

Introdução v2:10/1



Prof. Luís Fernando Garcia
LUIS@GARCIA.PRO.BR

Introdução

UML ...

- **Não é:**
 - Não é linguagem de programação
 - Não é processo de desenvolvimento
 - Não é dependente (de linguagem, processo)

Introdução

UML ... É:

- Linguagem Visual,
- Baseada no paradigma da orientação a objetos,
- **Padrão** para:
- Modelagem - Auxiliar os engenheiros de software a definirem as características do software:
 - Requisitos
 - Comportamento
 - Estrutura lógica
 - Dinâmica dos processos
 - Necessidades físicas

Introdução

Modelo/Modelagem

- Imitação de algo ...
- Representação em pequena escala ...
- Molde
- Exemplo ou norma
- Modelo de exemplo prático
- Réplica de algo
- Estilo ou design de algo
- Algo sobre estrutura/comportamento de um sistema

Introdução

Modelo/Modelagem

- Representação simplificada e abstrata de algo para observação, estudo e análise
- Modelo baseado em uma descrição formal de algo

- Abstração de algo que existe na realidade
- Simplificação
- Permitindo análise/reprodução
- Permitindo detalhamento

Introdução

Modelo/Modelagem

- Desenho
- Planta
- Maquete
- Diagrama/esquema
- Mapa

- FORMAIS e/ou INFORMAIS ...
- Depende da necessidade/aplicação

Introdução

Por que Modelar?

- Enfoques ARTESANAL (“de cabeça”) x MODELAGEM
- Desenvolver software é MUITO **complexo**:
- Levantamento de requisitos
- Estimativas
- Projetos
- Prazos
- Custos
- Documentação
- Manutenção
- Reusabilidade

Introdução

Modelagem

- “Modelagem é uma parte central de todas as atividades que levam a implantação de um bom software...”
(Rumbaugh)
- Um modelo é uma abstração (simplificação) da realidade.
- Modelos são construídos para a melhor compreensão do sistema sendo desenvolvido.
 - Os modelos ajudam a visualizar o sistema
 - Os modelos permitem especificar a estrutura e/ou comportamento de um sistema
 - Os modelos proporcionam um guia para a construção do sistema
 - Os modelos documentam as decisões.

Introdução

Definição de UML ...

- “Uma linguagem visual de modelagem unificada, expressiva, poderosa e não-proprietária para tratamento em escala de aplicações de missão crítica, tempo real e cliente/servidor...”
- “Linguagem padrão para especificar, visualizar, documentar e construir sistemas ... Pode ser utilizada ao longo de todo processo de desenvolvimento e com diferentes tecnologias de implementação ...”

UML

Necessidades ...

- O grande problema do desenvolvimento de novos sistemas utilizando a orientação a objetos nas fases de análise de requisitos, análise de sistemas e design é que não existia uma notação padronizada e realmente eficaz que abranja qualquer tipo de aplicação que se deseje ...
- Cada simbologia existente possuía seus próprios conceitos, gráficos e terminologias, resultando numa grande confusão...
- Existiam mais de 50 modelos diferentes ...

Introdução

Origem Métodos Orientados a Objetos

- Booch
- OMT
- OOSE
- Shalaer/Mellor
- Coad/Yourdon
- Martin/Odell
- Wirfs-Brock
- Embley/Kurtz

UML

Origem ...

- Booch [BOOCH]
- Rumbaugh [OMT]
- Jacobson [OOSE]
- UML é a junção do que havia de melhor nestas três metodologias adicionado novos conceitos e visões da linguagem.
- Versão 1.0 → 1997 → OMG
- Versão 2.0 → 2005

Introdução

Orientação a Objetos ...

- "Uma nova maneira de pensar os problemas utilizando modelos organizados a partir de conceitos de mundo real ..." (Rumbaugh)
- **Objeto** → Combina estrutura e comportamento em uma única entidade
- Baixa dependência externa ...
- Alta coesão interna ...
- Reusabilidade ...

Introdução

OO – Conceitos Básicos

- **Objeto**
 - Uma ocorrência específica (instância) de uma classe ...
 - Possui tudo o que é necessário para conhecer a si próprio ...
- **Mensagem**
 - Comunicação de ciclo completo enviada a um objeto requisitando um serviço/operação.
- **Classe**
 - Coleção de objetos que podem ser descritos com os mesmos atributos e operações.
 - Uma instância herda as características de classe a que pertence.

Introdução

OO – Conceitos Básicos

- **Polimorfismo**
 - Capacidade de vários objetos de uma classe responderem de modos diferenciados/especializados a mensagens
 - Mensagens iguais \leftrightarrow objetos diferentes \leftrightarrow comportamentos diferentes.
- **Herança**
 - Capacidade de um novo objeto tomar atributos e operações de um objeto existente ...
 - Pode reimplantar (sobreescrever) operações ...

UML

Fases / Processo ...

- **1) Análise de Requisitos**
- **2) Análise**
- **3) Design (projeto)**

- 4) Programação – não é nosso enfoque ...
- 5) Testes – não é nosso enfoque ...

- “ Executadas concomitantemente de forma que problemas detectados numa certa fase modifiquem e melhorem as fases desenvolvidas anteriormente...”

Introdução

UML para ...

- **Visualizar**
 - Visualização gráfica mas não ambígua
 - Semântica clara e definida dos diagramas
- **Especificar**
 - Especificação precisa, completa e não ambígua

Introdução

UML para ...

- **Construir**
 - UML → código – Eng. Direta
 - Código → UML – Eng. Reversa
- **“Documentar”**
 - UML → documentos

UML

Questão:

UML para

Entender x Documentar

UML



Visão Geral



Prof. Luís Fernando Garcia
LUIS@GARCIA.PRO.BR

UML

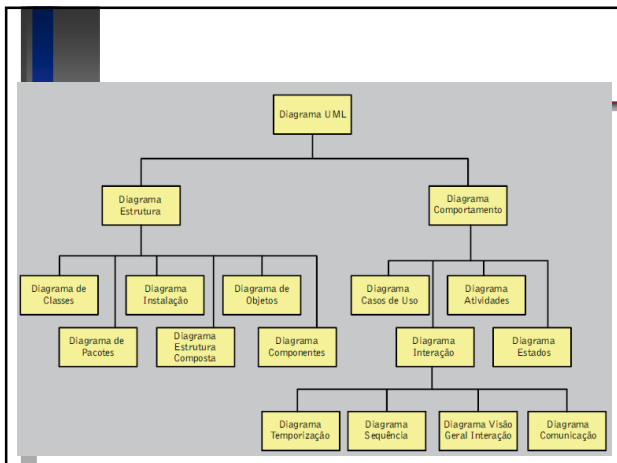
Notações ...

- **Visões**
- **Modelos de Elementos**
- **Mecanismos Gerais**
- **Diagramas**

UML

Notações ...

- **Visões**
 - Mostram diferentes aspectos do sistema que está sendo modelado. A visão não é um gráfico, mas 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. As visões também podem servir de ligação entre a linguagem de modelagem e o método/processo de desenvolvimento escolhido.



UML

Notações ...

- **Visões**

The diagram illustrates five types of UML views, each represented by an oval shape:

- Visão de Componentes
- Visão Lógica
- Visão de Use-case
- Visão de Organização
- Visão de Concorrência

UML

Notações ...

- **Modelos de Elementos**

- Um modelo de elemento é definido com a semântica, a definição formal do elemento com o exato significado do que ele representa sem definições duvidosas ou ambíguas e também define sua representação gráfica que é mostrada nos diagramas da UML.
- Alguns exemplos de modelos de elementos são as **classes**, **objetos**, **estados**, **pacotes** e **componentes**. Os **relacionamentos** também são modelos de elementos, e são usados para conectar outros modelos de elementos entre si.

UML

Notações ...

- **Mecanismos Gerais**

- Ornamentos
- Notas

The diagram shows a class box labeled 'Cliente' connected by a dashed line to a note box. The note contains the following text:

A classe Cliente manterá um vetor com todos os clientes do banco

UML - DIAGRAMAS

- de use case / casos de uso
- de classes
- de objeto
- de pacotes
- de sequencia
- de comunicação
- de máquina de estados
- de atividade
- de visão geral de interação
- de componentes
- de implantação
- de estrutura composta
- De tempo / temporização

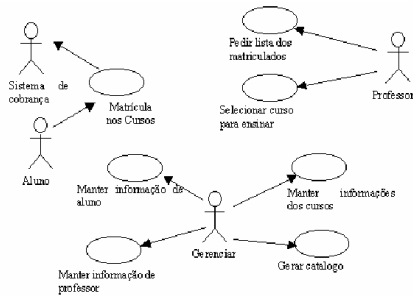
UML

Caso de Uso

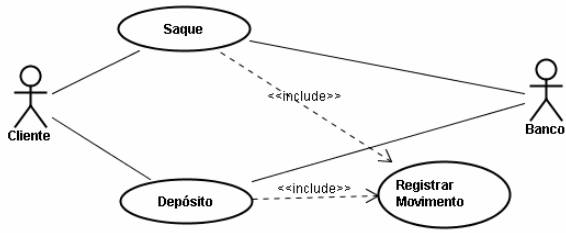
- Diagrama utilizado para se identificar como o sistema se comporta em várias situações que podem ocorrer durante sua operação.
- Técnica usada para descrever e definir os requisitos funcionais de um sistema.
- Termos de **atores (CLASSE)**, **use-cases (CLASSE)** e o **sistema modelado**.



UML



UML

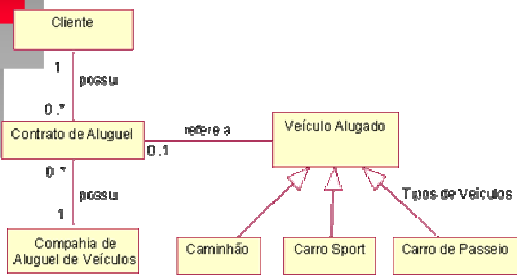


UML

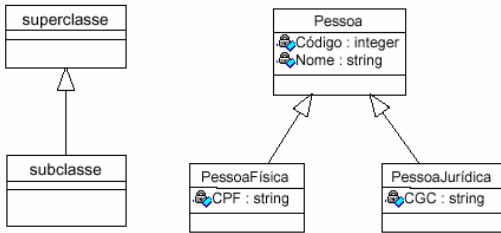
Classes

- Demonstra a estrutura estática das classes de um sistema
- Classes podem se relacionar com outras através de diversas maneiras:
 - associação (conectadas entre si),
 - dependência (uma classe depende ou usa outra classe),
 - especialização (uma classe é uma especialização de outra classe),
 - pacotes (classes agrupadas por características similares)
- Todos estes relacionamentos são mostrados no diagrama de classes juntamente com as suas estruturas internas, que são os atributos e operações.

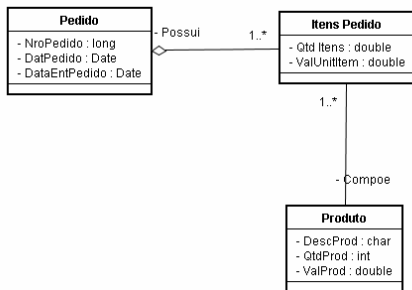
UML



UML



UML

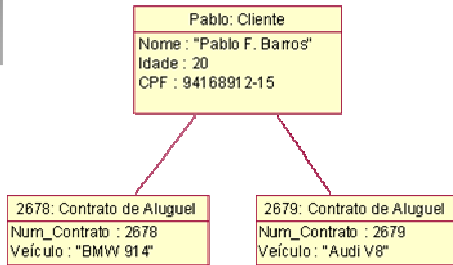


UML

Objeto

- O diagrama de objetos é uma variação do diagrama de classes e utiliza quase a mesma notação. A diferença é que o diagrama de objetos mostra os objetos que foram instanciados das classes. O diagrama de objetos é como se fosse o perfil do sistema em um certo momento de sua execução.
- Diagramas de objetos também são usados como parte dos diagramas de colaboração, onde a colaboração dinâmica entre os objetos do sistema são mostrados.

UML



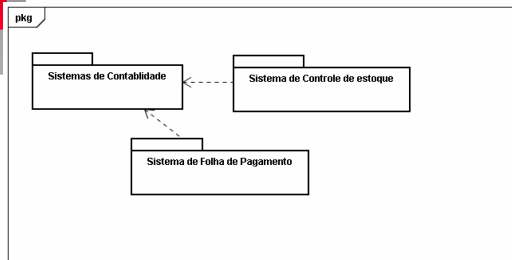
UML

Pacotes

- Diagramas de Pacotes é um diagrama estrutural que tem por objetivo representar os subsistemas ou submódulos do sistema, de modo a determinar as partes que o compõe.
- Independente ou associado a outros diagramas
- EX: Camadas de software
- EX: Camadas de um processo

UML

Diagrama de Pacotes

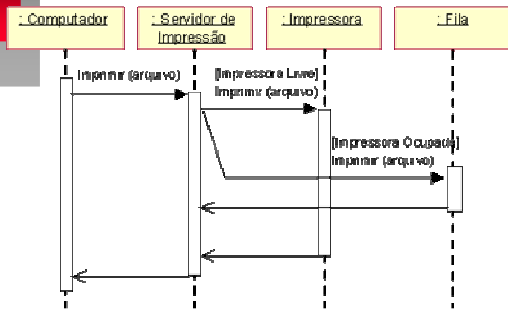


UML

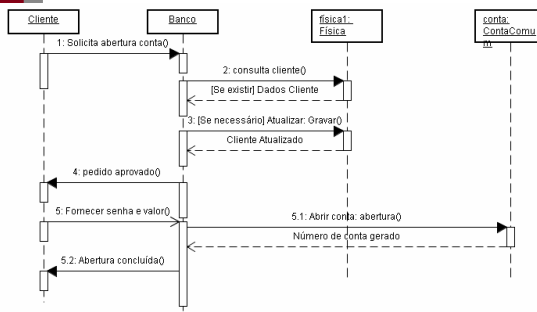
Seqüência

- Um diagrama de seqüência mostra a colaboração dinâmica entre os vários objetos de um sistema. O mais importante aspecto deste diagrama é que a partir dele percebe-se a seqüência de mensagens enviadas entre os objetos.
- Ele mostra a interação entre os objetos.
- Diagramas de seqüência possuem dois eixos: o eixo vertical, que mostra o tempo e o eixo horizontal, que mostra os objetos envolvidos na seqüência de uma certa atividade. Eles também mostram as interações para um cenário específico de uma certa atividade do sistema.

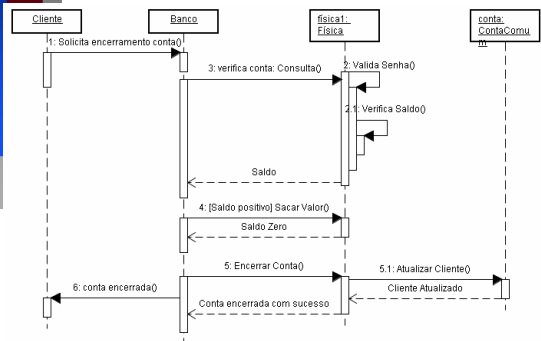
UML



UML



UML

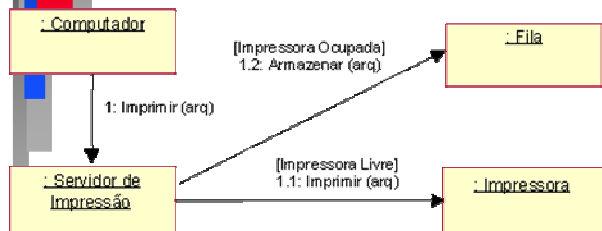


UML

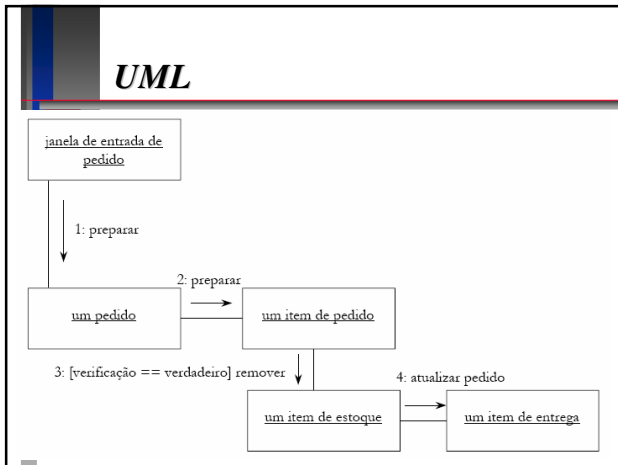
Comunicação (ex-colaboração)

- Mostra de maneira semelhante ao diagrama de seqüência, a colaboração dinâmica entre os objetos. Normalmente pode-se escolher entre utilizar o diagrama de colaboração ou o diagrama de seqüência.
- Se a ênfase do diagrama for o decorrer do tempo, é melhor escolher o diagrama de seqüência, mas se a ênfase for o contexto do sistema, é melhor dar prioridade ao diagrama de comunicação.

UML



UML

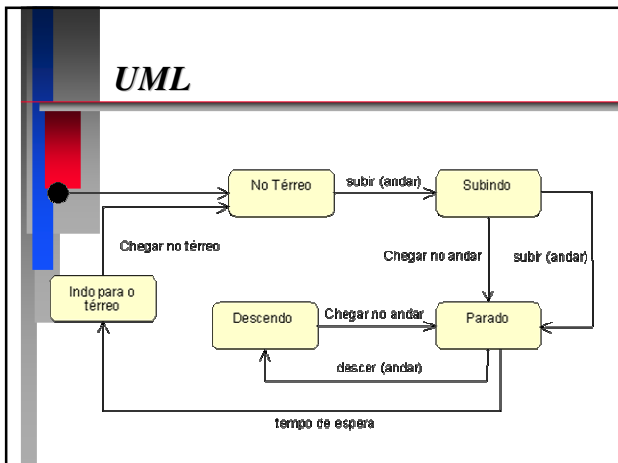


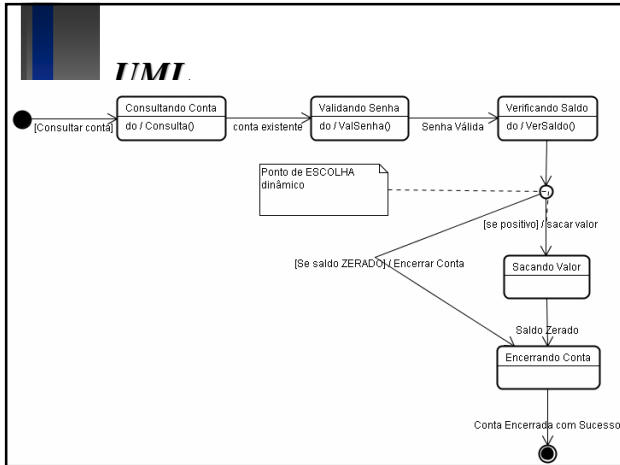
UML

Máquina de Estados

- O diagrama de estado é tipicamente um complemento para a descrição das classes.
- Este diagrama mostra todos os estados possíveis que objetos de uma certa classe podem se encontrar e mostra também quais são os eventos do sistema que provocam tais mudanças.

UML





UML

Atividade

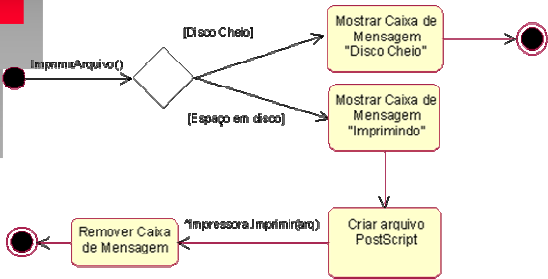
- Diagramas de atividade capturam ações e seus resultados. Eles focam o trabalho executado na implementação de uma operação (método), e suas atividades numa instância de um objeto.
- O diagrama de atividade é uma variação do diagrama de estado e possui um propósito um pouco diferente do diagrama de estado, que é o de capturar ações (trabalho e atividades que serão executados) e seus resultados em termos das mudanças de estados dos objetos.

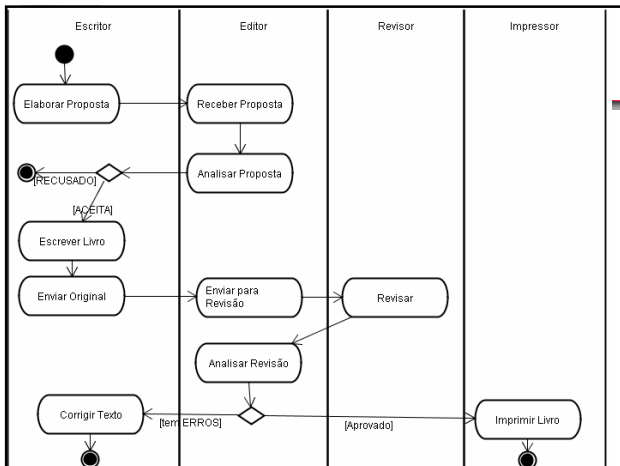
UML

Diagramas de atividade - Propósitos

- Para capturar os trabalhos que serão executados quando uma operação é disparada (ações).
- Para capturar o trabalho interno em um objeto.
- Para mostrar como um grupo de ações relacionadas podem ser executadas, e como elas vão afetar os objetos em torno delas.
- Para mostrar como uma instância pode ser executada em termos de ações e objetos.
- Para mostrar como um negócio funciona em termos de trabalhadores (atores), fluxos de trabalho, organização, e objetos (fatores físicos e intelectuais usados no negócio).

UML

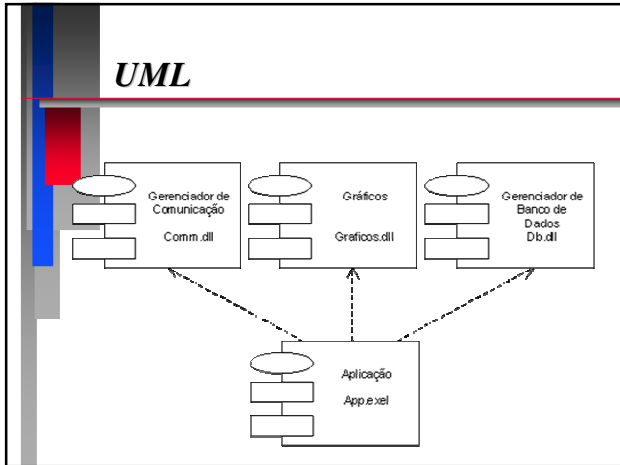




UML

Componentes

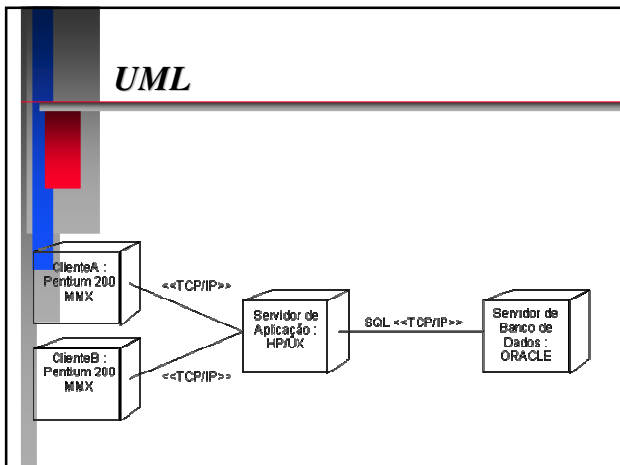
- O diagrama de componente e o de execução são diagramas que mostram o sistema por um lado funcional, expondo as relações entre seus componentes e a organização de seus módulos durante sua execução.
- O diagrama de componente descreve os componentes de software e suas dependências entre si, representando a estrutura do código gerado. Os componentes são a implementação na arquitetura física dos conceitos e da funcionalidade definidos na arquitetura lógica (classes, objetos e seus relacionamentos). Eles são tipicamente os arquivos implementados no ambiente de desenvolvimento.



UML

Implantação

- O diagrama de execução mostra a arquitetura física do hardware e do software no sistema. Pode mostrar os atuais computadores e periféricos, juntamente com as conexões que eles estabelecem entre si e pode mostrar também os tipos de conexões entre esses computadores e periféricos.



Exemplo – Estudo de Caso

Descrição

- uma modelagem em UML para criarmos um sistema de manutenção e controle de contas correntes e aplicações financeiras de um banco.

Exemplo – Estudo de Caso

Observações

- O sistema suportará um cadastro de clientes, onde cada cliente cadastrado poderá ter várias contas correntes, vários dependentes ligados a ele, e várias contas de poupança.
- Cada dependente poderá possuir várias contas de poupança, mas não poderão ter uma conta corrente própria.
- Entendemos poupança como uma conta que possui um valor, um prazo de aplicação a uma taxa de juros (definida no vencimento da poupança).
- Entendemos Aplicações Pré-fixadas como uma aplicação de um valor, em um prazo pré-determinado a uma taxa de juros previamente definida.
- Tanto a conta corrente quanto a poupança deverão manter um histórico de todas as movimentações de crédito, débito, transferências e aplicações de pré-fixados (pré-fixados apenas para conta corrente).
- Uma conta corrente poderá ter várias aplicações pré-fixadas ligadas a ela.

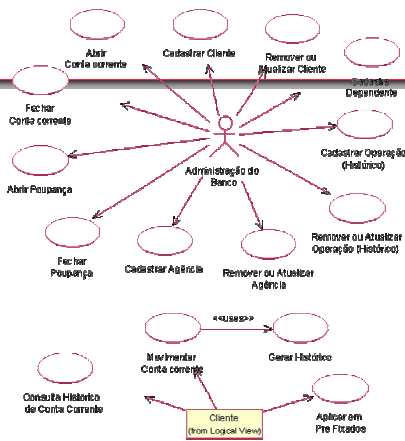
Exemplo – Estudo de Caso

Análise de Requisitos

- Cadastrar novo cliente
- Excluir ou editar cliente
- Cadastrar dependente
- Excluir ou editar dependente
- Abrir conta corrente
- Fechar conta corrente
- Abrir poupança
- Fechar poupança
- Movimentar conta corrente
- Aplicar em pré-fixados
- Consultar histórico de conta corrente ou poupança
- Cadastrar Agência
- Excluir ou Editar Agência

Exemplo – Estudo de Caso

Diagrama de Use-Case



Exemplo – Estudo de Caso

Diagrama de Classes
