do analista, pois vai ter impacto para todo o projeto. Quanto melhor for o entendimento do sistema, melhor será a qualidade da modelagem e, portanto, mais eficiente será a tradução das demandas do cliente para a implementação do sistema. Os modelos e as informações são representados por meio de diagramas conectados.

MAIS ALGUMAS CONSIDERAÇÕES SOBRE A MODELAGEM DE DADOS E DE NEGÓCIOS;

O objetivo principal de um modelo de dados é oferecer meios para que os objetos de dados sejam representados com precisão. Para isso, eles devem ser detalhados o suficiente para ser aplicados na construção do banco de

dados físico. As informações obtidas a partir deles podem ser usadas para definir quais são as chaves primárias e estrangeiras das tabelas, o relacionamento entre elas, além de outras informações que serão úteis para fazer manipulação e manutenção dos dados.

A modelagem de dados é um importante instrumento para a comunicação entre desenvolvedores e clientes, além de ajudar a documentar os mapeamentos de dados, comumente utilizados no processo de extração-transformação-carga de dados (ETL). Por fim, a modelagem de dados identifica fontes corretas de dados para preencher o banco de dados que será utilizado pelo sistema.

Tanto a modelagem de dados, como a modelagem de negócios são etapas fundamentais na RAD para compreender corretamente como o negócio funciona, como deve ser formalizado e como os dados devem ser estruturados de modo que sejam adequadamente armazenados e estejam disponíveis para serem usados quando houver necessidade.

CONSIDERAÇÕES FINAIS;

Os benefícios da aplicação da RAD têm impacto desde a melhor utilização do tempo das partes envolvidas até a minimização dos riscos do projeto, através de um processo colaborativo. A escolha da linguagem Python para projetos RAD é natural devido à vasta disponibilidade de documentação e de pacotes de componentes gráficos e de manipulação de dados que facilitam o desenvolvimento.