Modelos conceituais

Técnicas em modelagem de aplicações
(CI1061-CI092-INFO7052)

Prof. Marcos Didonet

UFPR
UNIVERSIDADE FEDERAL DO PARANÁ

Modelos conceituais

1. Classificação/Instanciação
2. Generalização/Especialização
3. Agregação(composição)/Decomposição
4. Associação

https://app.diagrams.net/
Ferramentas para criação de modelos conceituais:
https://modeling-languages.com/text-uml-tools-complete-list/

Representação concreta

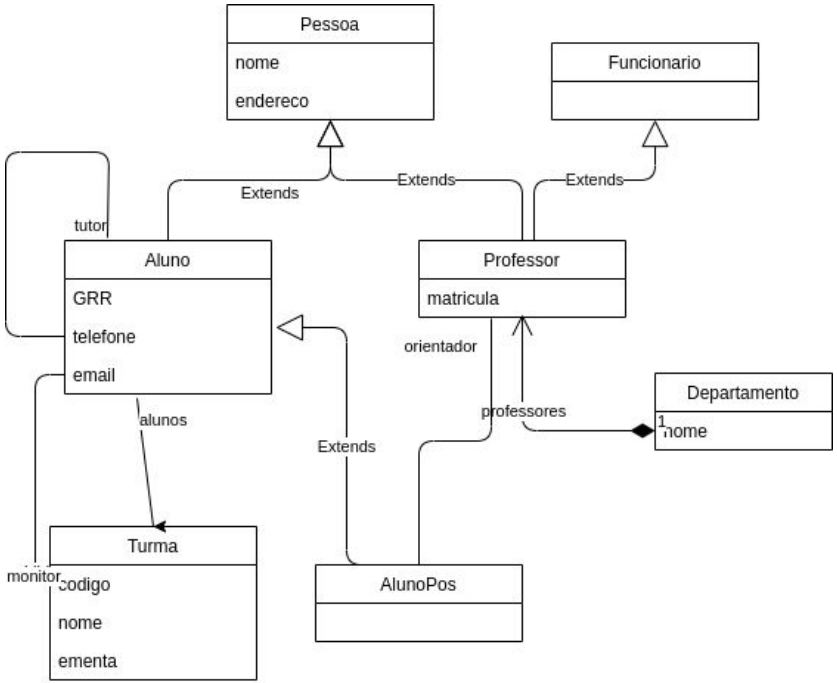
- Abstrações escolhidas devem ser passadas para representação concreta
- Muitos formalismos e plataformas existentes
- Uma plataforma não é A MELHOR, mas a mais adaptada para um determinado conjunto de problemas
- Escolha será feita na fase de projeto (de software/de dados)

- Mas ***como escolher ?***

Estabelecer critérios para escolha

- Expressividade
  - Relações entre elementos, atributos, suporte a métodos, etc.
- Existência de ferramentas de suporte
  - Linguagens de definição e consulta (textuais, gráficas)
  - Ferramentas de manipulação
- Desempenho ?
- Armazenamento, se necessário

Modelo conceitual



Modelo relacional

- Armazenamento e consulta
  - Tabelas (relacoes), colunas (atributos), tuplas (linhas)
  - Mapeamento a partir de chaves
  - Linguagem SQL
  - Concorrência, transação, acesso
- Vários SGBD’s
- Exemplo:

```
CREATE TABLE PESSOA (
    COLUMN NOME VARCHAR(50);
    COLUMN ENDERECO VARCHAR(50);
)
```

Hierárquico

- Pai – filho
- Hierarquia
- Fácil acesso aos elementos
  - difícil otimização
- Consulta com extensoes de SQL
- Banco de dados – IMS – 1968
- Mais rápido que BD relacional para acesso direto, mas mais difícil de implementar

XML (hierárquico)

- Árvore, com relações de composição
  - Nós, atributos e subnós
  - Semi-estruturado
- Baseado em arquivos
  - Registros (texto ou binário)
- Suporte extenso
  - Bancos de dados, APIs Javas e outras linguagens
  - Editores (Eclipse, NetBeans, XMLSpy)
  - Visualizadores (browser)
- Linguagem
  - XPath, XQuery

```
<alunos>
   <aluno nome = "José" cpf = "123456">
       <turmas>
         <turma codigo = "ci1061" nome ="tecnicas em
modelagem"\>
         <turma codigo = "ci057" nome ="alg3"\>
       </turmas>
   </aluno>
</alunos>
```

XML: exemplo

```
<alunos>
   <aluno nome = "José" cpf = "123456">
       <turmas>
         <turma codigo = "ci1061" nome ="tecnicas em modelagem"\>
         <turma codigo = "ci057" nome ="alg3"\>
       </turmas>
   </aluno>
</alunos>
```

Objeto

- Surgiu como novo paradigma de programação
- noção de classe e objetos
- Expressividade alta
  - Objetos contém atributos
  - Se relacionam com outros objetos
  - Métodos para troca de mensagens
- Altamente dependente de linguagens de programação
  - Muitas linguagens, com suporte variado
- Sem linguagem de consulta padrão
- Navegação simples

Objeto: exemplo em Java

```
public class Pessoa {
            String nome;
            String endereco;
            ArrayList<Turma> turmas;
}
public class Turma {
            private String nome;
            String endereco;
            String getNome() {
                return nome;
            }
            void setNome(String nome) {
                nome = nome;
            }
}
```

NoSQL

- Chave/valor
  - Chave, valor
  - Acesso direto. Exempo: Google
  - Não é feito para fazer queries
  - MapReduce, Hadoop

Documento

- Documentos (arquivos) aninhados → JSON
- Casos de uso
  - Interoperabilidade
  - Retorno de APIs : https://simcaq.c3sl.ufpr.br/api/v1//enrollment?dims=adm_dependency_detailed&filter=min_year:%222019%22,max_year:%222019%22
  - Dados “simples”
- Armazenamento
  - MongoDb, CouchDb, entre outros

JSON: exemplo

```
{
  "alunos": [
    {
      "nome": "José",
      "cpf": "12341234",
      "telefone": "9999999",
      "": [
        {
          "codigo": "CI1061",
          "nome": "Técnicas em modelagem"
        },
        {
          "codigo": "CI1057",
          "nome": "Algoritmos 3"
        }
      ]
    }
  ]
}
```

Ontologias

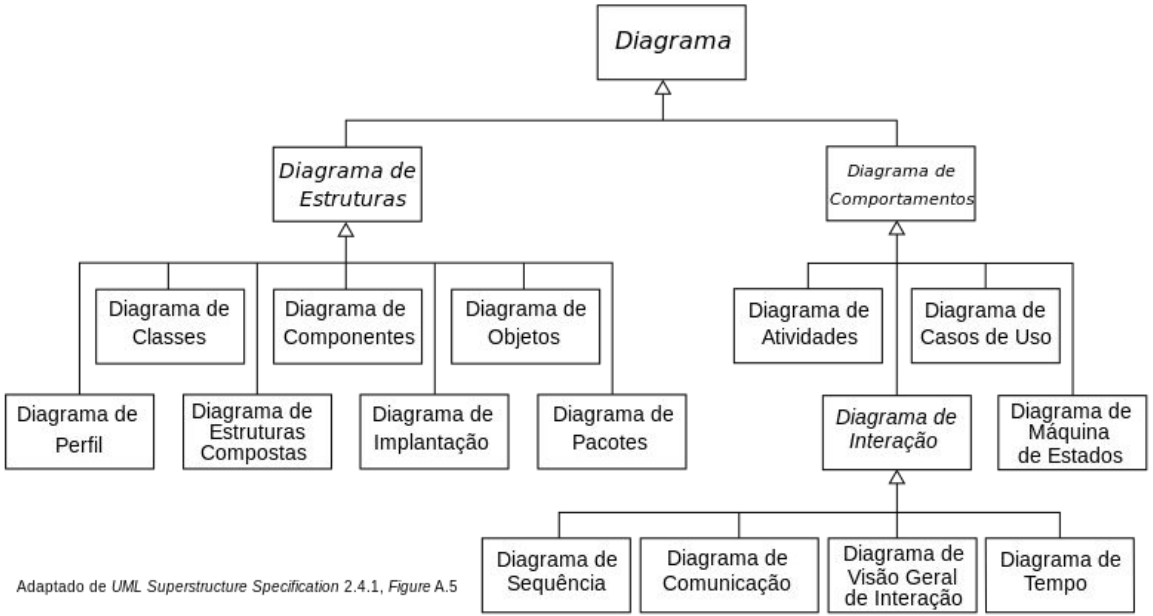
- Formato de representação de conhecimento
  - objeto; propriedade;valor (triplas)
    - professor, orienta, aluno
    - José, categoria, professor
    - José, idade, 15
- Linguagens/frameworks
  - OWL, Sparql
  - Protegé
- Formalismo: logica de descrição (logica de primeira ordem)
  - problema : desempenho, otimização, grandes sistemas

Grafos

- Elementos e relações, e conexões entre eles
- Grande expressividade
- Independente de linguagem e/ou paradigma
- Elementos do grafo podem ser tipados de acordo com o dominio
- Suporte crescente
    - Neo4J, OrientDb

Modelagem de software

- UML (Unified Modeling Language)

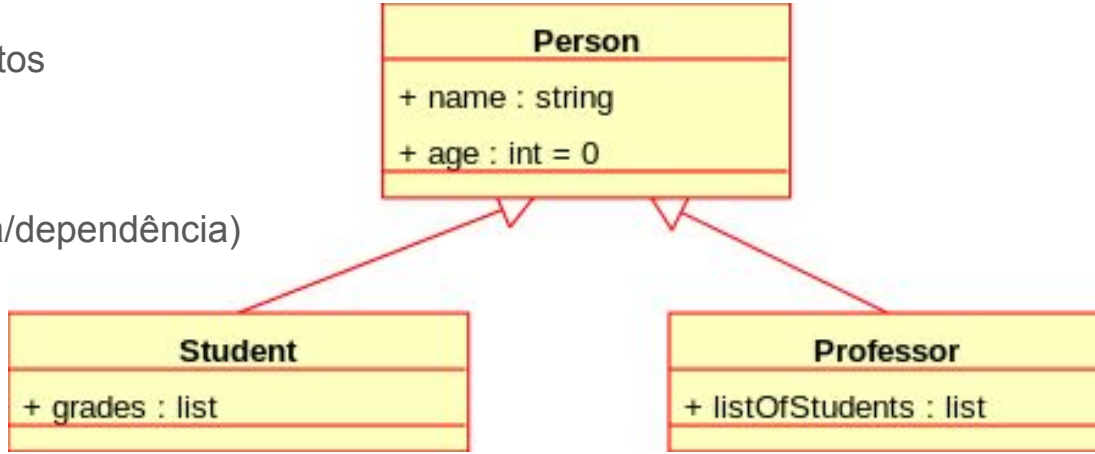


Adaptado de *UML Superstructure Specification* 2.4.1, *Figure* A.5

Diagrama de classes

- Membros
  - Atributos,métodos, relacionamentos
- Relacionamentos
  - Referências (associação/composição/herança/dependência)
- Métodos
  - Retorno, parâmetros, visibilidade

Um exemplo simples



- Mais detalhes em:
  - https://pt.slideshare.net/jcabot/modeldriven-software-engineering-in-practice-chapter-4