

Introdução à UML e Ferramentas para C++

AEDA@2019/2020



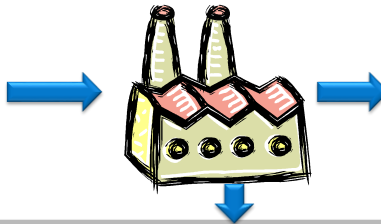
Índice

- Uma fábrica de software
- O ciclo de desenvolvimento
- O que é a UML?
- Modelos e diagramas:
 - Visão Geral
 - Diagrama de classes
 - Diagrama de sequências
 - Diagrama de atividades
- Ferramentas
 - Umbrello (linux)
 - Eclipse UML2
 - Visio
 - Enterprise Architect
- Exemplo de modelação
 - A descrição
 - A lista de candidatos a classes
 - A lista de candidatos a atributos
 - A lista de acções candidatas
 - Identificar relações entre classes
 - Diagrama de classes
 - Diagrama de sequências
- Comentar e Documentar
 - O que é e porquê fazê-lo?
 - O que é preciso para o fazer?

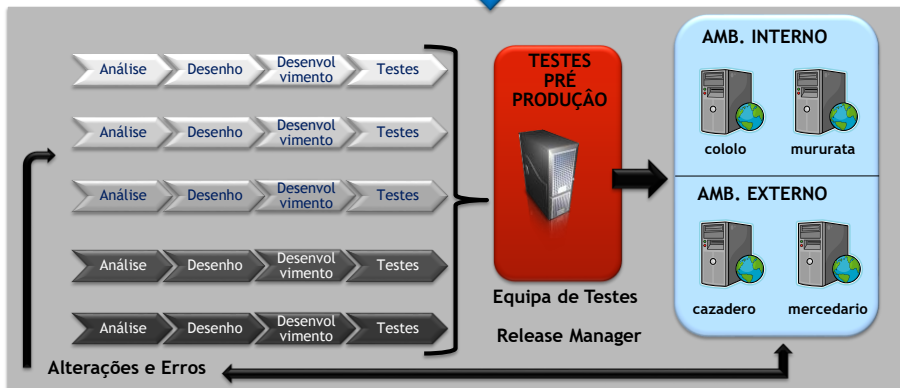


Fábrica de Software - Ciclo de Produção

Requisitos



Sistema



O Ciclo de Desenvolvimento



Compreender o
que o cliente
quer



Especificação Requisitos



Desenhar o
Sistema de
acordo com os
requisitos



Planta, Projeto, Plano



Implementar de
acordo com o
plano



O sistema (a casa)



+

=



?

O que é UML?

- UML = *Unified Modeling Language*
- UML é uma linguagem (notação com semântica associada) para
 - visualizar
 - especificar
 - construir
 - documentar

os artefactos de um sistema com uma componente intensiva de software (*software intensive system*)

- UML não é uma metodologia
 - não diz quem deve fazer o quê, quando e como
 - UML pode ser usado segundo diferentes metodologias, tais como RUP (*Rational Unified Process*), FDD (*Feature Driven Development*), etc.
- UML não é uma linguagem de programação

Valor da UML

- É um standard aberto
 - versão 1.1 aprovada pelo OMG (*Object Management Group*) em Novembro de 1997 (<http://www.uml.org/>)
 - versão 2.5.1 aprovada em Dezembro de 2017
- Suporta todo o ciclo de vida do software
 - modelação do negócio (processos e objetos do negócio)
 - modelação de requisitos alocados ao software
 - modelação da solução de software
- Suporta diversas áreas de aplicação
- É baseado na experiência e necessidades da comunidade de utilizadores
- É suportado por muitas ferramentas

Modelos e Diagramas

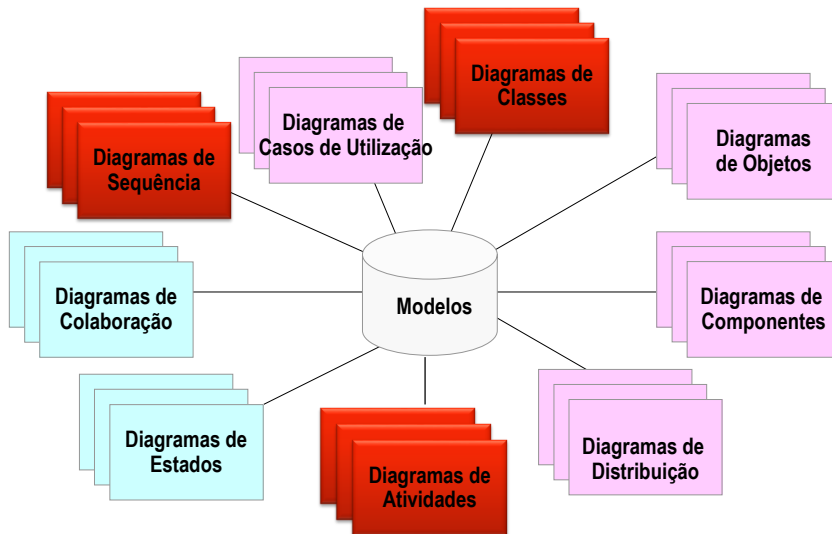
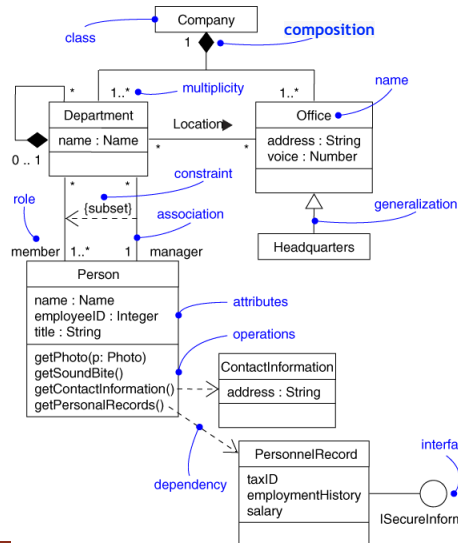


Diagrama de Classes

- Captura o vocabulário de um sistema
- Construído e refinado ao longo do desenvolvimento
- Objetivo
 - Nomear e modelar conceitos no sistema
 - Especificar colaborações
 - Especificar esquemas lógicos de bases de dados
- Desenvolvido por analistas, *designers* e implementadores



Diagrama de Classes

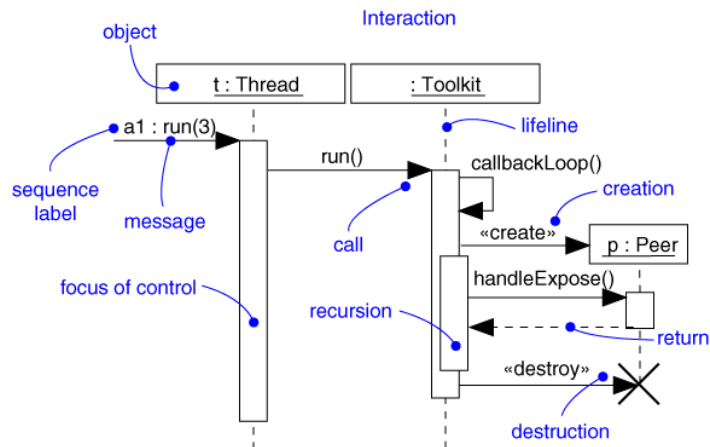


Fonte: Grady Booch

Diagrama de Sequência

- Captura comportamento dinâmico (orientado ao tempo)
- Objetivo
 - Modelar fluxos de controlo
 - Ilustrar cenários típicos

Diagrama de Sequência

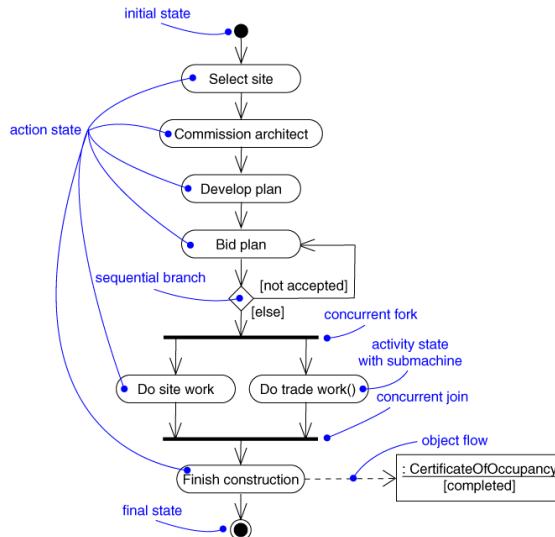


Fonte: Grady Booch

Diagrama de Atividades

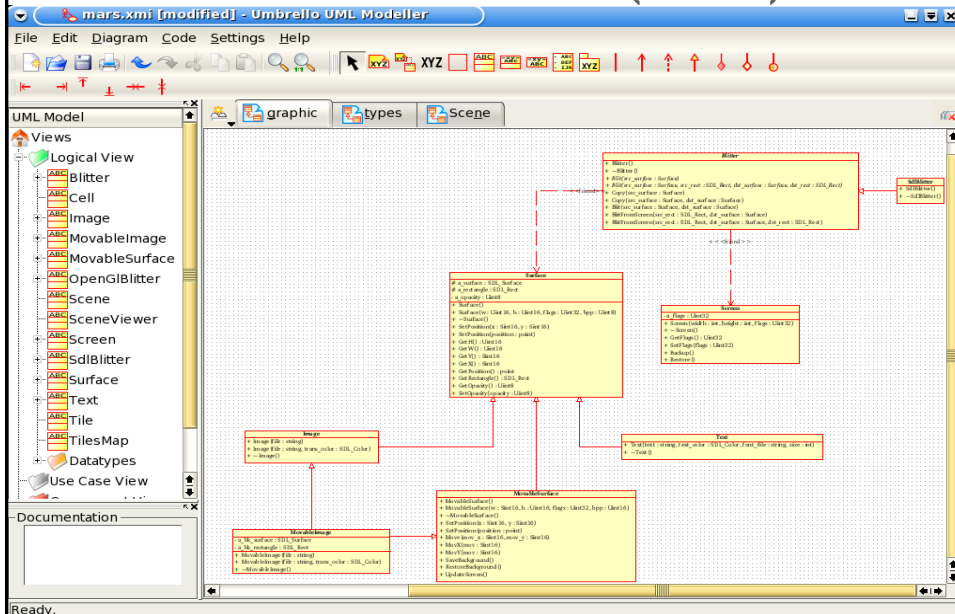
- Captura comportamento dinâmico (orientado a atividades)
- Objetivo
 - Modelar processos de negócio e *workflows*
 - Modelar operações (algoritmos)

Diagrama de Atividades

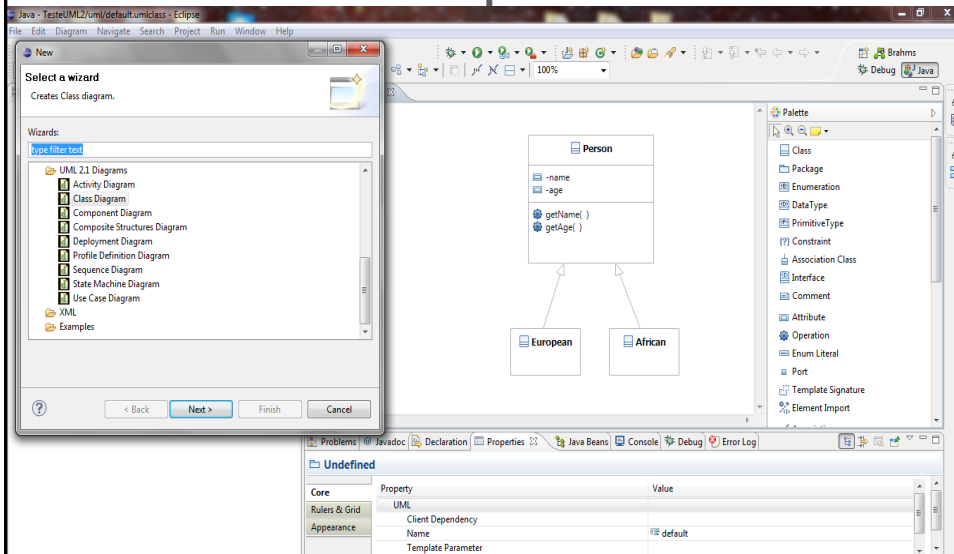


Fonte: Grady Booch

Ferramentas - Umbrello (Linux)

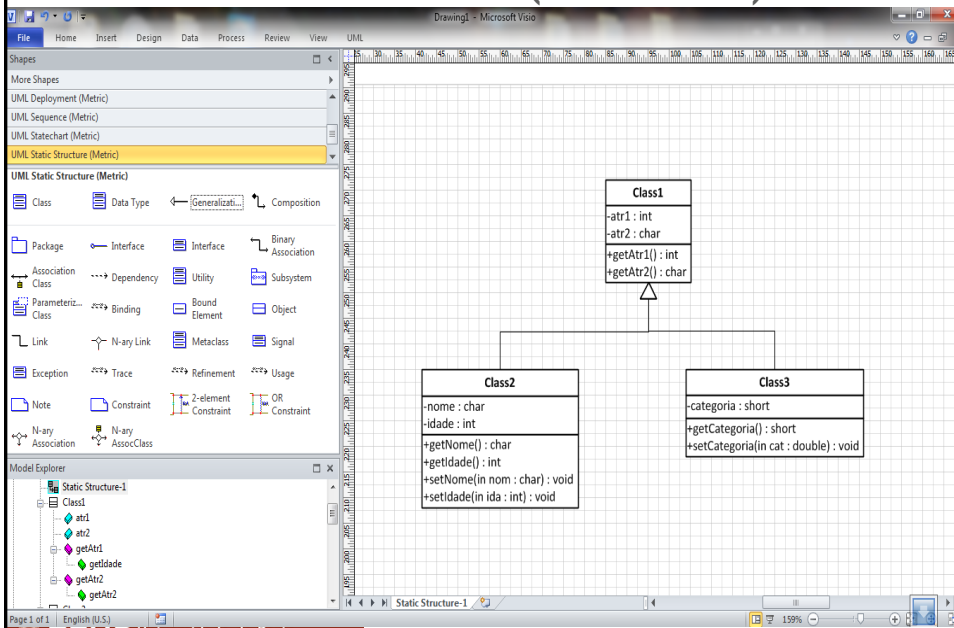


Ferramentas - Eclipse UML2



<http://www.vogella.de/articles/UML/article.html>

Ferramentas - Visio (Microsoft)



Ferramentas - Enterprise Architect

The screenshot displays the Enterprise Architect software interface. The main window shows a UML class diagram for the 'AirTrafficControl' class. The diagram includes several associated classes and their relationships: 'EnrouteAircraftMalfunction', 'FlightDepartureDelay', 'CrewDelay', 'LoadingDelay', 'PassengerDelay', 'WeatherConditions', 'AircraftMalfunction', and 'CrewNoShow'. A 'Class: AirTrafficControl' dialog box is open in the foreground, showing properties such as Name, Stereotype, Author, Scope, Complexity, Language, and Version. The background shows a project browser with a hierarchy of models and a properties window for the selected class.

FEUP Universidade do Porto
 Faculdade de Engenharia

Introdução ao UML e Ferramentas para C++ - Outubro de 2016

Exemplo

“Um tripulante [*de um dado voo de uma dada companhia aérea*], após ter efetuado o login, deverá ter acesso a várias opções de consulta, que serão mostradas de acordo com o seu perfil, isto é, dependendo da sua categoria profissional e da base a que pertence. Escolhendo a opção consultar escala, deverá indicar a data de início e de fim do período que quer consultar. Deverá aparecer no monitor a informação dos voos e atividades, que o tripulante realiza nesse período. Deverá constar a data, número do voo, hora de saída, aeroporto de partida e de chegada e hora esperada da chegada, para além do avião que realiza o voo. A duração do voo deverá ser calculada automaticamente, assim como o seu tipo (longo curso ou médio curso). Se for uma atividade, deverá aparecer a data e hora de início e de fim, bem como uma descrição da mesma.”

Partindo desta descrição, como é que modelariam um sistema ou funcionalidade, que fizesse o que é pedido?

Exemplo

“Um **tripulante**, após ter efetuado o login, deverá ter acesso a várias opções de consulta, que serão mostradas de acordo com o seu **perfil**, isto é, dependendo da sua **categoria** profissional e da **base** a que pertence. Escolhendo a opção consultar **escala**, deverá indicar a **data de início e de fim** do período que quer consultar. Deverá aparecer no monitor a informação dos **voos** e **atividades**, que o **tripulante** realiza nesse período. Deverá constar a **data**, **número do voo**, **hora de saída**, **aeroporto de partida** e de **chegada** e **hora esperada da chegada**, para além do **avião** que realiza o **voo**. A **duração do voo** deverá ser calculada automaticamente, assim como o seu **tipo** (**longo curso** ou **médio curso**). Se for uma **atividade**, deverá aparecer a **data e hora de início e de fim**, bem como uma **descrição** da mesma.”

(1)

procurar SUBSTANTIVOS que correspondam a entidades ou eventos



Fazer uma lista de candidatos a classes e outra de candidatos a atributos das classes

<u>Classes</u>	<u>Atributos</u>	<u>Atributos</u>
1. Tripulante	.Categoria (1, 2, 3)	.Aeroporto partida (6, 8)
2. Perfil	.Base (1, 2, 4)	.Aeroporto chegada (6, 8)
3. Categoria	.Data inicio (7)	.Avião voo (6, 9)
4. Base	.Data fim (7)	.Descrição (7)
5. Escala	.Data voo (6)	.Duração voo (6)
6. Voo	.Número voo (6)	.Tipo voo (6)
7. Atividade	.Hora saída (6)	
8. Aeroporto	.Hora chegada (6)	
9. Avião		

Associar cada atributo a uma classe



2) Identificar frases com verbos Sugerem comportamentos -> ações -> métodos

“Um tripulante, após ter efetuado o login, deverá ter acesso a várias opções de consulta, que serão mostradas de acordo com o seu perfil, isto é, dependendo da sua categoria profissional e da base a que pertence. Escolhendo a opção consultar escala, deverá indicar a data de início e de fim do período que quer consultar. Deverá aparecer no monitor a informação dos voos e atividades, que o tripulante realiza nesse período. Deverá constar a data, número do voo, hora de saída, aeroporto de partida e de chegada e hora espera da chegada, para além do avião que realiza o voo. A duração do voo deverá ser calculada automaticamente, assim como o seu tipo (longo curso ou médio curso). Se for uma atividade, deverá aparecer a data e hora de início e de fim, bem como uma descrição da mesma.”

Sugerem comportamento -> ações -> métodos



Fazer uma lista de ações candidatas

1. Efetuar Login (**Menu**) (*)
2. Acesso opções consulta (**Menu**) (*)
3. Mostrar acordo perfil (**Menu**, **Perfil**, **Base**, **Categoria**)
4. Indicar data inicio e fim (**Escala**)
5. Mostrar voos (**Escala**, **Voo**, **Aeroporto**, **Avião**)
6. Mostrar actividade (**Escala**, **Voo**)
7. Calcular duração voo (**Voo**)
8. Definir tipo voo (**Voo**)
9. Mostrar descrição (**Actividade**)

Associar cada ação (comportamento) a entidade(s) (Classe(s))

Nesta fase podem aparecer classes novas como as indicadas com (*)



3) Identificar as relações entre classes

Associação

Aquilo que uma classe “precisa de saber” acerca da outra.

Ex: a classe Tripulante precisa de saber a Base

Generalização/Especialização

Também conhecidas como relações “is-a”. A classe *supertype* é geral porque contém os atributos e comportamentos comuns da hierarquia. A classe *subtype* é especializada porque contém atributos e comportamentos únicos da própria, mas herda os atributos e comportamentos da *supertype*.

Descobrem-se olhando para o diagrama de classes e verificando dois aspetos:

- Fazendo a seguinte pergunta para as classes que têm multiplicidade de 1:1: “O objecto X é um tipo do objecto Y?” Se for, então podemos ter uma relação generalização/especialização
- Procurando as classes que tenham atributos e comportamentos comuns.

Agregação/Composição

- Do tipo “has-a”, em que o objeto A contém o objeto B, e B é parte do objeto A (na composição B não existe sem o A. Na agregação, pode existir)

Exercício

Construa agora o modelo conceptual (Diagrama de Classes) para o exemplo enunciado.

Escolha alguns aspetos dinâmicos e construa os respetivos diagramas de Sequência e Atividades.

Associar cada ação (comportamento) a entidade(s) (Classe(s))

Nesta fase podem aparecer classes novas como as indicadas com (*)

Comentar e Documentar o Código

- Porquê comentar o código?
 - Não estamos sozinhos a desenvolver código. Outra pessoa, mais tarde ou mais cedo, vai fazer manutenção ao código que desenvolvemos. Os comentários, quando bem feitos, são uma grande ajuda que contribui para a qualidade do código. E,.....
 - Passados uns meses, já não nos lembramos do que fizemos. Os comentários ajudam-nos a recuperar a memória. Pouparamos tempo, trabalho e contribuímos para a qualidade do código.
- Afinal, como é que comento o código?
 - Utilizando *blocos de documentação especiais* do tipo `/** ... */` colocados no início das classes, atributos, métodos
 - Utilizando *tags* dentro desses blocos de documentação. Por exemplo:
`@param`
- Documentar é a mesma coisa que comentar?
 - Não, embora os comentários aparecem na documentação.
 - Documentar significa apresentar todas as classes, atributos e métodos num documento de consulta fácil (html, pdf), com diagramas e pesquisa

O que é necessário para documentar?


- Software Doxygen
 - <http://www.stack.nl/~dimitri/doxygen/>
- Eclox Plug-in para o eclipse (ajuda bastante....)
 - <http://home.gna.org/eclox/>

Comentar e Documentar FP02

Questões?

Rosaldo J. F. Rossetti
rossetti@fe.up.pt

António J. M. Castro
ajmc@fe.up.pt

 **FEUP** Universidade do Porto
Faculdade de Engenharia

Introdução ao UML e Ferramentas para C++ - Outubro de 2016

<#>