Engenharia de Software e QA básico em Python

Carlos Tavares

10 de Dezembro de 2024

Engenharia de Software - "Os processos, atividades e ferramentas que conduzem à entrega de software."

O principal objetivo: Entregar software correto a tempo!

Programação vs Engenharia de Software

A primeira diz respeito à escrita de código correto que encaixa numa especificação. A segunda inclui uma ampla gama de atividades, maioritariamente envolvendo pessoas (desenvolvedores e clientes) para fazer programas muito complexos ganharem vida com qualidade.

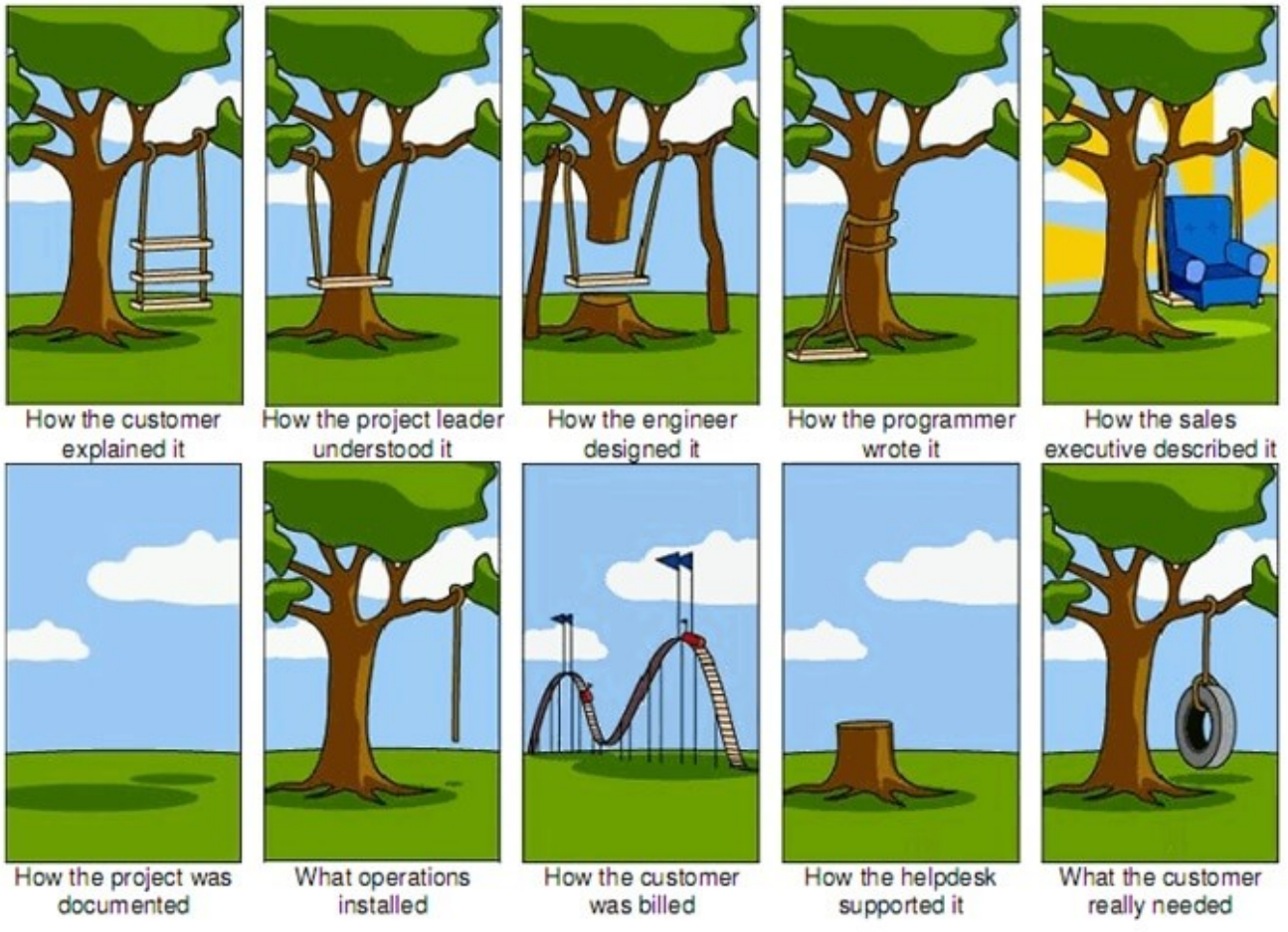
Subprocessos de Engenharia de Software i

Existem cinco atividades principais num processo de software:

1. Elicitação de requisitos;
2. Arquitetura e design de software;
3. Desenvolvimento;
4. Testes e garantia de qualidade;
5. Implementação;

O **processo de software**: a forma como todas as atividades para desenvolver software devem ser realizadas: ferramentas, métodos, documentação, diretrizes obrigatórias para escrever código, etc.

Subprocessos de Engenharia de Software ii



Controlo de qualidade i

Software não crítico

- Testes (sistema, componentes, testes unitários);
- Ambientes de staging.

Software crítico

- O processo é bastante rigoroso, além dos testes, podem ser utilizados métodos formais;
- Existem organismos de normalização que certificam o software: **IEC 60880** para software de centrais nucleares, FAA para aviação.

Software para aviónica i

O padrão de processo de software para sistemas aviônicos é definido no padrão **DO-178C**;

Para sistemas terrestres, o padrão relevante é o **DO-278A**;

Estes documentos determinam diretrizes de como preparar e apresentar o caso para a certificação junto a uma autoridade reguladora: como coletar e apresentar evidências.

Os componentes de software precisam ser certificados por autoridades reguladoras: Federal Aviation Administration (FAA) nos Estados Unidos, European Union Aviation Safety Agency (EASA) na União Europeia.

QA em Python

Exceções i

Os programas podem gerar problemas em tempo de execução. Por exemplo:

```
x = int(input("Escreva um número: "))
y = 1/x
print("Y: ", y)
```

Vai gerar um erro se x for zero, e o programa será interrompido.

Exceções ii

No entanto, erros também são objetos em Python e podem ser "capturados"por blocos try... except.

```
try:
    x = int(input("Escreva um número: "))
    y = 1/x
    print("Y: ", y)
except:
    print("Algum erro foi gerado. Poderia fornecer uma entrada válida da próxima vez?")
```

Exceções iii

O Python pode gerar erros por várias razões:

- **SyntaxError**
- **IndexError**;
- **SystemError**;
- **TypeError**;
- **RuntimeError**
- **KeyboardError**

Exceções iv

A captura de exceções pode ser específica para um tipo de exceção.

```
try:
    x = int(input("Escreva um número: "))
    y = 1/x
    print("Y: ", y)
except ZeroDivisionError:
    print("Divisão por zero")
except:
    print("Outro erro ocorreu")
else:
    print("Tudo correu bem")
```

Exceções v

```
try:
    x = int(input("Escreva um número: "))
    y = 1/x
    print("Y: ", y)
except ZeroDivisionError:
    print("Divisão por zero")
except:
    print("Outro erro ocorreu")
else:
    print("Tudo correu bem")
finally:
    print("Independentemente do que acontece, este bloco será sempre executado")
```

Explosão do Foguete Ariane 5 - O foguete explodiu devido a uma exceção de overflow não capturada;

Glitch de negociação da Knight Capital (2012): Um erro no deployment de software fez a Knight Capital, uma empresa de serviços financeiros, perder mais de 440 milhões em apenas 45 minutos. Uma função obsoleta falhou, não foi tratada corretamente, e começou a vender barato e comprar caro.

Lançar exceções i

Pode ser interessante sinalizar situações erradas num programa. Podemos lançar exceções com o comando **raise**. Exemplo:

```
x = -1

if x < 0:
    raise Exception("Desculpe, não são permitidos números abaixo de zero")
```

Lançar exceções ii

Outro exemplo:

```
x = "hello"

if not type(x) is int:
    raise TypeError("Apenas inteiros são permitidos")
```

A instrução assert i

Outra forma de lançar exceções se certas condições não forem cumpridas é através da instrução **assert**:

assert condição:

ou

assert condição, mensagem.

Quando uma condição não é cumprida, uma AssertionException é lançada. Pode também ser usada com uma mensagem.

A instrução assert ii

Exemplo:

```
x = 10

assert x < 9, "Este número é maior que 9"
```

O uso de exceções e asserts

É claro que uma boa gestão de exceções é fundamental no software do mundo real, ou seja, é essencial para a tolerância a falhas do software.

Também pode ser usada para depurar e testar programas:

- Identificar erros em programas;
- Depurar, rastrear e corrigir os erros em programas.

Depuração (Debugging)

Desenvolver um programa para calcular:

- Receber um número entre 20 e 40;
- Extrair os dois dígitos;
- Somar o ano do seu nascimento a cada dígito;
- Multiplicar por 46^2;
- Somar o resultado módulo 389;
- Dividir por 10;
- Apresentar o resultado ao utilizador.

Testes

Os testes unitários são os testes mais simples realizados em software e, normalmente, são implementados pelo programador. O objetivo é testar pequenas partes do código, com objetivos bem definidos.

Perguntas?