Tecnologia de Objectos

Ademar Aguiar
www.fe.up.pt/~aaguiar
ademar.aguiar@fe.up.pt

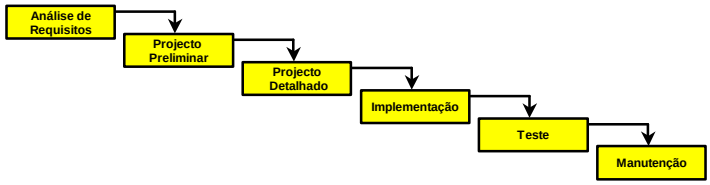
Universidade do Porto
Faculdade de Engenharia
FEUP

Tecnologia de Objectos

- O que é orientação por objectos ?
- Que tipo de produtos existem ?
- O que é software de qualidade ?
- Porquê é tão difícil construir software de qualidade ?
- Como se desenvolve software orientado por objectos ?

Engenharia de Software

- A engenharia de software é o ramo de engenharia que visa o desenvolvimento de sistemas de **software de qualidade**
- Os **modelos** do ciclo de vida do desenvolvimento de software compreendem várias fases bem definidas. Exemplo: "modelo clássico"



Qualidade de Software

- Qualidade é uma noção multifacetada descrita por um conjunto de **factores externos** e **internos**
- O software de qualidade deve ser:
  - **correcto**,
  - fácil de usar,
  - **robusto**,
  - eficiente,
  - expansível,
  - portável,
  - **reutilizável**,
  - verificável,
  - compatível,
  - íntegro,
  - **modular** (interno),
  - compreensível (interno)

Complexidade do Software

- Sistemas de software simples e complexos
  - Os sistemas especificados, construídos, mantidos e usados por uma mesma pessoa, são normalmente **sistemas simples**
  - Os sistemas com que nos deparamos na indústria de software são normalmente **sistemas complexos**
  - A compreensão de todas as subtilezas de um sistema complexo é muito difícil de conseguir por uma só pessoa
  - Os sistemas complexos quase sempre excedem a capacidade intelectual humana
  - A complexidade diz-se ser uma propriedade inerente ao software porque, apesar de contornável, não é eliminável

Complexidade do Software...

- Porque razão é o software inerentemente complexo?
  - A complexidade do **domínio do problema**
  - A dificuldade na **gestão do processo de desenvolvimento**
  - A **flexibilidade possível** no software
  - A dificuldade em caracterizar o comportamento de sistemas **discretos**
- Consequências de não restringir a complexidade
  - Quanto mais complexo é um sistema mais sujeito está a ruptura
  - Projectos terminados para além dos prazos e custos previstos
  - Projectos que não cumprem na íntegra os requisitos iniciais

Sistemas Complexos

- Cinco Atributos
  - Hierarquia de vários subsistemas interrelacionados, até aos componentes primitivos
  - A escolha dos componentes primitivos dum sistema é arbitrária, e é subjectiva ao observador
  - Os sistemas hierárquicos são compostos por poucos tipos de subsistemas diferentes, mas dispostos em combinações várias
  - É possível separar as relações que envolvem a estrutura interna dos componentes, das relações entre componentes
  - **Um sistema complexo que funcione é fruto da evolução dum sistema mais simples que funcionava**

Técnicas para Organizar o CAOS

- **Decomposição** ("dividir para reinar")
  - Decomposição algorítmica: ilustra a ordem dos eventos
  - Decomposição orientada por objectos: mostra quais os agentes activos ou passivos nos eventos
- **Abstracção**
  - Organização dos estímulos em várias dimensões e sucessivamente em sequências de blocos de informação
  - Consiste em ignorar detalhes não essenciais para o problema
- **Hierarquia**
  - A hiearquia de objectos (agregações) ilustra os padrões de interacção entre objectos diferentes (mecanismos)
  - A hierarquia de classes (generalizações) evidencia a redundância do sistema

A Importância dos Modelos

Um modelo é uma abstracção de uma entidade real ou não, que tem o propósito de promover a sua compreensão antes de se iniciar a sua construção ou modificação

- Os modelos:
  - Omitem detalhes não essenciais que os tornam mais fáceis de tratar do que a entidade original
  - Facilitam a Percepção de Sistemas Complexos
  - Auxiliam-nos a lidar com a complexidade, porque a sua construção apela aos princípios da **decomposição**, **abstracção** e **hierarquia**, as principais ferramentas mentais que temos
  - Os modelos permitem-nos falhar em situações controladas

Orientação por Objectos

Orientação por Objectos

- Orientação por objectos (OO) é bem mais do que um paradigma de programação
- Aplica-se a várias actividades relacionadas com tecnologias de informação: análise de requisitos, processos, desenho de software, construção e teste de sistemas de software
- A ideia central consiste na construção de modelos de sistemas em torno de entidades que unificam dados e procedimentos: **objectos**

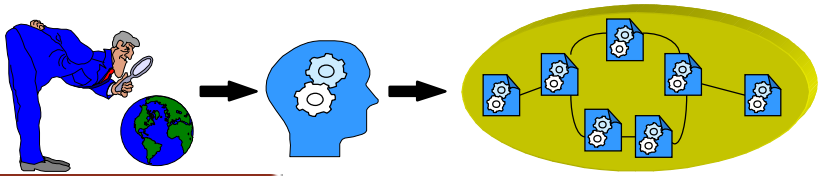
Evolução do Modelo Objecto

- 1ª Geração (1954-1958)
  - A abstracção introduzida foi a das expressões matemáticas.
  - Exemplos: FORTRAN I, ALGOL 58, IPL V.
- 2ª Geração (1959-1961)
  - Introduzidas abstracções algorítmicas.
  - Exemplos: FORTRAN II, ALGOL 60, COBOL, LISP
- 3ª Geração (1962-1970)
  - Descrição de tipos de dados complexos, com implementação deixada ao cuidado da linguagem de programação.
  - Exemplos: PL/1, ALGOL 68, Pascal, Simula.
- Última Geração (1970-1996)
  - Linguagens baseadas em objectos e orientadas aos objectos.
  - Exemplos: Eiffel, C++, Smalltalk, e Ada, Java.

Objectos

"Um **objecto** é algo que possui **estado**, **comportamento** e **identidade**, e pode ser definido de uma forma bastante precisa"

- A **estrutura** e o **comportamento** de objectos idênticos são definidos na sua **classe** comum
- Cada objecto é uma **instância** de uma classe
- Na abordagem OO, as entidades do mundo real são vistas como uma colecção de objectos cooperantes



FEUP Universidade do Porto Faculdade de Engenharia MRSC, Programação em Comunicações 13

Objectos ...

- Os objectos correspondem directamente a coisas, pessoas, ou processos existentes no mundo real

- Num modelo orientado por objectos, as características do mundo real são representadas por **atributos** e os comportamentos por **operações**

FEUP Universidade do Porto Faculdade de Engenharia MRSC, Programação em Comunicações 14

Princípios OO Fundamentais

- Abstracção...
  - Ilustra as características essenciais de um objecto
- Encapsulamento...
  - Oculta os detalhes de um objecto
- Modularidade...
  - É a propriedade de um sistema que foi decomposto num conjunto de módulos coesos e fracamente ligados
- Hierarquia...
  - Ordena as abstracções de um sistema
- Tipos...
  - Associam os objectos às suas classes, restringindo sua troca
- Concorrência...
  - É a propriedade que distingue objecto activos de passivos
- Persistência...
  - É a propriedade de um objecto através da qual a sua existência transcende o tempo e/ou espaço

Vantagens da OO

- O modelo de objectos é diferente dos modelos seguidos pelos métodos mais tradicionais de análise, projecto e programação
- Introduz elementos novos que assentam nos antigos
- Orienta na construção de sistemas que incorporam os cinco atributos dum sistema complexo
- Auxilia na exploração do poder expressivo das linguagens de programação orientada por objectos
- O uso do modelo de objectos incentiva a reutilização, não apenas de software (bibliotecas de classes), mas também de projectos inteiros (frameworks)
- O uso do modelo de objectos produz sistemas construídos sobre formas intermédias estáveis, por conseguinte mais flexíveis a alterações e mais capazes de resistir ao tempo
- O modelo de objectos reduz o risco de desenvolvimento de sistemas complexos

Processos de Desenvolvimento de Software

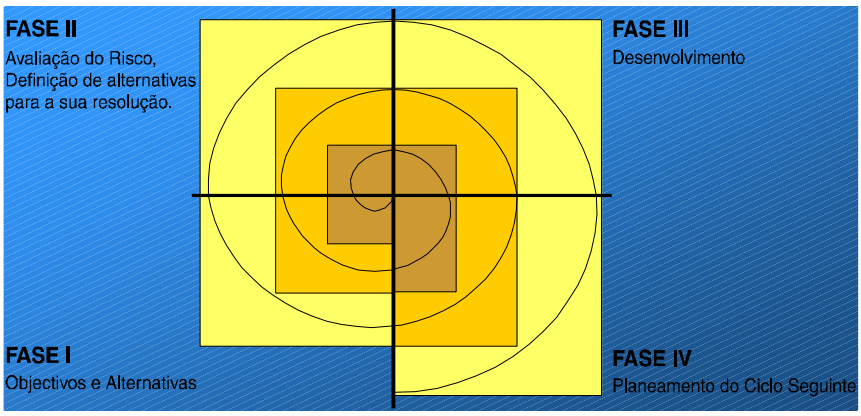
Processos de Desenvolvimento

- O desenvolvimento de software é um processo que começa com a definição dos requisitos iniciais e se prolonga até às fases de teste e manutenção do sistema
- O ciclo de vida de software OO prevê uma maior interligação entre as várias fases, adaptando-se assim mais facilmente à **natureza iterativa e incremental** do processo

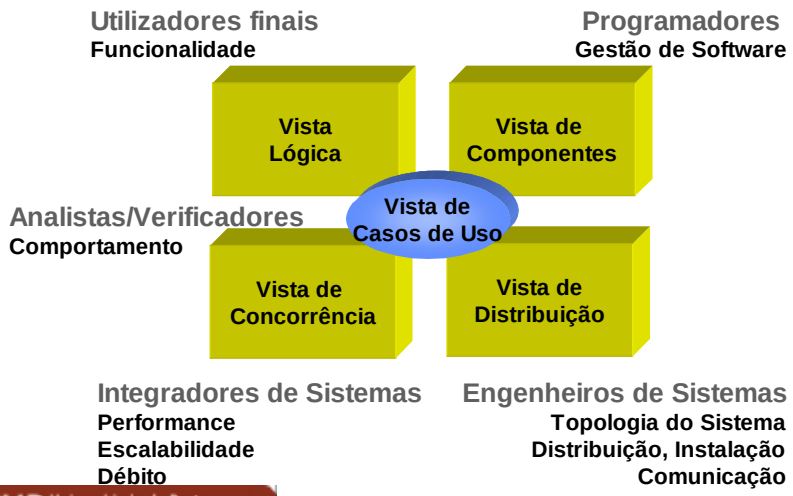
Processos de Desenvolvimento...

- Aumentam a produtividade e sucesso
  - A presença de um processo **bem definido** e **bem gerido** é um aspecto determinante de diferenciação entre projectos produtivos e projectos mal-sucedidos
- Vantagens
  - **Orienta** as diversas actividades dos elementos de uma equipa
  - **Especifica** os artefactos que devem ser desenvolvidos
  - **Dirige** as tarefas de elementos individuais e de uma equipa como um todo
  - Oferece critérios para **monitorização e avaliação** dos produtos e actividades de um projecto
- Iterativos e Incrementais: os MELHORES!

Processo Iterativo e Incremental



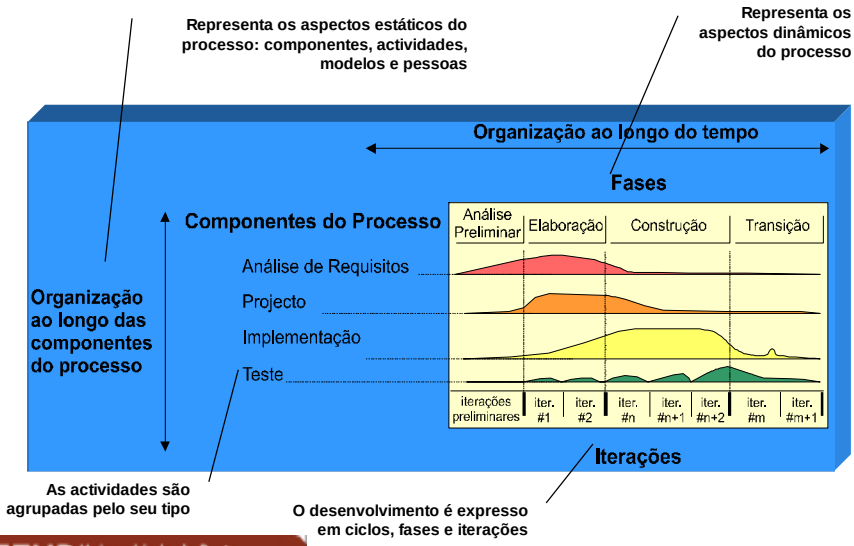
"Unified Process"



"Unified Process"...

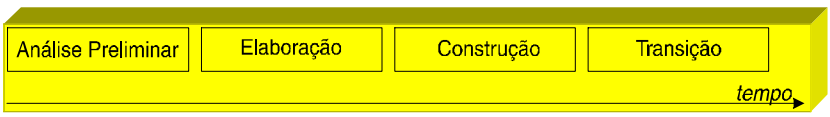
- É sustentado em **UML**
- Suporta técnicas de orientação por objectos
- É um processo iterativo e incremental.
- Promove a definição inicial de uma **arquitectura de software robusta**, que facilita a paralelização, reutilização e manutenção.
- A identificação de **casos de uso** e cenários típicos de utilização é a actividade que conduz todo o processo de desenvolvimento, desde a análise de requisitos até ao teste do sistema final.

"Rational Unified Process"



Fases do Processo

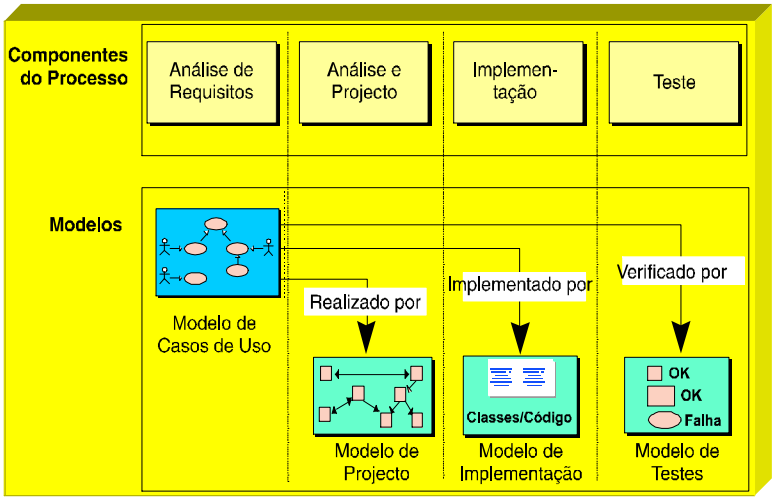
- **Iterações**
  - Cada uma das fases anteriores pode ainda ser decomposta em iterações.
  - Uma iteração é um ciclo de desenvolvimento completo cujo resultado pode ser um produto executável, ou um subproduto do produto final, que incrementalmente cresce de iteração para iteração até ao sistema final.
  - Cada iteração abrange todas as componentes do processo de desenvolvimento, embora com ênfases diferentes em cada componente consoante a fase a que corresponde.



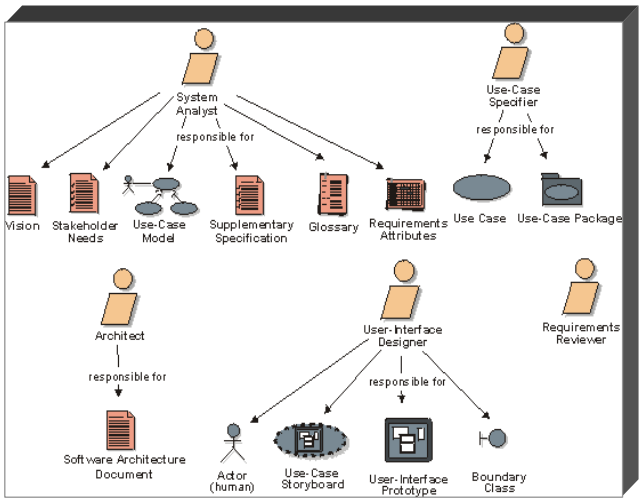
Componentes do Processo...

- **Implementação**
  - O objectivo é construir o sistema, produzindo todo o código necessário para a criação do sistema executável.
  - Os modelos de projecto são a base da implementação.
  - A implementação inclui o teste de classes e módulos separados, mas não a verificação do seu funcionamento integrado.
- **Teste**
  - O objectivo é verificar o sistema na sua totalidade.
  - Inicialmente verifica-se cada caso de uso separadamente e posteriormente o sistema na sua totalidade.
  - No final desta componente, o sistema está pronto para ser utilizado.

Componentes do Processo...

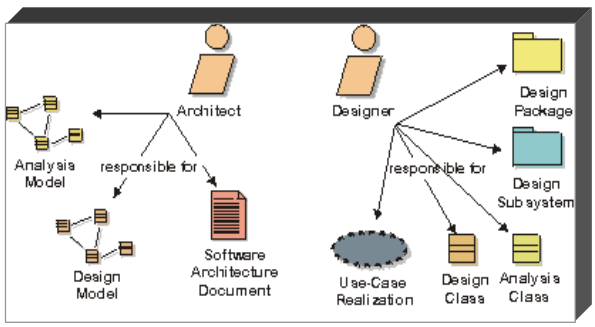


RUP: Análise de Requisitos



from **Rational Unified Process**

RUP: Análise e Projecto



from **Rational Unified Process**

Processos: reflexões

- Os processos têm que ser configurados
  - Todas as organizações, pessoas e projectos são diferentes
  - Para serem úteis, têm que ser adaptados à realidade em mãos
- Compreensão, Disciplina e Competência
  - São factores cruciais para o sucesso em software
- Cuidados ao configurar um processo
  - "Documentar" não basta para "Compreender"
  - "Formalidade" não é equivalente a "Disciplina"
  - "Processo" não é equivalente a "Competência"

Visão Geral sobre a Tecnologia

Programação, Metodologias

- Linguagens de programação orientadas por objectos
  - Smalltalk, Ada, C++, Objective-C, Eiffel, Java, 1980-1994
- Metodologias de análise e projecto orientado por objectos
  - Shlaer/Mellor - análise de requisitos, 1988
  - Grady Booch - análise e desenho, 1991
  - James Rumbaugh - análise e desenho, 1991
  - Ivar Jacobson - modelos de processos, reengenharia, 1992
  - Booch, Rumbaugh, Jacobson "UML-Unified Modeling Language", - análise e projecto, v1.0/jan97, v1.3/jun99

Bases de Dados, CASE's...

- Sistemas de gestão de bases de dados orientadas por objectos, e bases de dados objecto-relacionais
  - Gemstone (1987), *servio corporation*
  - O_2 (1989), *O_2 technology*
  - Objectstore (1990), *object design*
- Ferramentas CASE-OO e visual modelers
  - System architect
  - Paradigm Plus, ex-Platinum
  - Rational'2000, Rational Software Corporation
  - Visio 5.0 (visual modeler)
  - Together Java/C++, TogetherSoft

Sistemas Distribuídos...

- Objectos distribuídos, interoperabilidade, portabilidade
  - OMG-*object management group*
    - Consórcio industrial criado em 1989 por 3Com, Canon, Data General, HP, Philips, Sun Microsystems, entre outras
    - Actualmente tem mais de 600 membros, o que torna a OMG o maior consórcio mundial de desenvolvimento de software
  - ODMG-*object database management group*
    - Criado em 1989, representa 90% do mercado existente de sgbdo's
    - Object Design, Objectivity, O_2 Technology, Versant, Ontos, Servio, liderado por Rick Cattell da Sunsoft
  - Alguns documentos importantes
    - "Common object request broker architecture (CORBA)"
    - "ODMG-93, ODMG-3.0: The object database standard", 1993-2000

Tópicos OO

- Engenharia de Requisitos
- Linguagens de Programação
- Notações para Modelos de Objectos
- Ferramentas CASE
- Processos de Desenvolvimento de Software
- Sistemas de gestão de bases de dados orientadas por objectos e objecto-relacionais
- Objectos distribuídos, interoperabilidade, portabilidade
- Padrões de Software
- Frameworks Orientadas por Objectos
- Componentes de software
- Arquitectura de Sistemas de Software OO
- Arquitectura de Linhas de Produção de Software OO

Desafios Futuros

- Aspectos positivos:
  - As aplicações de software aumentarão bastante a sua sofisticação
  - O software constituirá para cada vez mais organizações um elemento estratégico de desenvolvimento
  - O software estará cada vez mais profundamente enraizado no nosso quotidiano
  - Existirá uma procura insaciável de software

  => Há que saber responder com Qualidade!

Desafios Futuros ...

- Aspectos "não-positivos":
  - O software no futuro será cada vez mais complexo
  - O software terá de se adaptar à cada vez maior conectividade entre sistemas de computação distribuídos
  - O software terá de ser capaz de satisfazer as expectativas de melhor visualização e melhor acesso aos dados
  - Muito software será desenvolvido sem programar, recorrendo a ambientes de programação visual e componentes já existentes, abordagem que é limitada em termos de complexidade admissível

Deverá haver uma preocupação acrescida na definição da **arquitectura** dos sistemas de software e nos **processos** de desenvolvimento adoptados.

Fim