3 Lab: Modelação com classes

Enquadramento

As Classes da UML são categorias de objetos, ou seja, *tipos* de coisas/conceitos. Os objetos podem ser entidades da análise ou da implementação. Na análise, os as classes correspondem a conceitos de interesse para o problema; na implementação, a entidades do software (e.g.: objetos em Java). A perspetiva, neste laboratório, é a do Analista e estamos à procura de "categorias de coisas".

Para cada classe vamos encontrar alguns atributos (dados que devem ser memorizados no sistema de informação). Nesta fase, o analista não está preocupado em representar métodos/funções nas classes (como será próprio da programação).

Os conceitos estão relacionados entre si, formando uma rede de conceitos, ligados por associações. O Diagrama de Classes fornece os elementos de modelação para construir esse mapa, seguindo a técnica de análise por objetos.

Objetivos de aprendizagem

- Identificar conceitos/classes na descrição de um problema.
- Caraterizar as estruturas de dados de um problema como classes e associações.
- Construir e interpretar diagramas de classes (perspetiva do analista).
- Utilizar associações "simples", agregações, composições e generalizações.

Preparação

- "class diagrams"- informação tutorial.

3.1

Considerando o modelo representado no Diagrama 1, explique se as seguintes afirmações têm ou não suporte no modelo, isto é, se são V/F face ao que está no diagrama.

a) Todas as Equipas precisam de indicar um Professor responsável.
b) Podem existir Professores que não coordenam nenhuma Equipa.
c) A Entrega (submissão) é feita por vários Alunos.
d) Uma Submissão é avaliada por um Membro do CC.
e) Uma Equipa poder ser composta por alunos de várias Instituições (i.e., a Equipa não é de uma Instituição).
f) Um Membro do CC só pode avaliar entregas resolvidas com linguagens de programação para as quais é especialista.
g) As Entregas de uma Equipa são sempre feitas pelo capitão da equipa.
h) As Entregas de uma Equipa relativa a um Desafio podem ser avaliadas por Docentes diferentes.

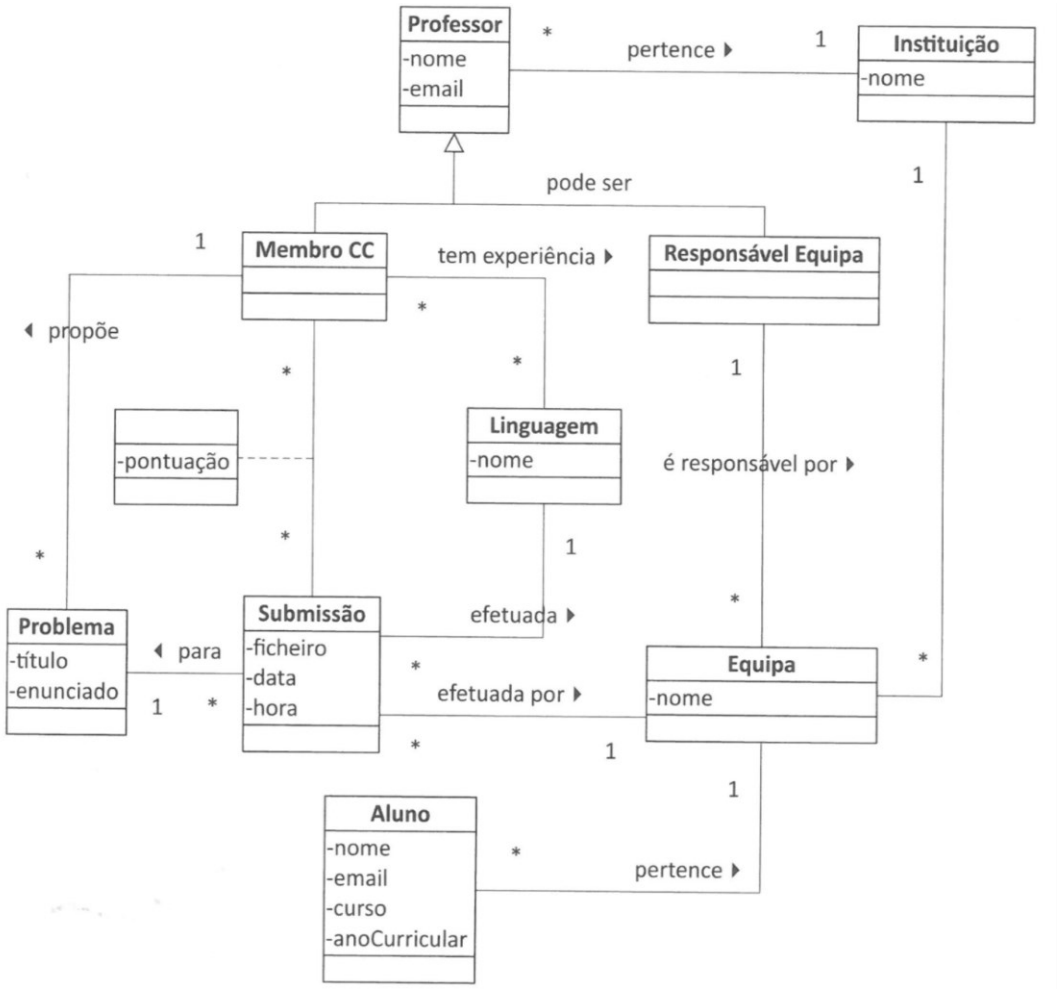

Diagrama 1 - ↑Modelo de classes relativo à organização de concursos de programação; CC: Conselho científico das provas. [In: Borges et al, "Modelação de Dados em UML: uma abordagem por problemas."]

3.2

Pesquise os seguintes livros no catálogo da Biblioteca da UA:

- "UML Distilled", de Martin Fowler.
- "Use case driven object modeling with UML", de D. Rosemberg.

Crie um diagrama UML para mapear os **conceitos do domínio** relativo ao funcionamento da biblioteca[1], com base no seguinte conhecimento da área do problema:

a) as obras podem ter vários autores. Para desambiguar os autores, usa-se o nome e o ano de nascimento.

b) os utilizadores pesquisam obras por autor, título, ano, ou uma combinação desses elementos.

c) Um livro pode ser classificado em diferentes assuntos (ou descritores).

d) para cada obra, podem existir vários exemplares, com cota e código de barras únicos, que podem ser levantados pelos utilizadores, em regime de empréstimo.

e) existem multas para devoluções tardias, mas nem todos os utilizadores têm o mesmo tempo para reter os livros em empréstimo domiciliário. Há que distinguir entre

[1] Neste exercício, pretende-se esboçar o mapa de conceitos. O mais importante é identificar os conceitos e as associações. Por isso, intencionalmente, pode-se omitir a especificação detalhada das classes (i.e.: lista complete de atributos, tipos de dados,...).

utilizadores que são alunos, professores ou utilizadores externos. O tempo de empréstimo normal é de 15, 90 e 30 dias respetivamente. Todos os utilizadores têm um número mecanográfico alfanumérico.

f) Para inscrever um utilizador externo, é necessário confirmar a sua identidade e a morada (com a apresentação de uma fatura, titulada ao utilizador, do fornecimento de eletricidade, água ou serviço similar).

g) Os utilizadores podem também pedir a reserva de obras para utilização numa data futura (sendo atribuído o primeiro exemplar disponível).

h) Existem vários polos (e.g.: Biblioteca Campus Santiago, Mediateca, Biblioteca ESTGA,...) nos quais se encontram os exemplares. Cada polo tem o seu próprio horário de funcionamento.

3.3

Considere a área da gestão de projetos, por exemplo, relativamente ao seguimento de projetos de desenvolvimento de software. Para concretizar, pode experimentar as funcionalidades da ferramenta de gestão de projetos Redmine, no respetivo site (ou até no code.ua.pt, que é baseado no Redmine).

A título exploratório, experimente (pelo menos) os seguintes passos, no contexto de uma equipa:

- Criar um projeto.
- Configurar os módulos que se pretende utilizar no projeto; incluir os módulos "Issues", "Time Tracking" e "Gantt". Quanto aos Trackers, pode-se aceitar a predefinição.
- Configurar a equipa, adicionando os respetivos membros (Settings > Members). Note que os membros podem ter papéis diferenciados. Será útil ter mais que um elemento do grupo registado na plataforma.
- Adicionar uma nova tarefa ao projeto (New Issue), relativa, por exemplo, "Protótipo da página de pesquisa de filmes por género".
- Configure a tarefa (Issue X) definindo, pelo menos, a descrição, prioridade, data de início e de finalização. Atribua a tarefa a um responsável (Assignee) e envolva mais pessoas no acompanhamento do progresso (Watchers).
- Verifique no cronograma (Gantt) o posicionamento da tarefa.
- Adicione agora uma segunda tarefa (Issue Y), como fez para a anterior, fazendo variar as caraterísticas (prioridade, datas, assignee, watchers, etc).
- Volte à listagem de Issues e aceda ao detalhe do Issue X (o primeiro). Mude o estado para "Em curso". Use a secção de Log time para registar tempo de trabalho. Atualize também a % Done.
- Volte ao cronograma e verifique as alterações. Experimente filtrar a informação no cronograma para um responsável específico (Assignee).
- Experimente livremente alterar o estado das tarefas, reportar trabalho, alterar a duração, etc.

A partir do texto anterior (em caixa), identifique os principais substantivos e a partir daí, anote numa tabela os conceitos e atributos candidatos a serem incluídos no modelo do domínio. [Veja a secção "O que incluir e o que deixar de fora?

Ao construir o modelo de domínio, o Analista levanta o conhecimento que carateriza aquele problema. É importante não deixar conceitos relevantes de fora. Um teste prático pode ser feito com a pergunta "**é importante memorizar esta informação**, para o futuro"? Por exemplo, é importante memorizar que obras os utilizadores pesquisam no domínio da gestão de bibliotecas? Normalmente, não. Já a informação sobre as obras e utilizadores diremos que é essencial. Por isso, Pesquisa não seria um conceito; mas Obra e Utilizador sim.

Classe ou atributo?

Um erro comum quando se cria um modelo de domínio é representar algo como um atributo quando deveria ter sido uma classe conceptual (um "objeto" autónomo). Uma regra prática para ajudar a evitar este erro é: se não **pensarmos na "coisa" X** como um número ou texto no mundo real, X é provavelmente uma classe conceptual, não um atributo.

a) Como identificar os conceitos de um domínio?" na pág. 4].

Conceito candidato	**Atributos candidatos**
Projeto	Título, data de início [...]
...	...

Nota: nem todos os substantivos revelados na análise textual serão relevantes. Para além disso, a linguagem natural terá repetições e ambiguidades, que é preciso filtrar.

b)

Uma outra abordagem para fazer o levantamento dos conceitos de um domínio é utilizar uma lista de categorias para procurar conceitos (classes).

Utilizando a chave incluída (veja a secção "O que incluir e o que deixar de fora?

Ao construir o modelo do domínio, o Analista levanta o conhecimento que carateriza aquele problema. É importante não deixar conceitos relevantes de fora. Um teste prático pode ser feito com a pergunta "**é importante memorizar esta informação**, para o futuro"? Por exemplo, é importante memorizar que obras os utilizadores pesquisam no domínio da gestão de bibliotecas? Normalmente, não. Já a informação sobre as obras e utilizadores diremos que é essencial. Por isso, Pesquisa não seria um conceito; mas Obra e Utilizador sim.

Classe ou atributo?

Um erro comum quando se cria um modelo de domínio é representar algo como um atributo quando deveria ter sido uma classe conceptual (um "objeto" autónomo). Uma regra prática para ajudar a evitar este erro é: se não **pensarmos na "coisa" X** como um número ou texto no mundo real, X é provavelmente uma classe conceptual, não um atributo.

Como identificar os conceitos de um domínio?" na pág. 4), procure identificar conceitos do domínio da gestão de projetos, para cada uma das categorias. Nota: poderá haver mais que um conceito em cada categoria, bem como poderá não existir nenhum.

Categoria conceptual	**Conceito identificado (domínio da gestão de projetos)**
e.g.: Lugares físicos	e.g.: Sala

...	...

c)

Utilizando a informação que obteve nas alíneas anteriores, crie um modelo do domínio da gestão de projetos.

O seu modelo de domínio deve ter a capacidade expressiva para permitir memorizar a **informação suficiente para suportar todos os resultados vistos** no Redmine (lista geral de *issues* e o seu estado, ficha com os detalhes do *Issue*, atribuição de tarefas a membros da equipa, Gantt, etc.).

Procure criar um modelo **completo** para o âmbito que foi experimentado[2].

Nota: o processo normal, será construir o modelo do domínio antes de haver sistema. Neste caso, estamos a fazer ao contrário (abstraindo a partir de um sistema real).

d)

A partir do resultado da alínea anterior, introduza as alterações (se necessário) para considerar ainda os seguintes requisitos:

— Um membro pode fazer parte da equipa do projeto com papéis diferentes, ao longo do tempo (e pretende-se memorizar quais).

— Um colaborador pode observar o progresso de uma tarefa (*Watcher*) durante períodos de tempo discretos, e não necessariamente durante toda a tarefa (e pretende-se memorizar quais).

Notas de estudo (Classes)

O que incluir e o que deixar de fora?

Ao construir o modelo do domínio, o Analista levanta o conhecimento que carateriza aquele problema. É importante não deixar conceitos relevantes de fora. Um teste prático pode ser feito com a pergunta "**é importante memorizar esta informação**, para o futuro"? Por exemplo, é importante memorizar que obras os utilizadores pesquisam no domínio da gestão de bibliotecas? Normalmente, não. Já a informação sobre as obras e utilizadores diremos que é essencial. Por isso, Pesquisa não seria um conceito; mas Obra e Utilizador sim.

Classe ou atributo?

Um erro comum quando se cria um modelo de domínio é representar algo como um atributo quando deveria ter sido uma classe conceptual (um "objeto" autónomo). Uma regra prática para ajudar a evitar este erro é: se não **pensarmos na "coisa" X** como um número ou texto no mundo real, X é provavelmente uma classe conceptual, não um atributo.

[2] A globalidade da ferramenta levará a um modelo complexo. A sugestão é focar nas interações que foram experimentadas.

Como identificar os conceitos de um domínio?

Larman apresenta duas estratégias que podem assistir o analista na descoberta dos objetos do domínio (conceitos): (I) seguir uma lista de categorias; (II) procurar nomes nas descrições disponíveis do problema (e.g.: na narrativa dos casos de utilização).

I) Pesquisa de conceitos partindo de uma lista de categorias

Categoria (de classes conceptuais)	Exemplos
• Transações comerciais Orientação: Classes essenciais num SI (envolvem dinheiro), por isso é um bom ponto para começar.	Sale, Payment Reservation
• Produto ou serviço relacionado transacionado Orientação: As transações comportam "coisas" individuais (um produto ou serviço). Considere-as a seguir.	Item Flight, Seat, Meal
• Onde é que a transação é registada? Orientação: Importante.	Register, Ledger FlightManifest
• papéis das pessoas ou organizações relacionadas com a transação; atores no caso de utilização Orientação: Normalmente precisamos de ter conhecimento sobre as partes envolvidas numa transação.	Cashier, Customer, Store Passenger, Airline
• local da transação; ponto de serviço	Store Airport, Plane, Seat
• eventos que merecem destaque, muitas vezes com uma hora ou lugar que precisamos de guardar	Sale, Payment Flight
• objetos físicos Orientação: É especialmente relevante na criação de software de controlo de dispositivos ou simulações.	Item, Register Airplane
• contentores de coisas (físicas ou informação)	Store, Bin Airplane
• coisas dentro de um “contentor”	Item Passenger
• histórico/registos contabilísticos, de trabalho, contratos, matéria jurídica	Receipt, Ledger MaintenanceLog
• instrumentos financeiros	Check, LineOfCredit TicketCredit
• horários, manuais, documentos que são regularmente referidos para a realização de trabalhos	DailyPriceChangeList RepairSchedule

II) Pesquisa de conceitos por análise textual

Outra técnica útil (e simples) é a análise linguística: identificar os substantivos nas descrições textuais de um domínio e considerá-los como classes conceptuais ou atributos candidatos

Alguns destes substantivos são classes conceptuais, alguns podem referir-se a classes conceptuais que são ignoradas nesta iteração, alguns podem ser atributos de classes, e outros serão para ignorar.

Um ponto fraco desta abordagem é a imprecisão da linguagem natural; substantivos diferentes

podem representar a mesma classe conceptual ou atributo, entre outras ambiguidades.

Fluxo Básico:

1. O Cliente chega a uma caixa POS com artigos para comprar.
2. Caixa inicia uma nova venda.
3. Caixa introduz o identificador do artigo.
4. O sistema regista a linha de venda e apresenta a descrição do item, o preço, e o total provisório. O preço é calculado a partir de um conjunto de regras de preços.

...