INSTITUTO DE CIÊNCIAS EXATAS

DEPARTAMENTO DE COMPUTAÇÃO

IC812 - Análise e Projeto de Sistemas

Eduardo Kinder Almentero
ekalmentero@gmail.com

Ementa

1. Revisão
1.1 Processo de Software
1.2 Ciclo de Vida
1.3 Modelos de Ciclo de Vida

2. UML 2.5
2.1 Visão Geral
2.2 Principais Diagramas

2. Engenharia de Requisitos
2.1 LAL
2.2 Cenários
2.3 Casos de Uso
2.3.1 Diagrama de Casos de Uso
2.3.2 Descrição de Casos de Uso

3. Arquitetura de Software
3.1 Introdução à Arquitetura
3.2 Padrões de Arquitutra

4. **Diagramas Estruturais**
4.1 Classe
4.2 Objeto
4.3 Pacotes

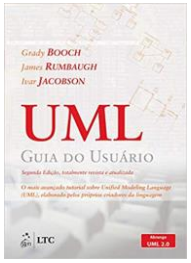
5. **Diagramas Comportamentais**
5.1 Atividade
5.2 Maquina de Estados
5.3 Diagramas de Interação
5.3.1 Diagrama de sequência
5.3.2 Diagrama de interação

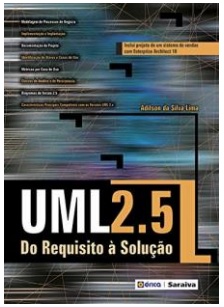
Avaliação

- (P1 + P2 + 2*Trabalho)/4
- Trabalho
  - Consistirá na divisão da turma em grupos para a utilização das técnicas de modelagem estudadas.
- Provas
  - Discursivas e individuais;
- Datas
  - P1: 11/03,
  - P2: 29/04,
  - Entrega do trabalho (previsão): 29/04 – sexta-feira,
  - Optativa: 02/05.

Bibliografia

- Principal
  - Uml - Guia do Usuário – 2ª edição (2006). Grady BOOCH, James RUMVAUGH e Ivar JACOBSON. Editora Novatec. Editora GEN LTC
- Complementares
  - Guia SWEBOK v3. (download gratuito disponível em https://www.computer.org/education/bodies-of-knowledge/software-engineering/v3)
  - Engenharia de Software: uma abordagem profissional – 7ª edição (2011). Roger S. Pressman. Editora AMGH.
  - UML 2.5 - do Requisito a Solução (2018). ADILSON DA SILVA LIMA. Editora Érica.





O que é um processo de software?

- Conjunto de **atividades**, **métodos**, **práticas** e **transformações** que **guiam** as pessoas no desenvolvimento de software
- Um processo **eficaz** deve, claramente, **considerar** as **relações** entre:
  - **artefatos produzidos** no desenvolvimento,
  - **ferramentas** e **procedimentos** necessários
  - **habilidade**, **treinamento** e a **motivação** do pessoal envolvido.

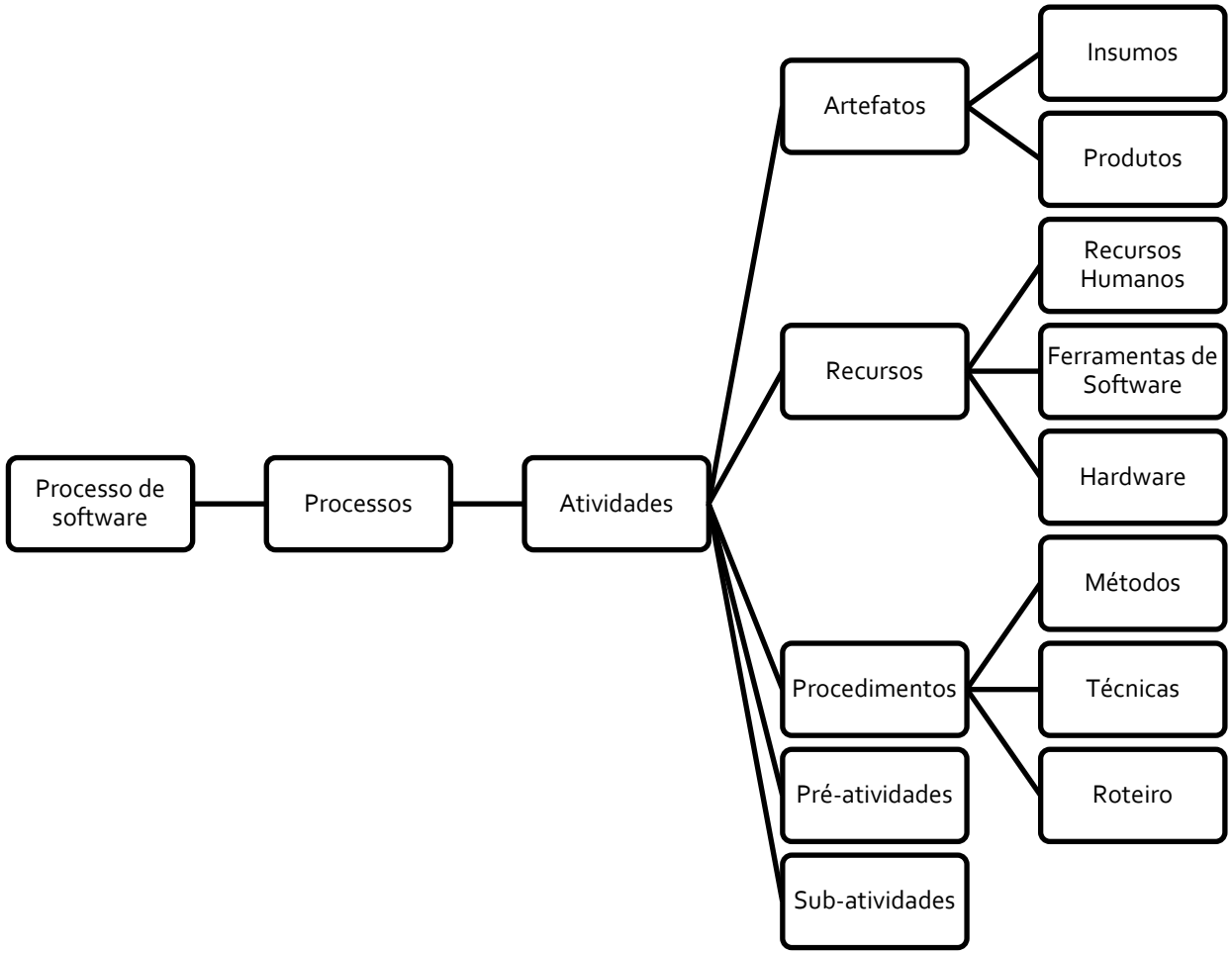
Processo de software

- Elementos que **compõem um processo de software**
  - **Processos de software** são, geralmente, decompostos em **diversos processos**:
    - Processo de desenvolvimento,
    - Processo de garantia da qualidade,
    - Processo de gerência,
    - ...

Elementos de um processo de software

- Os **sub-processos** são compostos de **atividades**, que também podem ser decompostas
- Para cada **atividade do processo**, é importante saber:
  - quais suas **subatividades**,
  - **atividades que devem precedê-las** (pré-atividades),
  - **artefatos de entrada** (insumos) e **saída** (produtos),
  - **recursos necessários** (humanos, hardware, software, etc.),
  - **procedimentos a serem utilizados** (métodos, técnicas, modelos etc.).

Elementos de um processo de software



Porque definir processos?

- Favorecer a **produção de software** de **alta qualidade**, atingindo as **necessidades** dos **usuários finais**, **respeitando** um **cronograma** e **orçamento previamente definidos**
- Um processo de software **não** **pode ser definido de forma universal**;
  - Por quê?
- Para ser **eficaz** e conduzir a construção de produtos de boa qualidade, o processo deve ser **adaptado as especificidades** do projeto em questão.

Definição de processo de software

- **Processos devem ser definidos caso a caso**, considerando as **características**:
  - **domínio do problema** (organização onde o software será utilizado), **tamanho** e **complexidade**,
  - **tecnologia utilizada na sua construção**, (paradigma de desenvolvimento, linguagem de programação, mecanismo de persistência etc.),
  - **organização onde o produto será desenvolvido**,
  - **equipe de desenvolvimento** (habilidades, conhecimentos, experiência etc.)

Definição do ciclo de vida

- A escolha de um **modelo de ciclo de vida** é o ponto de partida para definição de um processo de software.
- O modelo de ciclo de vida **organiza** as **macro-atividades** básicas do processo, estabelecendo **precedência** e **dependência** entre as mesmas.
- **Cada atividade** descrita no modelo de ciclo de vida **deve ser decomposta** e para suas **sub-atividades** devem ser **associados métodos**, **técnicas**, **ferramentas** e **critérios de qualidade**.

Definição do ciclo de vida

- Atividades de **cunho gerencial** devem ser definidas, entre elas atividade de **gerência de projetos** e **controle e garantia da qualidade**.
- Outros fatores que influenciam na definição:
  - tipo de software (ex: sistema de informação, sistema de tempo real);
  - estabilidade dos requisitos.

Modelo de Processo (ou ciclo de vida)

- Pode ser visto como uma **representação abstrata** de um **esqueleto de processo**;
- Inclui tipicamente **atividades principais** e **ordem de precedência** entre elas;
- Pode incluir **artefatos requeridos** e **produzidos** pelas atividades;
- De maneira geral, um **modelo de processo** descreve a **organização das atividades, estabelecendo fases e como elas se relacionam**.

Modelo de Processo (ou ciclo de vida)

- O modelo de processo **NÃO** descreve um curso de ações **preciso**, **recursos**, **procedimentos** e **restrições**.
- É um importante ponto de partida para definir **como** o projeto deve ser **conduzido**
- **NÃO** é o **suficiente** para **guiar** e **controlar** um projeto de software na prática

Modelo de Processo (ou ciclo de vida)

- Ainda que processo tenham que ser definidos caso a caso, de maneira geral, o **ciclo de vida** de um **software envolve**, pelo menos, as seguintes **fases**:
  - Planejamento;
  - Análise e especificação de requisitos;
  - Projeto (arquitetura);
  - Implementação (desenvolvimento);
  - Testes;
  - Entrega e implantação;
  - Operação;
  - Manutenção;

Fases do ciclo de vida: 1) Planejamento

- Fornece uma **estrutura** que possibilita o gerente fazer **estimativas razoáveis** de recursos, custos e prazos;
- Após elicitar os **requisitos iniciais**, um **plano de projeto** deve ser elaborado, **configurando o processo** a ser utilizado no desenvolvimento do software;
- À medida que o **projeto progride**, o **planejamento deve ser detalhado** e atualizado regularmente.

Fases do ciclo de vida: 1) Planejamento

- Pelo menos **ao final de cada uma das fases** do ciclo de vida (análise e especificação de requisitos, projeto, implementação e testes) o **planejamento como um todo deve ser revisto** e o **planejamento da etapa seguinte deve ser detalhado**.
- O **planejamento e o acompanhamento do processo** fazem parte do **processo de gerência**.

Fases do ciclo de vida: 2) Análise e Especificação de Requisitos

- **Intensificação do processo da engenharia de requisitos**;
- O **escopo é refinado** e **os requisitos definidos de forma mais detalhada**.
- Para entender a natureza do software a ser construído, o engenheiro tem que compreender o **domínio do problema**, bem como as **funcionalidades** e **comportamento** esperado.

Fases do ciclo de vida: 2) Análise e Especificação de Requisitos

- Uma vez capturados os requisitos do sistema a ser desenvolvido, estes devem ser **modelados**, **avaliados** e **documentados**.
- Uma das partes vitais desta fase é a **construção de um modelo** descrevendo **o que o software tem que fazer** (e **NÃO como** fazê-lo).

Fases do ciclo de vida: 3) Projeto

- Responsável por incorporar **requisitos tecnológicos** aos **requisitos essenciais** do sistema.
- Requer que a **plataforma de implementação seja conhecida**.
- Envolve duas grandes etapas: **projeto da arquitetura** do sistema e **projeto detalhado**.
- O objetivo da do projeto da arquitetura é definir a **arquitetura geral** do software, tendo como **base** o **modelo** construído na fase de análise de **requisitos**.

Fases do ciclo de vida: 3) Projeto

- O propósito do **projeto detalhado** é **especificar o projeto do software para cada componente** identificado na etapa anterior.
- Os **componentes do software devem ser sucessivamente refinados**, em níveis maiores de detalhamento, até que possam ser **codificados e testados**.

Fases do ciclo de vida: 4) Implementação

- O **projeto deve ser traduzido** para uma forma **passível de execução pelo hardware**;
- Nesta fase, **cada unidade de software especificada** no **projeto detalhado** é **implementada**;
- De maneira geral, o **código-fonte** necessário para que o **software tenha o comportamento especificado** é **escrito**;
- Esta fase também **inclui a escrita de código-fonte para realização de testes automatizados** (normalmente **testes unitários**).

Fases do ciclo de vida: 5) Testes

- Inclui **diversos níveis de teste**, como:
  - **Teste de unidade**,
  - **Teste de integração** e
  - Teste de sistema.
- **Cada unidade de software** implementada **deve ser testada** e os **resultados documentados**.
  - Evidência de teste.
- Os **componentes desenvolvidos** devem ser **integrados**, até que se obtenha o **sistema**, que deve ser **testado como um todo**.

Fases do ciclo de vida: 6) Entrega e implantação

- Após testado, o software deve ser colocado em **produção**.
- Para tal, é necessário **treinar os usuários**, **configurar o ambiente** de produção e, muitas vezes, **converter bases de dados**.
- O propósito desta fase é estabelecer que o **software satisfaz os requisitos dos usuários**.
- Isto é feito instalando o software e conduzindo **testes de aceitação.**
- Quando o software tiver demonstrado prover as **capacidades requeridas**, ele pode ser **formalmente aceito** e a **operação iniciada.**

Fases do ciclo de vida: 7) Operação

- Nesta fase o **software é utilizado pelos usuários no ambiente de produção (real)**;
- Eventuais "**bugs**" podem ser **documentados** para **futuras atualizações** do software;
- **Novas funcionalidades** podem ser **anotadas** para **futura evolução** do software;
- Eventuais **configurações** podem ser **necessárias** para **adaptar o software** a **mudanças do ambiente**.

Fases do ciclo de vida: 8) Manutenção

- Na grande maioria das vezes, o software **sofrerá mudanças após ter sido entregue** aos clientes.
- Alterações ocorrerão devido a:
  - **Erros** encontrados;
  - **Mudanças** no **ambiente externo**;
  - **Novas funcionalidades**;
  - **Aumento do desempenho**;
  - ...

Fases do ciclo de vida: 8) Manutenção

- Dependendo do **tipo e porte da manutenção** necessária, essa fase pode **requerer a definição e condução de um novo processo**;
- Neste novo processo, **cada uma das fases anteriores** é **refeita** no **contexto de um software existente**.

Modelos de Ciclo de Vida

- Modelos sequenciais
  - Como o nome indica, organizam o processo em uma **sequência linear de fases**.
  - O principal modelo desta categoria é o **modelo em cascata** (*waterfall* em inglês).
  - Este modelo serve como base para outros modelos, inclusive **modelos incrementais** e **evolutivos**.

Modelo em V

- É um **modelo sequencial**, muito **similar** ao **cascata**;
- A diferença é que, **a partir da implementação** (escrita do código), há uma **série de etapas de avaliação da qualidade** (testes);
- Cada etapa envolve a **avaliação de artefatos específicos**, e a **retroalimentação gerada** está relacionada a etapa responsável pela **criação destes artefatos**;
- De maneira geral, em **relação ao modelo em cascata**, **este fornece uma flexibilidade maior** de **retorno** a **etapas anteriores para modificações.**

Modelos Incrementais

- Há situações onde os **requisitos** são **bem definidos**, mas o **tamanho** e **complexidade** do **sistema dificulta a adoção do modelo sequencial.**
  - Principalmente se o **usuário desejar resultados rapidamente**.
- Para estes casos, uma possibilidade é o uso de um **modelo incremental;**
- No desenvolvimento incremental, o sistema é **dividido** em partes (**módulos**) com base em suas **funcionalidades.**

Modelo Incremental

- "Ser incremental" é uma filosofia **básica**, que comporta diversas **variações**.
- Princípio fundamental: a **cada** nova **iteração**, uma **versão operacional** do sistema será desenvolvida e **entregue** para avaliação do **cliente**.
- Para isto, é preciso um l**evantamento inicial de requisitos**.
- É preciso **verificar** se o sistema pode ser produzido em **módulos**.
  - Para que seja possível **planejar** os **incrementos.**

Modelos Evolutivos

- **Sistemas de software**, assim como quaisquer sistemas complexos, **evoluem** ao longo do tempo.
- Seus **requisitos**, muitas vezes, são **difíceis** de serem estabelecidos ou **mudam com frequência** ao **longo do desenvolvimento**.
- É importante ter como opção **modelos de ciclo** de vida que **lidem** com **incertezas** e acomodem melhor as **contínuas mudanças**.
- **Alguns modelos incrementais**, dado que preconizam um desenvolvimento iterativo, **podem** ser **aplicados** a esses casos.
- Porém, **a grande maioria** deles toma por pressuposto que os **requisitos são bem definidos e estáveis**.

Modelos Evolutivos

- Modelos **evolutivos** (ou **evolucionários**) buscam **preencher essa lacuna**.
- **Modelos incrementais** têm como base a entrega de **versões operacionais** desde o **primeiro ciclo.**
- Nos **modelos evolutivos** os **primeiros ciclos** produzem apenas **protótipos** ou, até, **apenas modelos**.
- Com o **avanço do desenvolvimento**, os **requisitos** se tornam **mais claros** e **estáveis**.
  - **Protótipos** dão lugar a **versões operacionais**, até que o **sistema completo** seja construído

Material de apoio

- Bibliografia básica
  - PRESSMAN, R, S. Engenharia de Software Uma Abordagem Profissional. 7a. ed. McGrawHill, 2011.
- Bibliografia complementar
  - P. Bourque and R.E. Fairley, eds., Guide to the Software Engineering Body of Knowledge, Version 3.0, IEEE Computer Society, 2014; www.swebok.org.
    - Download gratuito.

INSTITUTO DE CIÊNCIAS EXATAS

DEPARTAMENTO DE COMPUTAÇÃO

Perguntas?