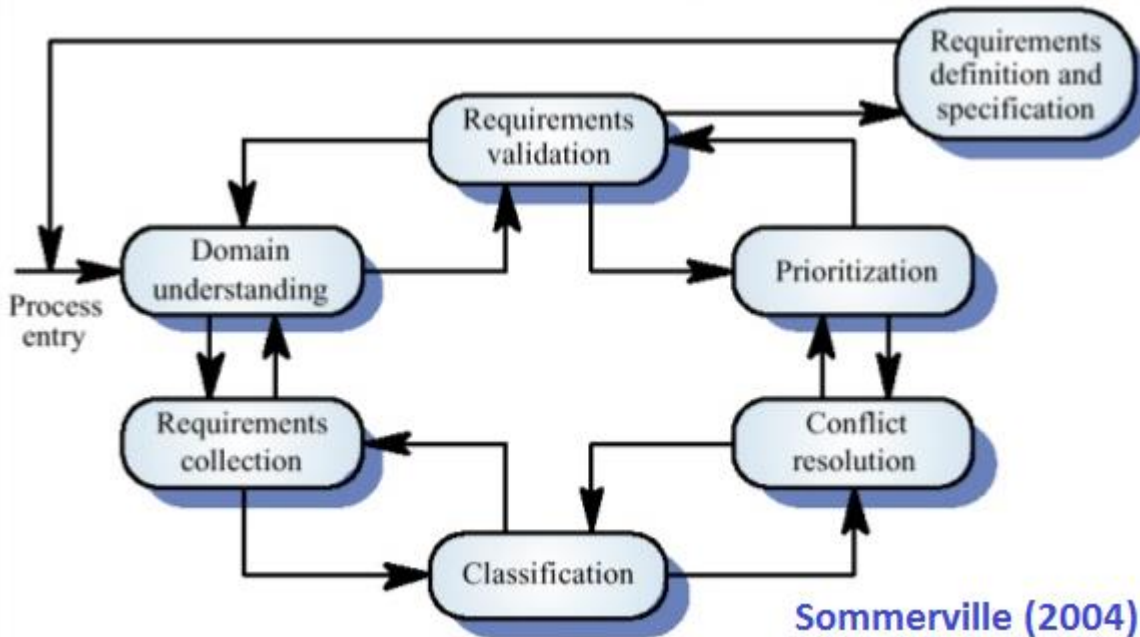




UniCesumar

EDUCAÇÃO PRESENCIAL E A DISTÂNCIA

Requisitos de Software




O que é um requisito?

- Característica, atributo, habilidade ou qualidade que um sistema deve prover para ser útil a seus usuários
- Os requisitos devem informar “o que deve ser feito ou atendido” para resolver o problema do usuário

- Base para o projeto (design)
- Mas, não consideramos a solução técnica

O que é um requisito?

- 
- Scripts SQL
 - Modelos ER físicos
 - Padrões de projeto
 - Estruturas internas de dados

Quem é responsável pelos requisitos?

- O Engenheiro de Software
- Profissional que deve ter a habilidade de antecipar e gerenciar mudanças de requisitos de um produto de software.
- Saber se expressar.
- Comunicar-se bem a fim de capturar e registrar adequadamente o documento de requisitos.

Quem são os fornecedores de requisitos?

- Quem tem conhecimento de uso ou de negócio para orientar a construção do produto?
- Quem sabe como o sistema funciona?
- Quem conhece as regras do negócio?

Classificação de requisitos

- Tipo
 - Requisitos Não Funcionais
 - Requisitos Funcionais
- Origem
 - Requisitos de Usuário
 - Requisitos de Sistema

Requisitos

- Voz ativa e no afirmativo
- Requisito é único
- Requisito não contradiz outro requisito
- É possível interpretar o requisito apenas de uma maneira
- Devem ser verificáveis

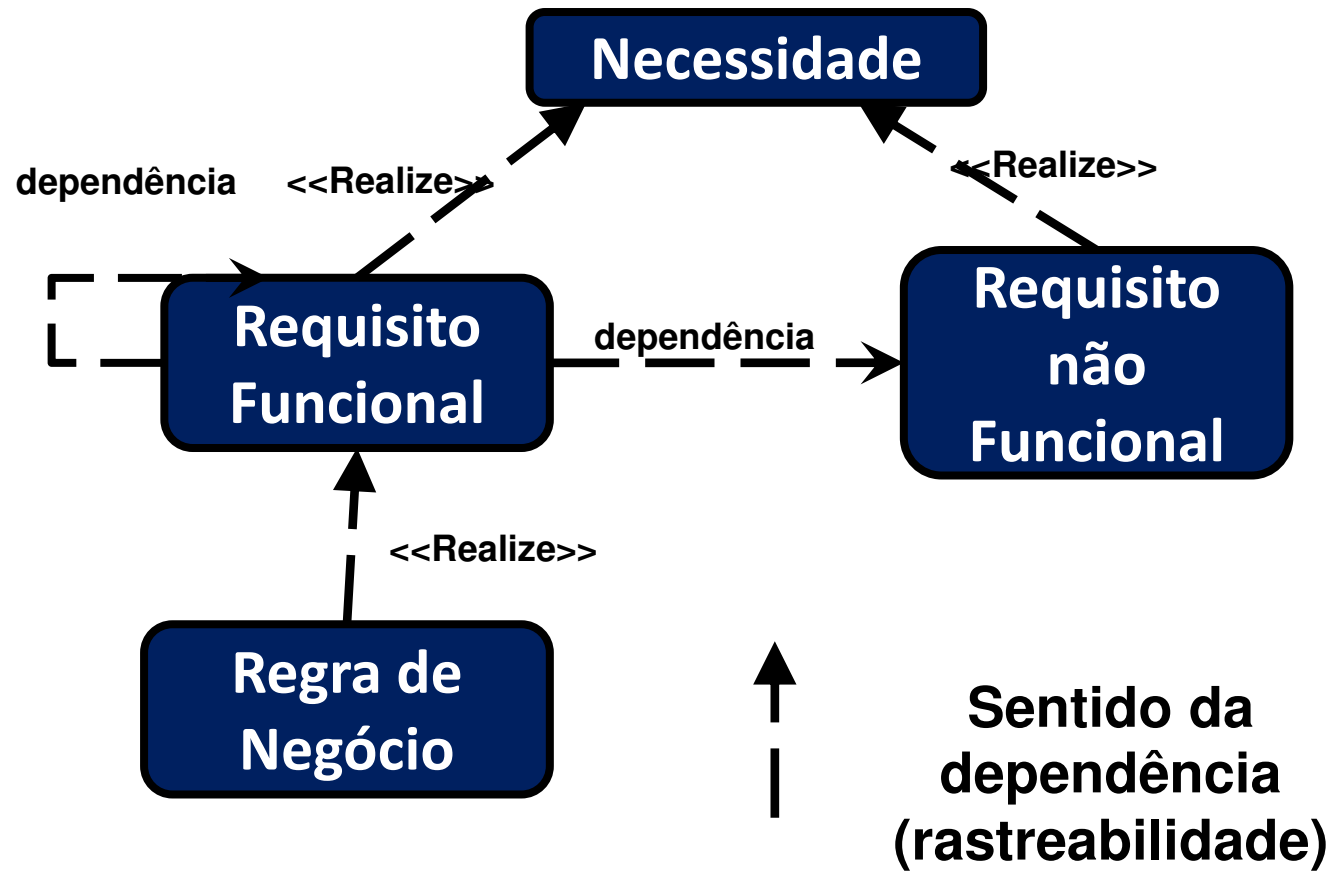


Dilbert by Scott Adams, 2006

Necessidade

- Representa algo que o usuário/cliente “precisa” para resolver um problema.
- Declarações com forte significado para o stakeholder.
- Foco naquilo que o sistema irá resolver e não no que o sistema irá fazer.
- Não confundir necessidades com requisitos funcionais

Requisitos



Necessidade

- Representa algo que o usuário/cliente “precisa” para resolver um problema.
- Declarações com forte significado para o stakeholder.
- Foco naquilo que o sistema irá resolver e não no que o sistema irá fazer.
 - Não confundir necessidades com requisitos funcionais.

Requisitos

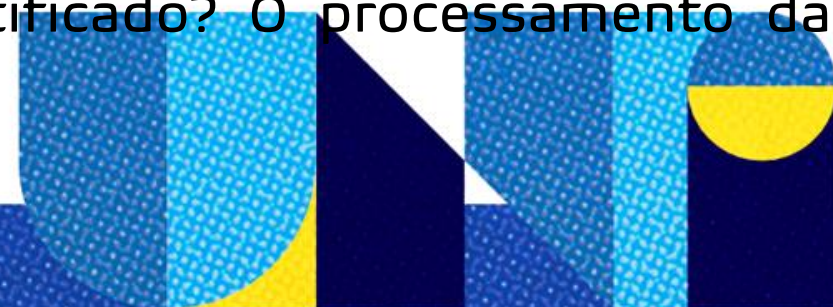
NE<NN> - <Descrição da necessidade>
onde ***<NN>*** representa um número sequencial.

Vamos analisar este exemplo:

NE001 - O sistema deve processar planilhas financeiras

NE001 - O sistema deve processar planilhas financeiras

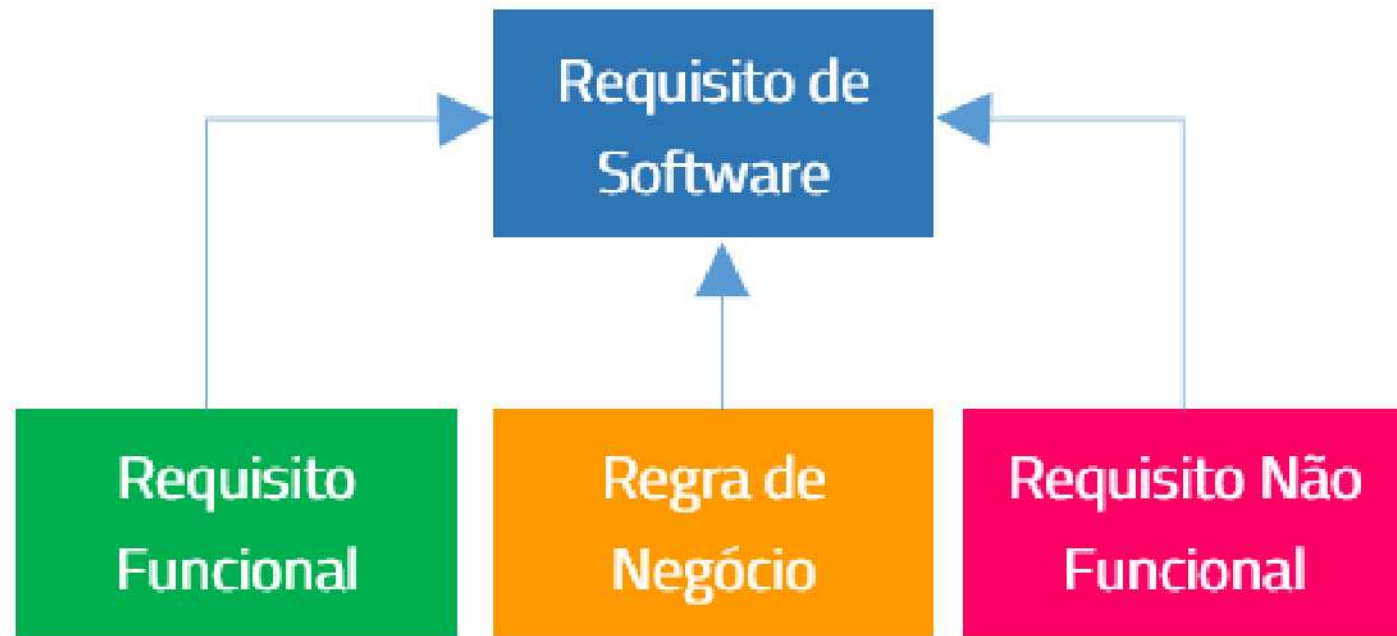
- Texto voltado para a solução, se aproximando de um requisito funcional
- Foco no que deve ser feito e não no problema do usuário
- Por que o usuário/cliente quer processar estas planilhas?
- A resposta é porque ele precisa calcular as despesas financeiras por categoria da sua empresa
- Qual o benefício que o usuário/cliente terá com o processamento das planilhas?
- A resposta é que o cálculo será feito automaticamente, aumentando a velocidade de resposta e eliminando erro humano.
- O usuário também deseja guardar um histórico dos cálculos para conferência futura.
- Existem outros modos de resolver o problema identificado? O processamento das planilhas é uma possível solução, mas é a melhor?





NE001 - Aumentar a velocidade dos cálculos, guardar histórico e eliminar erro humano no cálculo.

Requisitos Funcionais



facebook.com/ateomomento
www.ateomomento.com.br

Requisito Funcional

- Expressam funcionalidades ou serviços que um sistema deve ou pode ser capaz de executar ou fornecer
- Ações:
 - Imprimir, Calcular, Gerar
 - Exportar, Importar, Consultar
 - Mostrar, Enviar, ...

Requisito Funcional

Forma 1	RF<MM><NNN> - O sistema deve permitir <descrição da função executada por algum usuário do sistema>
Quando usar	Note que o verbo “ permitir ” deve ser usado aqui para indicar que a função será executada por um usuário do sistema
Forma 2	RF<MM><NNN> - O sistema deve <descrição da função executada pelo sistema>
Quando usar	Note que a ausência do verbo “permitir” indica que a função será executada diretamente pelo sistema, sem a interação direta com um usuário.

RF	Prefixo adotado para representar um requisito funcional
<MM>	Número sequencial com dois dígitos, identificando o módulo que o requisito pertence.
<NNN>	Número sequencial com três dígitos, utilizado para a identificação do requisito dentro de cada módulo.

Requisitos não funcionais

- Declaram restrições ou atributos de qualidade
- Categorias:
 - Segurança
 - Desempenho
 - Usabilidade
 - Confiabilidade
 - Portabilidade
 - Padrões e normas

Requisito não Funcional

Forma	RNF<NNN> - O sistema deve permitir <descrição da restrição ou condição de qualidade a ser atendida pelo sistema >
-------	---

RNF	Prefixo adotado para representar um requisito não funcional
<NNN>	Número sequencial com três dígitos, utilizado para a identificação única do requisitos não funcional.



Revisando

- RNF007. O sistema deve funcionar nos browsers IE8 e IE9.

Regra de negócio

- Afirmações que definem ou restringem o negócio
 - O objetivo é definir a estrutura ou o comportamento do negócio
 - Elas complementam o entendimento sobre os requisitos e detalham o "como" do ponto de vista do negócio
- Podem ser cálculos, deduções, validações ou restrições que devem ser consideradas na execução dos processos existentes em uma organização
- Elas podem ser leis e regulamentos impostos ao negócio, ou mesmo regras específicas a um determinado caso de uso



Documentar

Forma	<i>RN<NNN> - <descrição>.</i>
-------	--

RNF	Prefixo adotado para representar uma regra de negócio
<NNN>	Número sequencial com três dígitos que identifica a regra de negócio.



Vejam os o exemplo

- RN01.003 – O preço líquido de um produto é calculado como:
- $\text{Preço Líquido} = \text{Preço Produto} - (\text{Preço Produto} * \text{Porcentagem de imposto} / 100)$
- Onde:
 - Preço Líquido: preço do produto já descontado os impostos.
 - Preço Produto: preço do produto com todos os impostos inclusos.
 - Porcentagem de Imposto: valor percentual do imposto pago ao governo sobre o produto.
- Exemplo: considerando que o preço do produto é R\$ 10,00 e o imposto sobre este é de 5%, o preço líquido é de R\$ 9,50



Especificação

[illegible]

Especificação História de Usuário

- Como um profissional de saúde eu quero poder incluir mais de um CID em um único atendimento
 - RF001 CID
Deve permitir incluir 1 ou mais CID no atendimento
 - RF002 - Relatório de final de atendimento
Deve permitir listar todos os CID no relatório de finalização do atendimento
 - RF003 - Histórico do atendimento
Deve permitir listar todos os CIDs adicionados no atendimento no histórico de atendimentos
 - RF004 - Classificar os CIDs em primário e secundário
Deve permitir classificar os CIDs em primário e secundário
 - RF005 - Classificar em suspeitos
Deve permitir classificar cada CID se é suspeito



Modelagem de Software



Modelagem de Software

- A modelagem é uma parte central de todas as atividades que levam à implementação de um bom software.
- Um modelo é uma simplificação da realidade.
Construímos modelos para compreender melhor o sistema que estamos desenvolvendo.



“Se você realmente quiser construir software equivalentes a uma casa ou a um prédio, o problema não se restringirá a uma questão de escrever uma grande quantidade de software – de fato, o segredo estará em criar o código correto e pensar em como será possível elaborar menos software. Isso faz com que o desenvolvimento de software de qualidade se torne uma questão de arquitetura, processo e ferramenta”

(UML – Guia do Usuário)

Modelagem de software

- 1. Os modelos ajudam a visualizar;
- 2. Permitem especificar a estrutura ou o comportamento de um sistema;
- 3. Proporcionam um guia para a construção do sistema;
- 4. Documentam as decisões tomadas.



Ferramenta Case

Ferramenta Case (Computer Aided Software Engineering)

- Ferramentas que apoia o desenvolvimento de software de forma visual
- Podem ser:
 - Ferramentas de diagramas
 - Demonstra o fluxo da informação. Fluxograma.
 - Ferramentas de modelagem de processos
 - Representam atividades ou tarefas. Exemplo BPMS
 - Ferramentas de gerenciamento de projetos
 - Ferramentas que auxiliam no controle de projetos. Exemplo Creative Pro Office
 - Ferramentas de Documentação
 - Auxiliam em manter a documentação do software. Exemplo DrExplain

Ferramenta Case (Computer Aided Software Engineering)

- Podem ser:
 - Ferramentas de Análise
 - Auxiliam a entender os requisitos. Exemplo CaseComplete.
 - Ferramentas de Design
 - Criação de componentes visuais.
 - Ferramentas de Gerenciamento e Configuração
 - Gerenciamento de versão. Exemplo Git.
 - Ferramentas de Controle de Mudança
 - Gerenciam as mudanças automatizadas do software.
 - Ferramentas de Programação
 - Auxiliam o desenvolvimento, algumas possuem módulos de simulação. Exemplo Eclipse.

Ferramenta Case (Computer Aided Software Engineering)

Podem ser:

Ferramentas de prototipagem

- Auxiliam na criação de produtos independentes de hardware ou do design.
Exemplo Mockup Builder

Ferramentas de desenvolvimento web

- Cria-se páginas com formulários, textos, scripts, etc e geram o seu código.
Exemplo Brackets

Ferramentas de garantia de qualidade

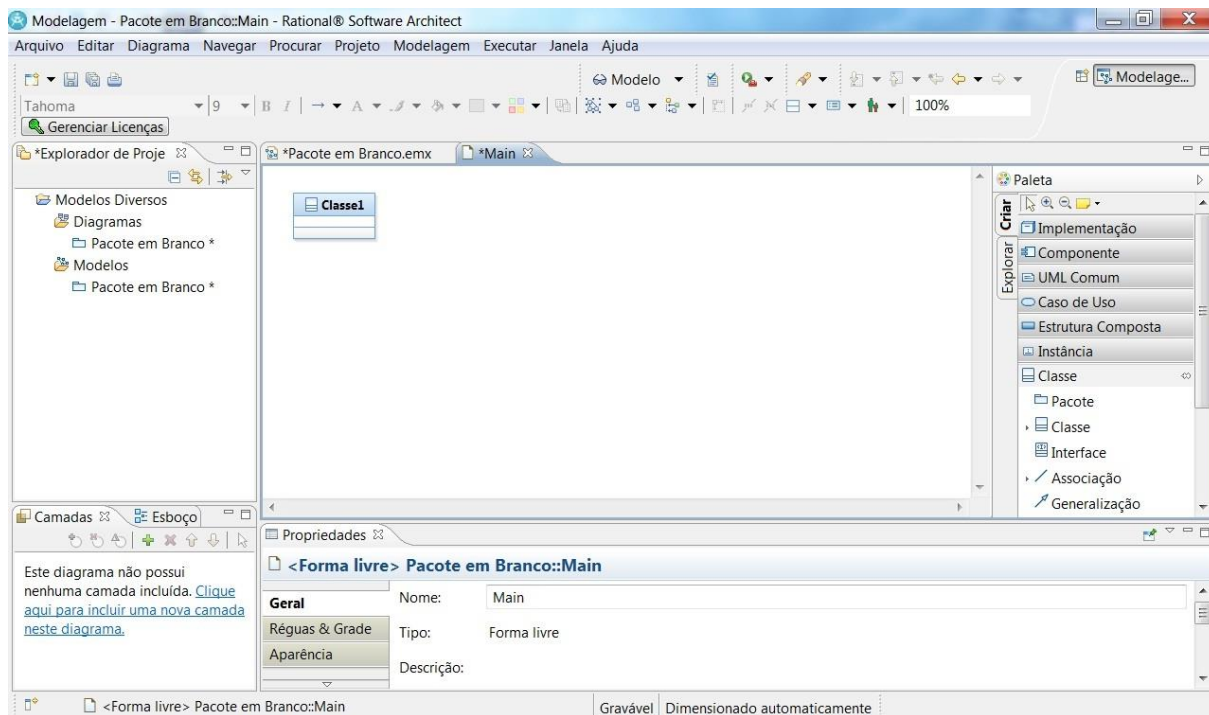
- Possibilitam a criação de testes. Exemplo JMeter

Ferramentas de Manutenção

- Auxiliam a manutenção do software, exemplo HP Quality Center



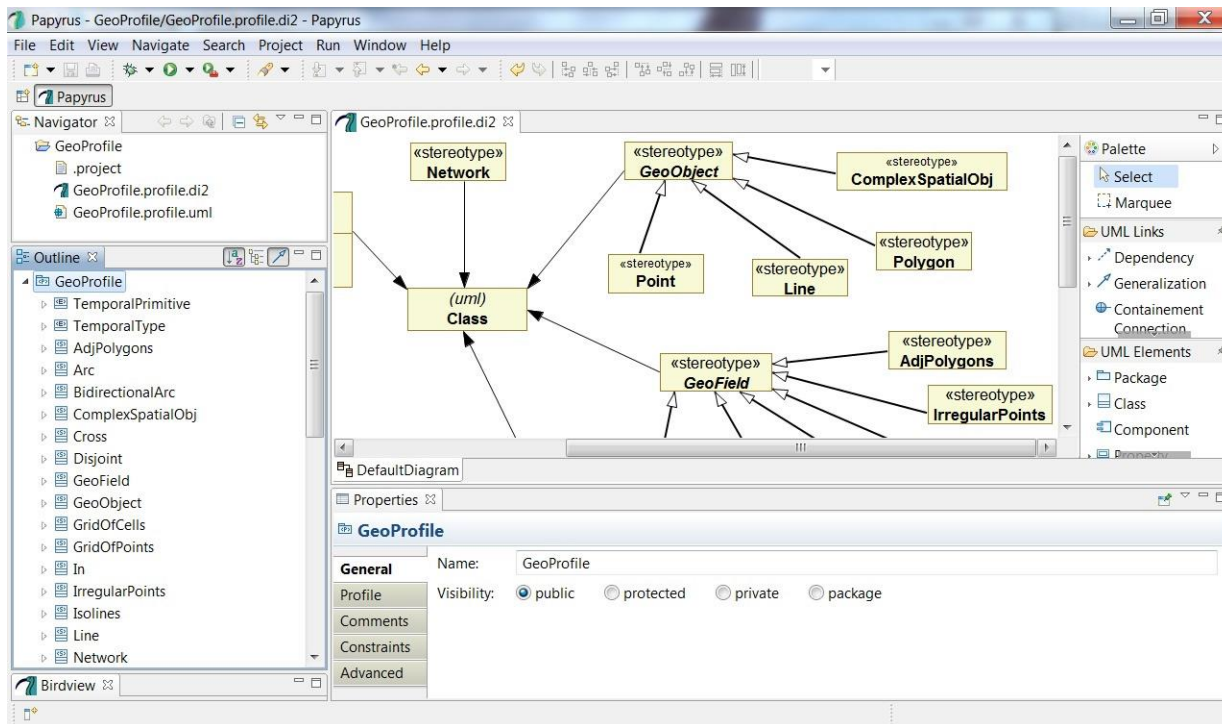
IMB Rational® Software Modeler



- O Rational® Software Modeler (RSM) é uma ferramenta CASE comercial que permite criar diagramas e perfis UML;
- Esta ferramenta é uma ótima alternativa para a especificação, oferecendo recursos como o suporte à linguagem OCL para definição de constraints, inclusão de ícones nos estereótipos e possibilidade de importação e exportação em vários formatos.
 - Object Constraint Language linguagem declarativa para descrever as regras que se aplicam aos modelos UML.
- Outra vantagem desta ferramenta é ser multilinguagem.



Papyrus UML2 Modeler

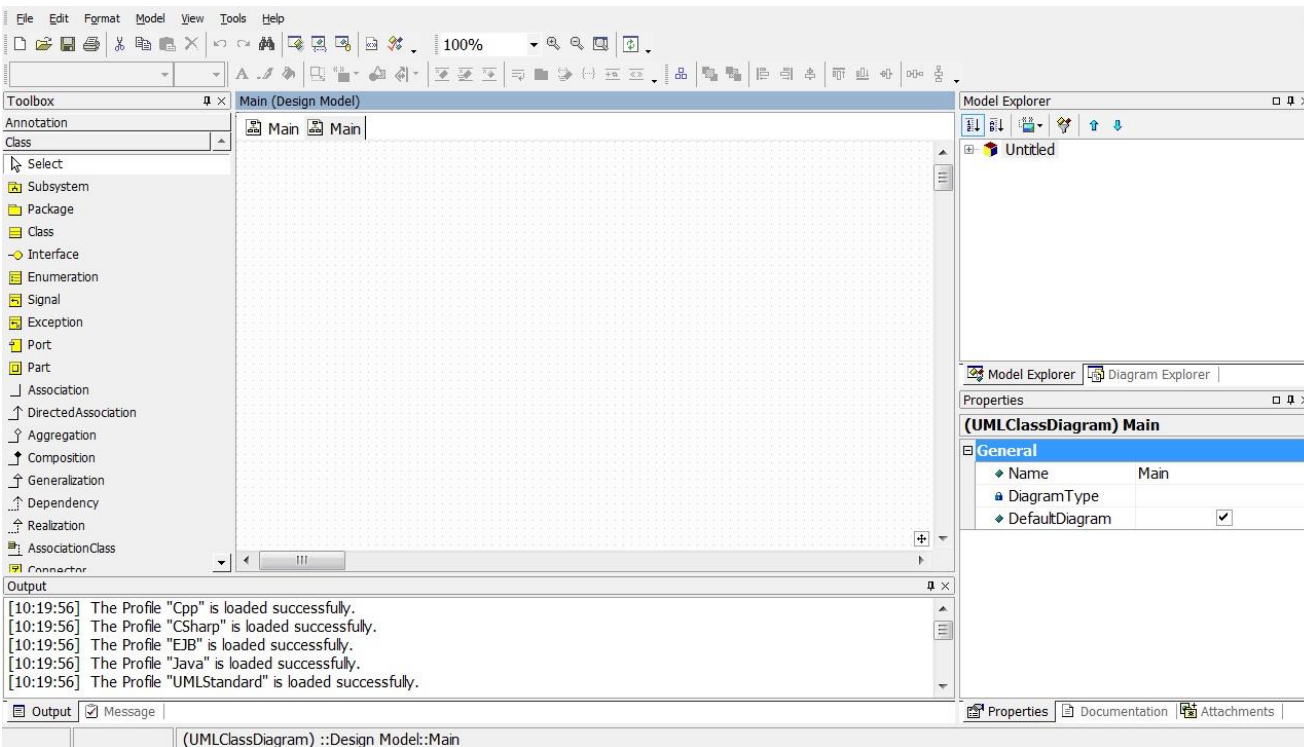


- A ferramenta de código aberto (opensource).
- Baseada no ambiente Eclipse, e está sob a licença EPL (Eclipse Public License).
- A ferramenta oferece suporte à linguagem OCL para definição de constraints, sendo as mesmas utilizadas para validar o esquema Zconceitual gerado.
- Não há opção para importação/exportação de modelos usando o formato XMI



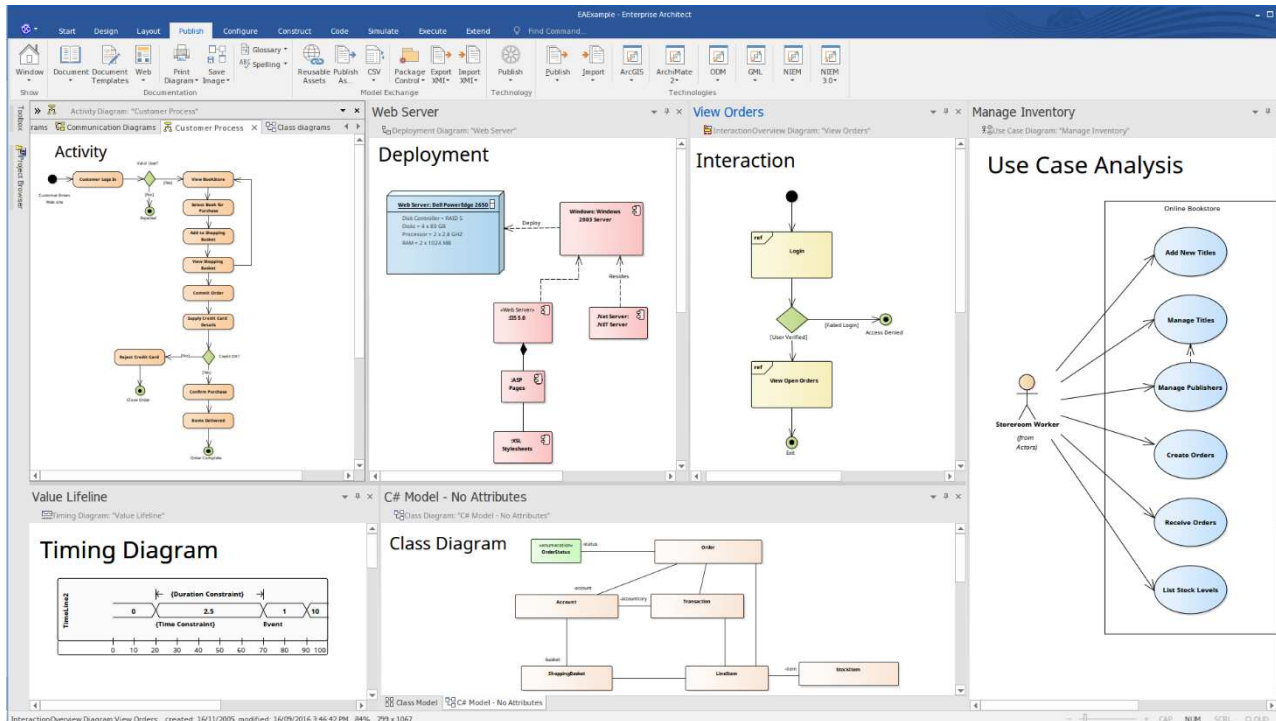
StarUML

- É uma ferramenta CASE de código aberto (opensource) e está sob a licença GPL (General Public License).
- Permite a modelagem de sistemas utilizando os diagramas da UML e também à MDA, com definições de transformações para algumas plataformas específicas.
 - Model Driven Architecture
- É permitida também a importação/exportação de modelos utilizando o formato XMI.
- A ferramenta também não dá suporte à definição de constraints na linguagem OCL.
 - Apesar disso, há opção para usar notação gráfica para os estereótipos.
 - Para usar esse recurso, é necessário declarar no código XML o ícone a ser utilizado.

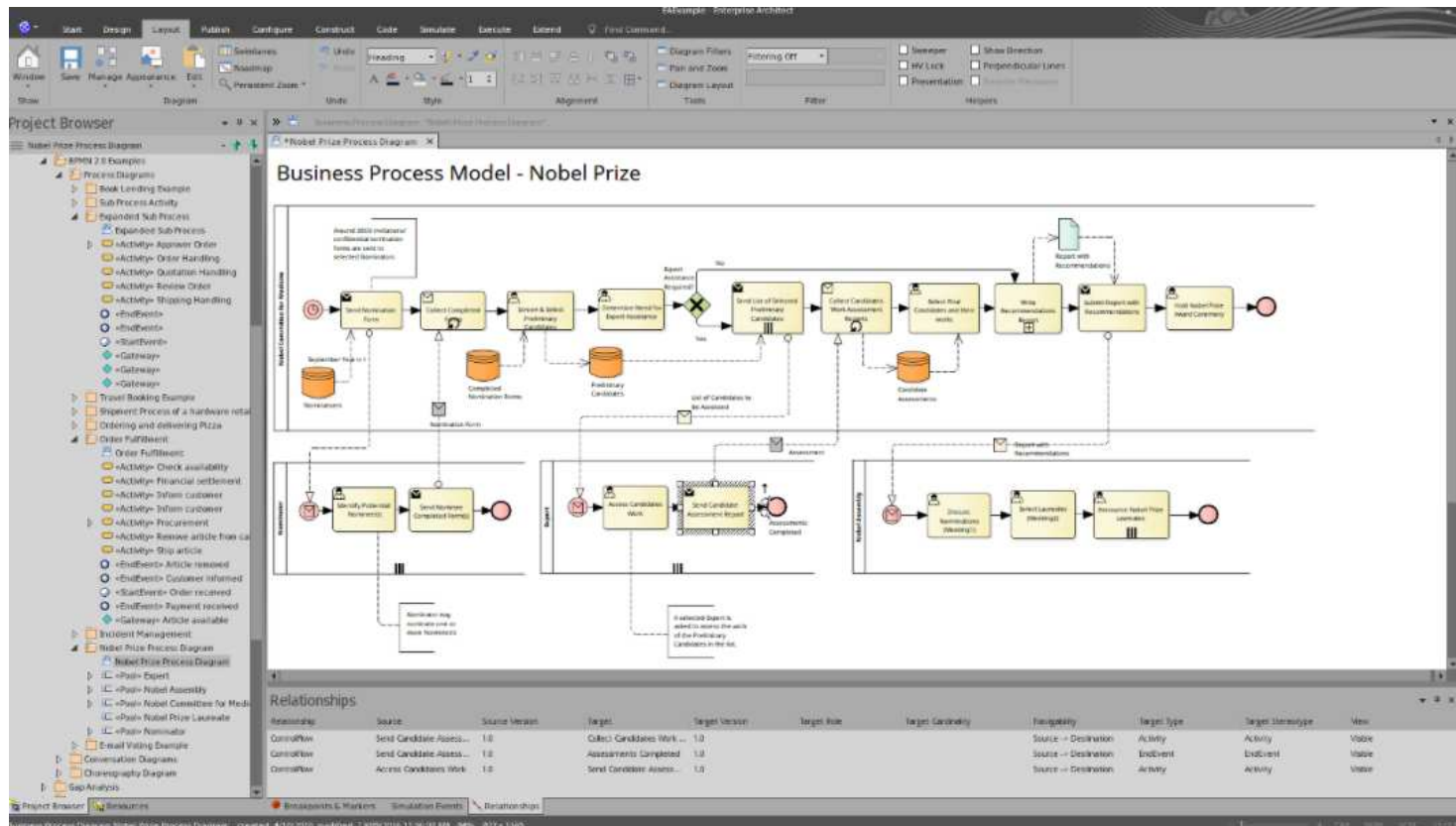


Enterprise Architect

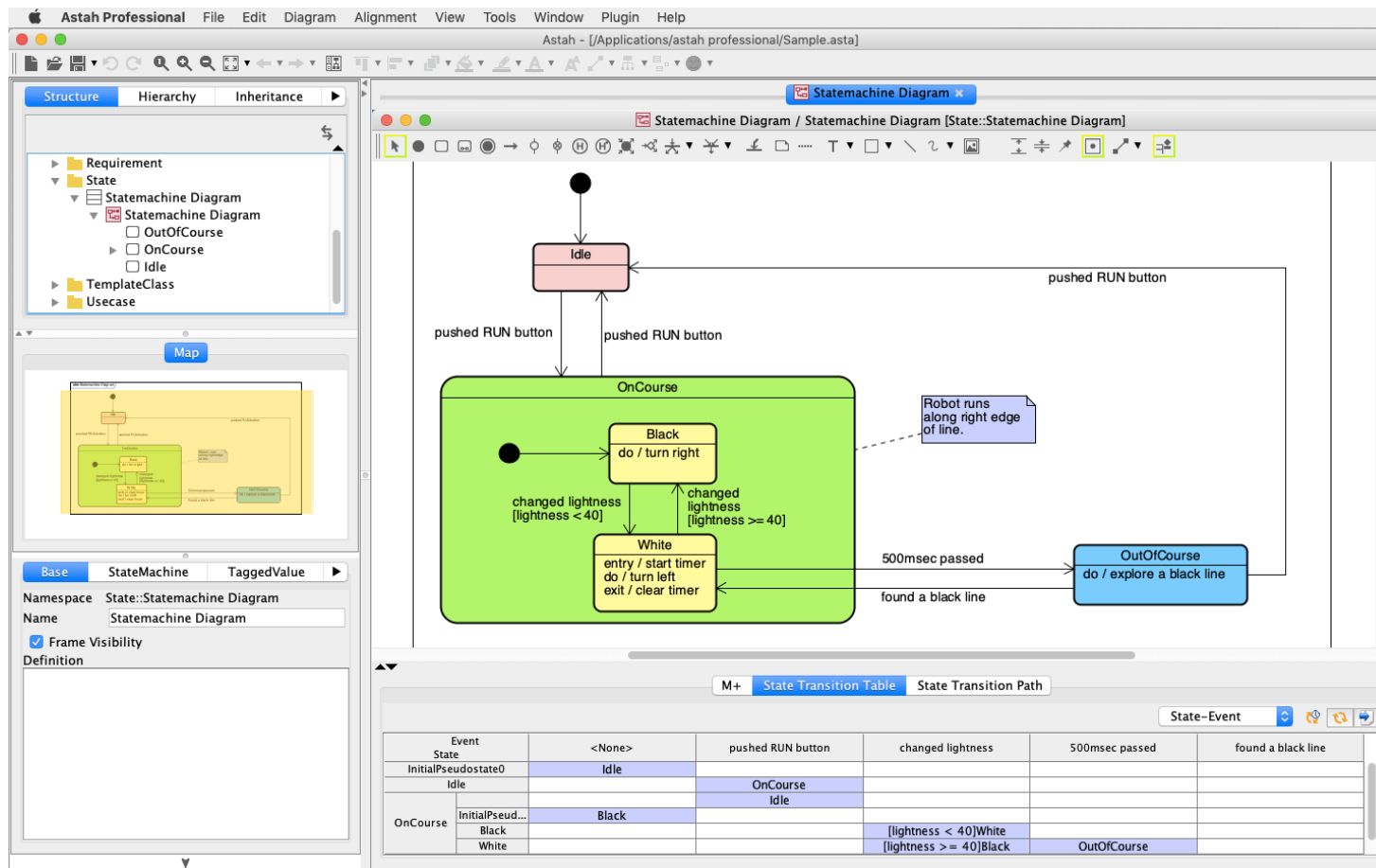
- Ferramenta CASE comercial que nos permite criar diagramas;
- Licenciada pela Sparx Systems
- Permitir a criação de diagramas da UML de forma visual oferece recursos como suporte a OCL para definição de constraints e importação e exportação em XMI (XML Metadata Interchange).
- Apesar de não permitir a inclusão e exclusão de estereótipos gráficos, para melhor visualização e entendimento do diagrama, a vantagem desta ferramenta é a possibilidade de transformações entre os diagramas da arquitetura MDA.
- Permite também a incorporação de outras ferramentas cases como modelagem de processos.



Enterprise Architect



Astah



UML



Os objetivos da UML



- Modelar sistemas (não apenas de software) usando os conceitos da orientação a objetos;
- Estabelecer uma união fazendo com que métodos conceituais sejam também executáveis;
- Criar uma linguagem de modelagem usável tanto pelo homem quanto pela máquina.



Partes que compõem a UML

- Visões

- Mostram diferentes aspectos do sistema que está sendo modelado.
- A visão é uma abstração consistindo em uma série de diagramas.
- Definindo um número de visões, cada uma mostrará aspectos particulares do sistema, dando enfoque a ângulos e níveis de abstrações diferentes e uma figura completa do sistema poderá ser construída.

Visão Lógica
(Logical View)

Visão de Processo
(Process View)

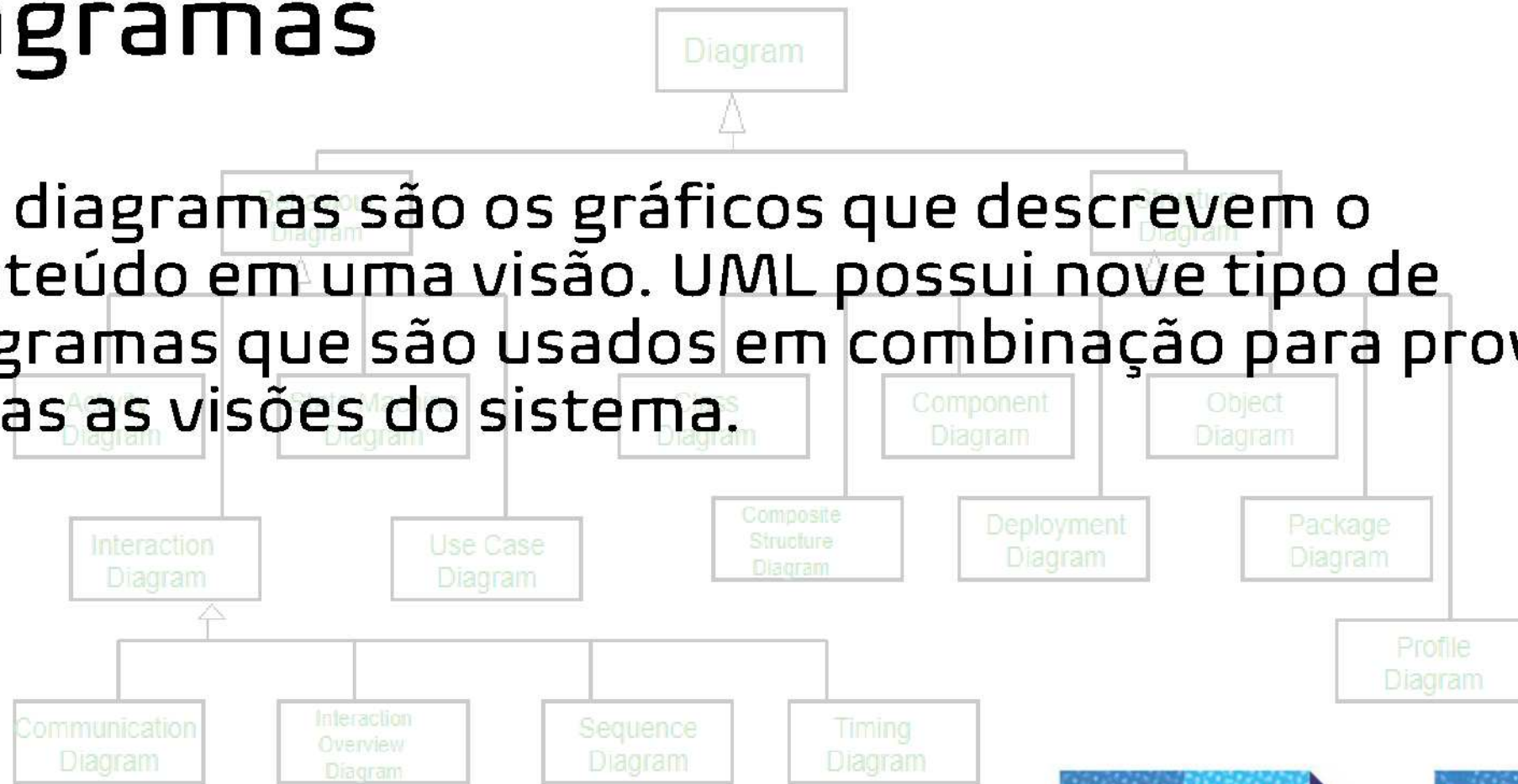
Visão de caso de uso
(Use Case View)

Visão Física
(Physical View)

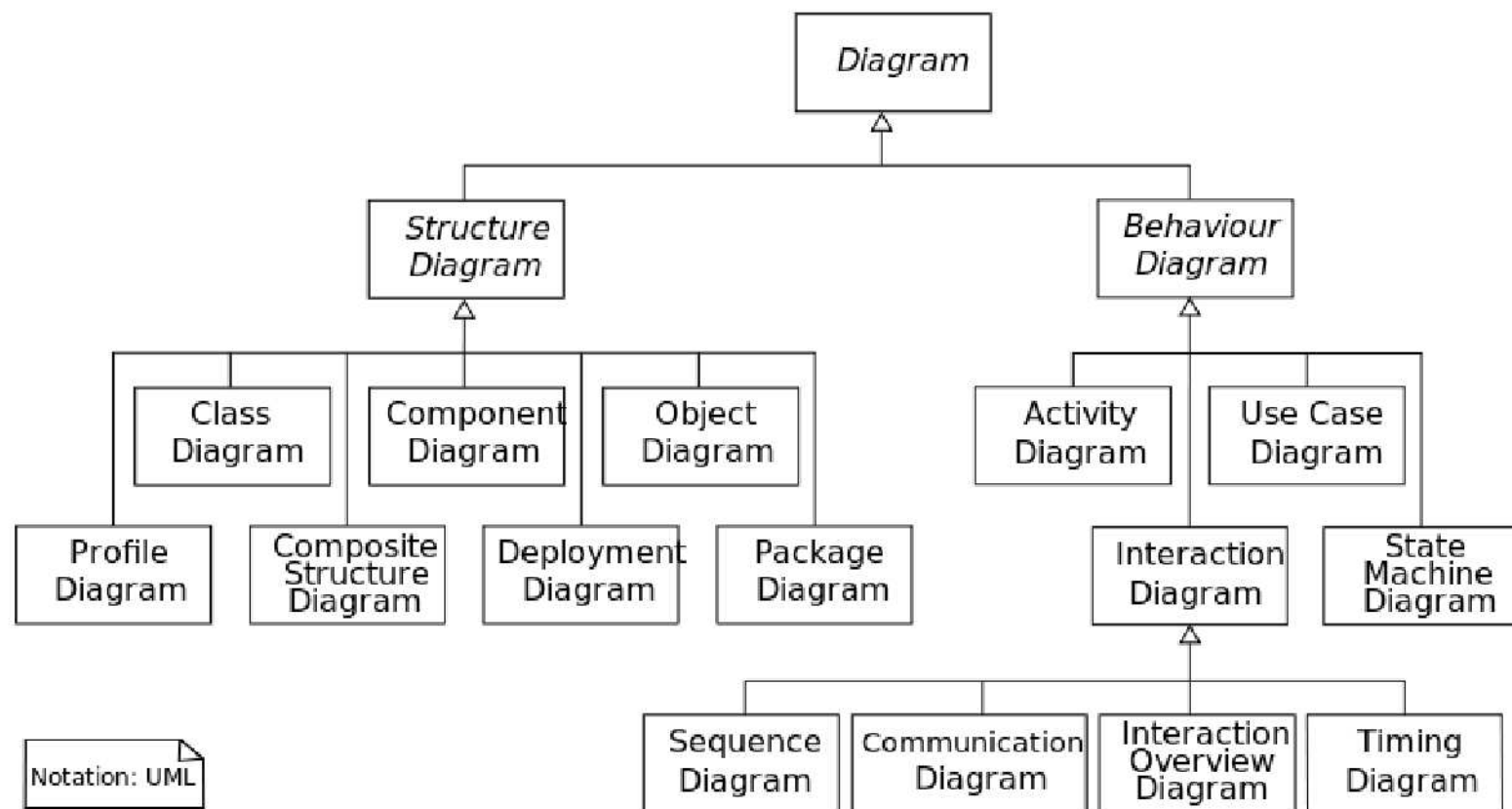
Visão de Desenvolvimento
(Development View)

Diagramas

- Os diagramas são os gráficos que descrevem o conteúdo em uma visão. UML possui nove tipos de diagramas que são usados em combinação para fornecer todas as visões do sistema.



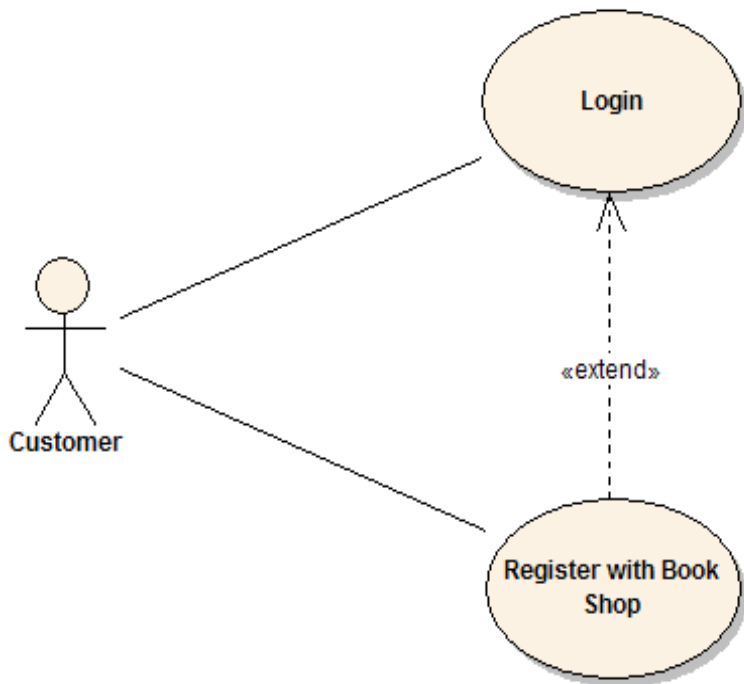
UML



Fonte: UML diagrams overview. Disponível em:
<http://en.wikipedia.org/wiki/Unified_Modeling_Language>

Exemplos de diagramas

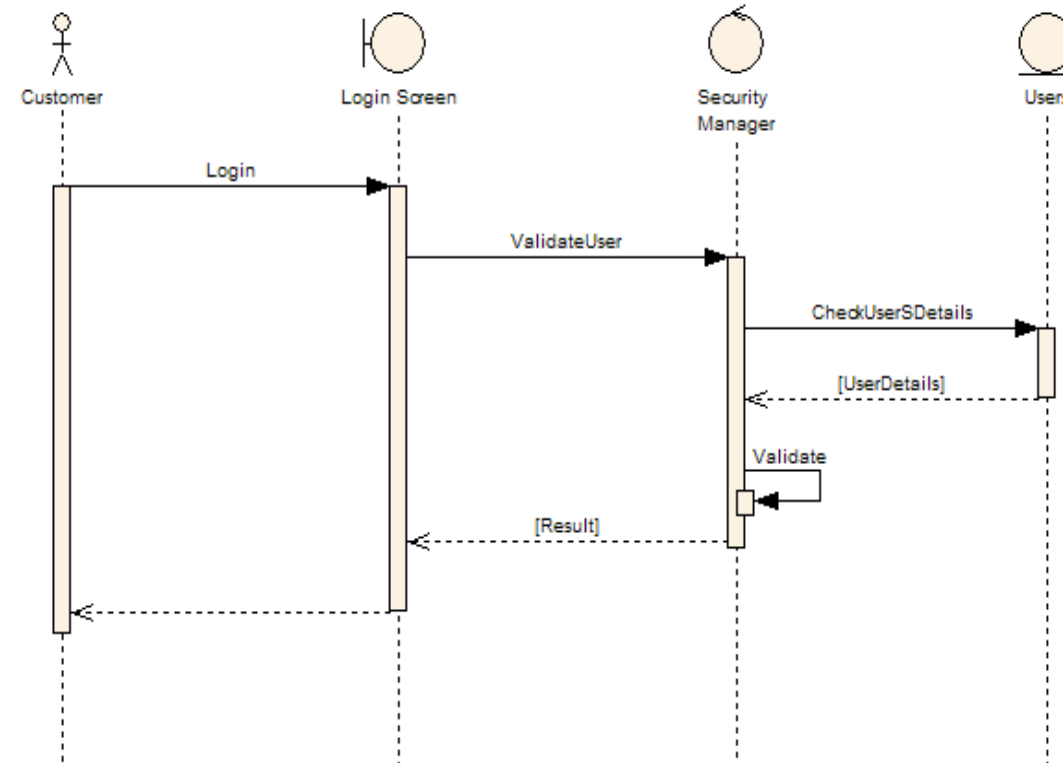
- Diagrama de Use Case



Fonte: Sparx Systems.

<<http://www.sparxsystems.com/uml-tutorial.html>>

- Diagrama de Sequência



Em resumo



- É o padrão para a modelagem Orientada a Objetos.
- Pode ser usada para especificação, construção, visualização e documentação de sistemas de software.
- Pode ser usado durante todo o ciclo de vida de um software.
- Pode ser usado com diferentes tecnologias de implementação.
- Oferece uma notação gráfica baseada em vários diagramas que permitem a modelagem visual de programas orientados a objeto independente de linguagem de programação.



Aonde você quer chegar?
Vai com a

