UNIVERSIDADE EDUARDO MONDLANE

FACULDADE DE ENGENHARIA

DEPARTAMENTO DE ENGENHARIA ELECTROTÉCNICA

Engenharia de Software 1

Discentes:

- Almeida, Henriques
- Boa, Valter
- Manuel, Modesta
- Muando, Felisberto
- Niuare, Geraldo
- Pelembe, Jaime

Docentes:

- Sergio Mavie , MSc
- Eng. Ivone Cipriano

Maputo, Março de 2024

Processos de Software

Introdução

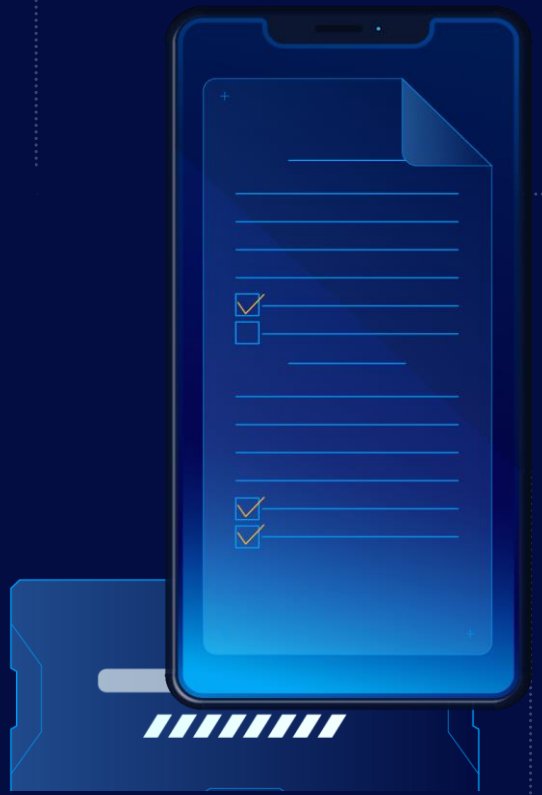
Objectivos

- Definição de processos de software
- Conhecimento sobre modelos

Metodologia

- Para alcançar o objetivo proposto, a pesquisa será realizada através da revisão bibliográfica de materiais relevantes, como:
  - Artigos científicos;
  - Livros;
  - Relatórios de pesquisa de mercado;
  - Tutoriais e documentações online.

Na concepção de softwares usando a engenharia, surge algumas atividades ou ações que são obedecidos para que segue na produção desejada a este novo conceito chamamos de **Processos de Software**, um conjunto coerente de atividades para a produção de software. Nesta etapa possui modelos e quando podem ser usados bem como devem ser organizados de maneira a lidar com as mudanças nos requisitos e projeto de software.



Processos de Software

Um processo de software é um conjunto de atividades relacionadas que levam à produção de um produto de software. (Sommerville,2011).

Atividades fundamentais para a engenharia de software no processo de software:

- ❑ Especificação de software.
- ❑ Projeto e implementação de software
- ❑ Validação de softwareDocumentação de Usuário
- ❑ Evolução de software.

Actividades

As quatro actividades fundamentais para a engenharia de Software são:

1. Especificação de software - A funcionalidade do software e as restrições a seu funcionamento devem ser definidas.

2. Projecto e implementação de software - O software deve ser produzido para atender as especificações.

3. Validação de software - O software deve ser validado para garantir que atenda as demandas do cliente.

4. Evolução de software - O software deve evoluir para atender as necessidades de mudança dos clientes.

Actividades (cont.)

A descrições do processo também podem incluir:

1. Produtos, que são os resultados de uma das actividades do processo.

2. Papéis, que reflectem as responsabilidades das pessoas envolvidas no processo.

3. Pré e pos-condicoes, que são declarações verdadeiras antes e depois de uma actividade do processo ou da produção de um produto.

Artefactos

Os **artefactos** do processo de desenvolvimento de software são os produtos ou documentos gerados durante as diferentes etapas do ciclo de vida do desenvolvimento de software.

Alguns artefactos comuns incluem:

- ❑ Documentos de Requisitos
- ❑ Modelos e Diagramas
- ❑ Documentação de Design
- ❑ Documentação de Usuário
- ❑ Código Fonte
- ❑ Entregáveis do Projeto

Modelos de processos

- São uma representação simplificada de um processo de desenvolvimento de software;
- Cada modelo representa uma perspectiva particular de um processo;
- O modelo é escolhido com base na natureza do projecto, métodos e ferramentas a serem usadas;

Ciclo de vida de Desenvolvimento de Software (Modelo CVDS)

Modelo cascata

Neste modelo o desenvolvimento os processos de desenvolvimento é feito de forma sequencial e sistemática.

É um fluxo sequencial de operações.

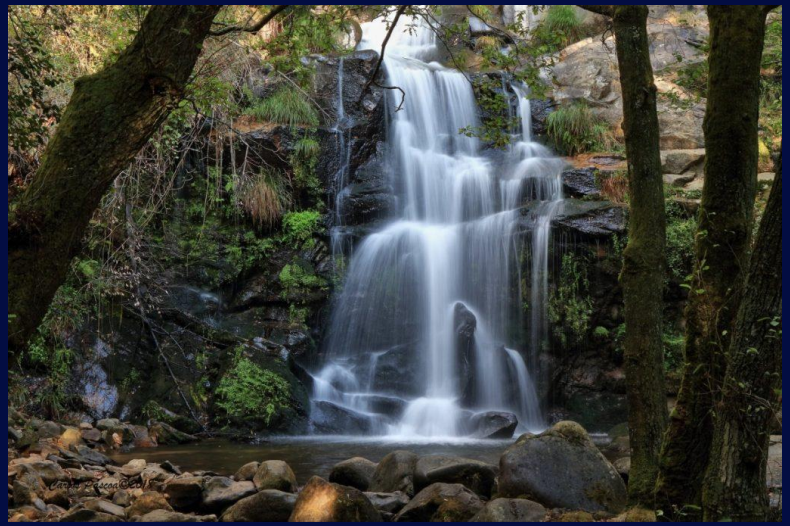


Fig 1. Cascata

Fig 2. Modelo Cascata

Modelo Cascata (cont)

- Vantagens:
  - Modelo de fácil entendimento
  - Eficiente quando os requisitos são bem definidos
  - Útil para adequações em sistema eistente
- Desvantagens:
  - É difícil o cliente estabelecer explicitamente todas as necessidades

Modelo Cascata (cont)

- Exemplos de aplicações que utilizam o modelo cascata
  - Sistemas de Gestão Empresarial
  - Aplicativos de Contabilidade
  - Aplicativos de Gerenciamento de Projetos
  - Sistemas de Gerenciamento de Banco de Dados
  - Sistemas de Reservas ou Agendamento

Modelo Iterativo

- É usado para identificar os requisitos e pode ser a melhor escolha quando um cliente definiu apenas objectivos gerais para o software;
- Tem inicio na colecta de requisitos, depois disso um projecto é elaborado e a partir dele é construído um protótipo de avaliação;

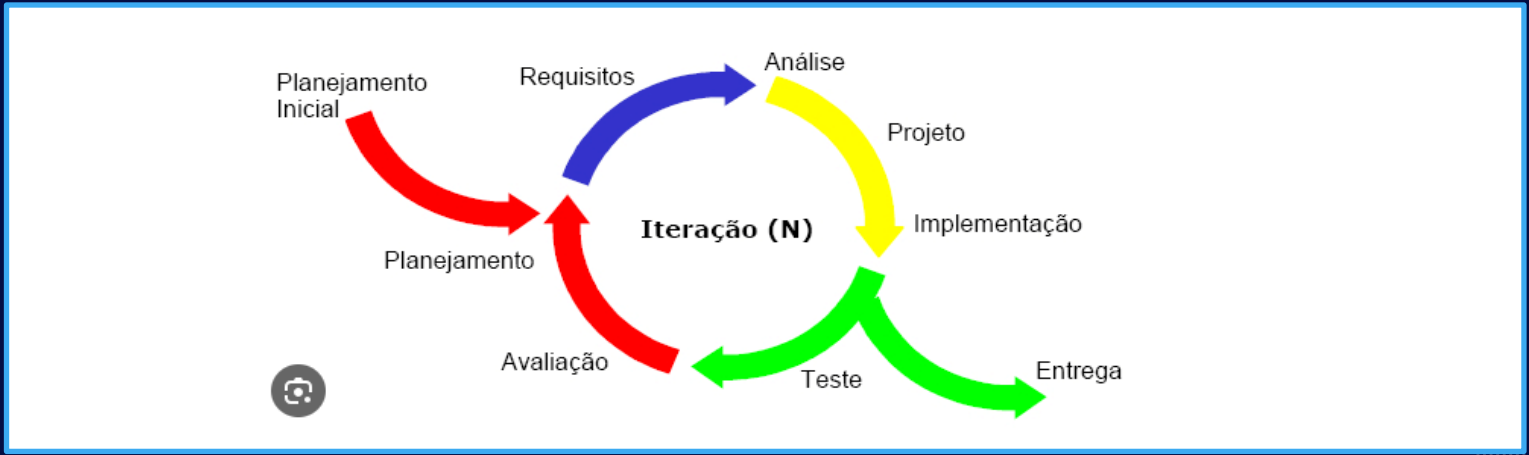


Fig 3. Modelo iterativo

Modelo Iterativo (cont)

- Vantagens:
  - Facilidade de determinar os requisitos inicias e a garantia de atingir as necessidades do cliente;
  - O produto final é mais corrente com os requisitos.
- Desvantagens:
  - A implementação rápida do protótipo pode ser comprometida;
  - Pode surgir a dificuldade de determinar o fim do desenvolvimento;
  - Tendência de utilizar o protótipo como produto final;

Modelo Iterativo (cont)

- Exemplos de aplicações desenvolvidas:
- Softwares de Colaboração: Google Docs, Microsoft Teams;
- Plataformas de E-commerce: Shopify, Magento;
- Sistemas de Gerenciamento de Conteúdo (CMS):WordPress, Joomla e Drupal

Modelo Incremental

- É baseado na ideia de desenvolver uma implementação inicial, expô-la aos comentários dos usuários e continuar por meio da criação de várias versões até que um sistema adequado seja desenvolvido;
- Cada incremento incorpora alguma funcionalidade necessária para o cliente;
- As actividades desse modelo são:
  - Planejamento;
  - Análise de riscos;
  - Engenharia;
  - Avaliação feita pelo cliente;

Modelo Incremental (cont)

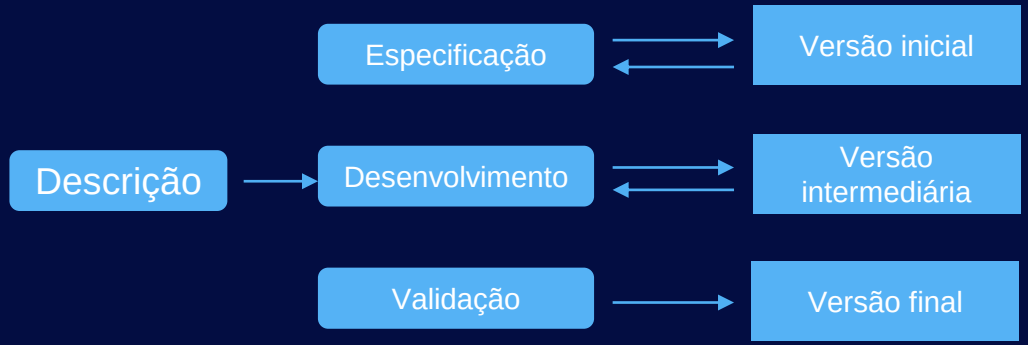


Fig 4. Modelo incremental

- Vantagens:
  - Custo reduzido ao acomodar mudanças nos requisitos;
  - É fácil obter feedback dos clientes sobre o desenvolvimento feito;
  - É possível obter entrega e implementação rápida de um software útil ao cliente;

Modelo Incremental (cont)

- Desvantagens:
  - Produção de muita documentação;
  - A estrutura do sistema tende a se degradar com adição de novos incrimentos

- Exemplos de aplicações desenvolvidas:
  - Redes sociais: Facebook, Instagram e Twitter.
  - Aplicativos de Mensagens: Whatsapp e Telegram.
  - Aplicativos de produtividade: Microsoft office, Google Workspace e Trello.
  - Jogos: Free Fire, Clash of Clans e GTA 5 online.

Conclusão

- A aplicação da implementação do ciclo de vida na concepção de softwares, reduz a crise de softwares e possíveis erros, devido ao facto de abordar vários conceitos importantes.

Referências

- Pressman, R.B. Software Engineering: A Practitioner's Approach McGraw-Hill, Third Edition, 1992, New-York, EUA;
- Sommerville, I.,Software Engineering.Addison-Wesley, 9a Edition, 2011

Obrigado!