Macoratti.net UML - Unified Modeling Language e Visual Modeler (Visual Basic 6)

Se você é daqueles que quando vai desenvolver um sistema senta na frente do computador e já começa a codificar , eu lamento informá-lo , mas você já começou de forma errada e provavelmente vai terminar assim. Esta atitude somente se justifica para pequenos sistemas pessoais.

Um sistema profissional tem o seu início na modelagem dos dados onde você pode projetar , documentar e visualizar o sistema, independente da tecnologia que será usada para sua implementação quer seja Visual Basic , C , Java , etc...

Vamos dar uma pequena introdução aos conceitos da UML e abordar o **Visual Modeler ;** uma ferramenta que o Visual Basic 6 oferece para modelar objetos ***(somente a versão Enterprise).***

Introdução

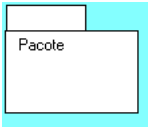
A **UML - Unified Modeling Language** - é um modelo de linguagem para modelagem de sistemas orientados a objetos (principalmente) . Com ela podemos fazer uma modelagem visual de maneira que os relacionamentos entre os componentes do sistema sejam melhor visualizados e compreendidos e documentados.

Podemos dizer também que a **UML** é uma linguagem para especificar , construir , visualizar e documentar um sistema de software que surgiu com a fusão das metodologias já anteriormente usadas e tem como objetivo

1. Modelar sistemas usando os conceitos da orientação a objetos
2. Estabelecer um elo explicito entre os artefatos conceituais e os executáveis
3. Tratar questões de representar sistemas complexos de missão crítica
4. Criar um linguagem de modelagem que possa ser usada por homens e por máquinas

A seguir algumas das notações mais usadas pela linguagem UML:

Pacote - Organiza as classes de objetos em grupos. É representado como um retângulo com uma aba no canto superior esquerdo.



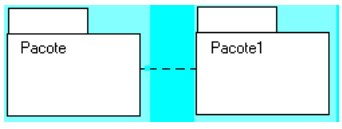
OBS:

- As classes são definidas dentro do pacote
- Pacotes diferentes podem ter classes com nomes iguais
- Para referenciar uma classe de um pacote usamos a sintaxe:
  - **NomedoPacote :: NomeDaClasse**

Visibilidade do Pacote :

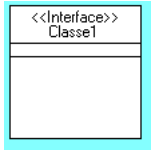
1. **Privado -** Só o pacote que define determinadas classes tem acesso a elas.
2. **Protegido** - Só os pacotes gerados a partir do pacote podem acessar suas classes.
3. **Público** - O conteúdo do pacote pode ser acessado por outros elementos
4. I**mplementação** - Idêntico a definição do pacote privado com algumas restrições para o uso dos elementos do pacote.

Uma dependência entre as classes de dois pacotes leva a dependência entre os pacotes que podemos representar como na figura ao lado.



Estereótipo

É um elemento de modelagem que rotula tipos de classes de objeto sendo que uma Classe de objetos pode ter um ou mais tipos de estereótipos. É descrito entre os sinais << e >> ou como um ícone no canto superior direito.



Os estereótipos mais comuns são :

- Classe Fronteiriça
- Classe de Entidade
- Classe de Controle
- Classe de Exceção
- Classe Utilitária

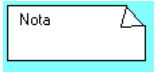
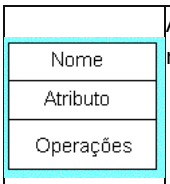
Nota - Comentário usado em um diagrama. É exibida como um retângulo com o canto direito superior dobrado.	

Diagrama de Classe

A UML utiliza uma notação própria para construção de diagramas . Para representar uma Classe de objetos a notação é :



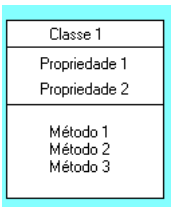
A classe de objeto é representada por um retângulo , subdividido em três áreas :

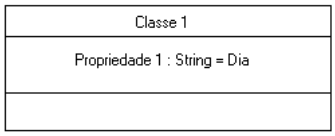
1. Nome da Classe
2. Atributos da classe
3. Operações da classe

O diagrama de classe é um gráfico que mostra a estrutura do sistema : classes , tipos e seus conteúdos e relações.

Para descrever os atributos a sintaxe é :

Visibilidade NomedoAtributo : TipoDeExpressao = ValorInicial {Propriedade}





A visibilidade pode ser:

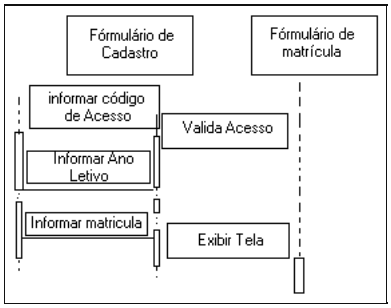
- + Publica
- # protegida
- - privada

Diagrama de sequência

O diagrama de sequência mostra a interação entre os objetos ao longo do tempo e apresenta os objetos que participam da interação e a sequência de mensagens trocadas. A notação usada pela UML para representar o diagrama de sequência é :

1. Os objetos são representados por um retângulo com seus nomes sublinhados
2. As linhas de vida dos Objetos são representadas por linhas verticais tracejadas
3. As interações entre objetos são indicadas por flechas horizontais que são direcionadas da linha vertical que representa o objeto cliente para a linha do objeto fornecedor
4. As flechas horizontais são rotuladas com mensagens
5. A ordem as mensagens no tempo é indicada pela posição vertical , com a primeira mensagem aparecendo no topo.
6. A numeração é opcional e baseada na posição vertical.

Abaixo um pequeno esboço de um diagrama de sequência :



Relacionamento

É a forma como as classes de objetos interagem entre si para formar o comportamento do sistema. Esse relacionamento é apresentado através de um **Diagrama de classes**. Os dois tipos principais de relacionamento sâo:

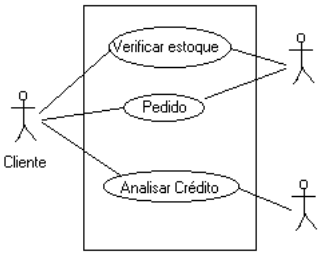
1- **Associação** : Representa uma dependência estrutural entre os objetos é representado por uma linha sólida entre as classes. O fluxo de dados pode ser uni ou bidirecional através da conexão e entre duas classes pode existir mais de uma associação.

2- **Agregação** - Permite mostrar que uma classe é composta por objetos de outra classe. É representada por uma linha sólida com um losango ligado a classe cliente.

Diagrama de Caso de Uso (Use-Case)

Os diagrama de caso de uso exibem a visão externa do sistema e suas interações com o mundo exterior descrevendo seus requerimentos e suas responsabilidades e possuem três elementos :

1. **Ator** - agente que interage com o sistema
2. **Caso de Uso** - o comportamento da classe
3. **Interação** - envio e recebimento de mensagens da comunicação ator - sistema.



Exemplo de diagrama de Caso de uso

Eu não vou me estender mais pois estou consciente da complexidade do assunto , apenas quis dar uma visão básica sobre a UML . Se você quiser se aprofundar sugiro um bom livro sobre o assunto - Utilizando UML e padrões , autor Craig Larman , editora Bookman , ISBN - 85.7307-651-8 - Porto Alegre - RS.

Para concluir podemos dizer que a modelagem visual orientada a objetos agora tem um padrão simples e robusto para especificar e descrever a grande maioria das funções , relacionamentos e técnicas de desenvolvimento orientado a objetos.

Os fabricantes de ferramentas CASE com certeza irão suportar a UML e a fase de codificação será cada vez mais substituída pela geração de código automático pelas ferramentas CASE. Podemos citar dois exemplos destas ferramentas : **ERviw/ERX** da Logik Works **e Rational Rose** da Rational Software Corp.

Usando o Visual Modeler do VB

O Visual Basic apresenta na sua versão Entrerprise uma ferramenta gráfica que podemos usar para modelagem de dados orientada a objetos - **Visual Modeler** - . Esta ferramenta apresenta as seguintes características básicas :

1. Diagrama de classes - A notação para o diagrama fornecido pelo Visual Modeler é um subconjunto da modelagem definida pela UML
2. Geração de código - O Visual Modeler gera automaticamente código Visual Basic e C++ a partir do desenho de modelo criado.
3. Engenharia Reversa - O VM cria tem a habilidade de criar automaticamente ou atualizar o modelo com as mudanças feitas pelo código Visual Basic
4. A combinação da modelagem , geração de código e engenharia reversa.

Usando o Visual Modeler você pode modelar as classes do seu sistema e após isto pode gerar o respectivo código de maneira automática ou pode fazer o contrário : você cria o código de suas classes e gera automaticamente o diagrama de classes.

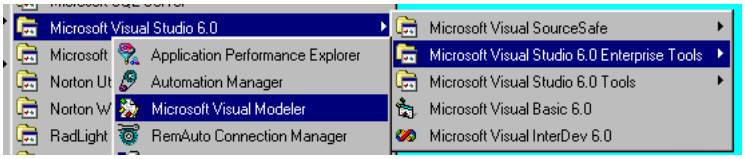
O Visual Modeler apresenta a visão de sistema de três formas diferentes:

1-) **Visão Lógica** - Aqui são descritas as classes e os relacionamentos

2-) **Visão de componentes** - Exibe a estrutura fisica do sistema : as bibliotecas e os executáveis

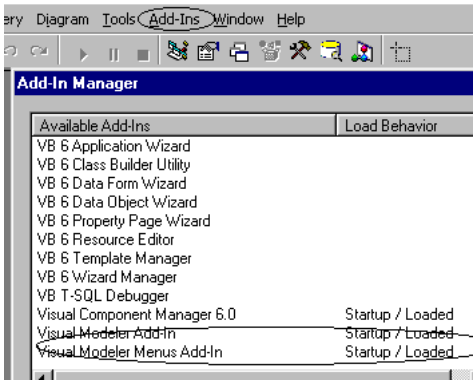
3-) **Visão de distribuição** - Exibe os nós do sistema e suas conexões bem como a alocação dos processo para os nós.

Para iniciar o Visual Modeler você deve selecionar o menu Iniciar->Programas->Microsoft Visual Studio 6.0 -> Microsoft Visual Studio 6.0 Enterprise Tools -> Microsoft Visual Modeler

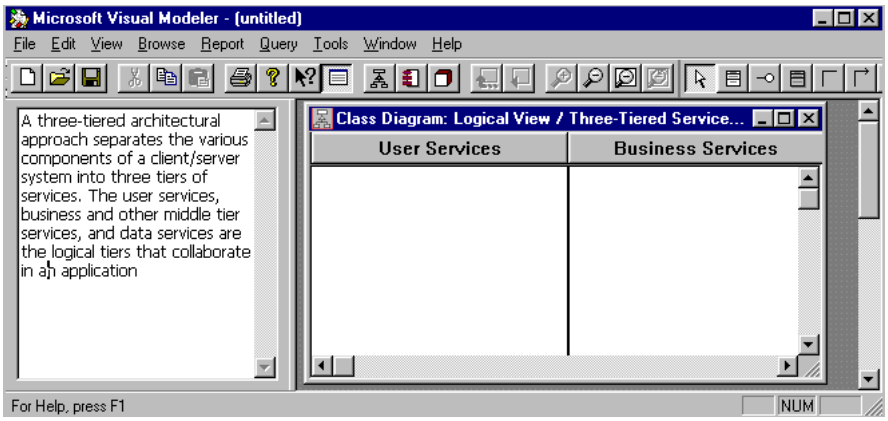


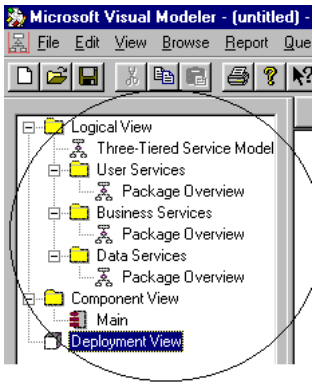
Ou , se preferir , pode carregar o VM como um add-ins do Visual Basic ; Inicie o VB e no menu **Add-Ins** selecione a opção **Add-In Manager... .**

Na janela Add-In marque para carregar (**Startup/Loaded)** os add-ins - Visual Modeler Add-in e Visual Modeler Menus Add-in. Pronto agora no VB estará visível a opção para carregar o VM.



Ao executar o Visual Modeler você deverá ver a janela principal do programa :





A visão padrão que o Visual Modeler oferece é a visão Logica onde temos as classes e seus relacionamentos.

Se você clicar expandir cada item terá o exibido na figura ao lado e verá o nome atual da visão lógica do modelo : **Three Tired Service Model.**

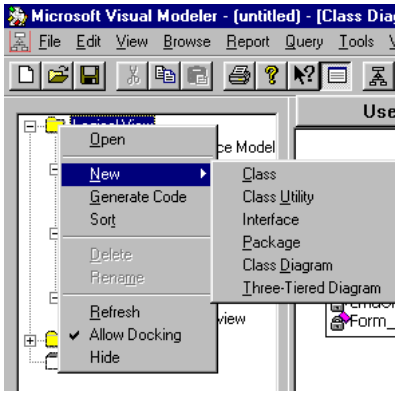
Para modificar clique com o botão direito do mouse e selecione - **Rename -** alterando o nome para o desejado.

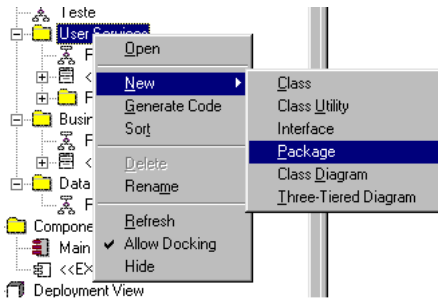
Os três pacotes exibidos correspondem ás três camadas de uma aplicação multicadamada :

- Users Services - Camada de Interface com o usuario
- Business Services - Camada de regra de negócios
- Data Services - Camada de dados

Cada camada já tem um pacote que mostra o conteúdo da camada. O nome de cada pacote é - **Package Overview.**

Você pode criar um novo modelo. Para isto clique com o botão direito do mouse sobre o item - **Logical View** - e selecione a opção - **New** - e a opção **Class Diagram** informando o nome do modelo.





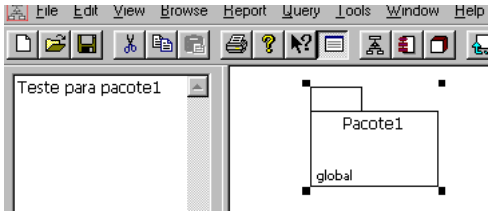
A primeira coisa que você deve fazer para criar o seu diagrama é criar os pacotes relativos a ele. Os pacotes irão conter classes relacionadas. Para fazer isto clique com o botão direito do mouse na camada onde deseja criar o seu pacote e selecione a opção - New - e a subopção **Package** e informe o nome do pacote.*(Eu usei o nome pacote1)*

Veja as figuras ao lado.

Agora que você criou o pacote se clicar sobre o ele duas vezes verá a janela - **Package Specification for Pacote1** que apresenta duas abas :

- **General** - Aqui você define a documentação para o pacote
- **Detail** - Aqui você define se o pacote será Global. (Para isto marque a opção Global)

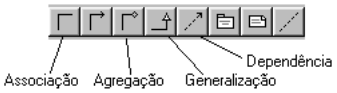
Após definir estes detalhes se você clicar no item - Package Overview da camada relacionada verá o novo pacote criado.



Vamos agora inserir uma classe ; para isto selecione o pacote desejado e clique duas vezes ele . classes irá surgir . Selecione a opção **Tools** do Menu , selecione a seguir **Create** e o item **Class**. o formato de uma cruz , arraste-o para a área do diagrama e clique para formar a classe.

Para incluir uma propriedade ou método na classe clique com o botão direito do mouse sobre a classe e selecione a opção **New Property** ou New Method.

A propriedade será inserida no diagrama dentro da classe e inserida na arvore. Se você quiser fazer qualquer alteração sobre as propriedades ou métodos clique duas vezes sobre ela na árvore e altere os dados na janela que surgirá.



Para criar relacionamentos entre as classes basta selecionar na barra de ferramentas o tipo de relacio arraste o mouse da primeira para a segunda classe relacionadas. Ao lado temos os principais relacion

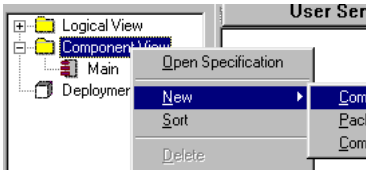
A visão de componentes

Além da visão Lógica o Visual Modeler exibe a visão de componentes onde um diagrama de componentes mostra a organização entre os componentes do sistema : bibliotecas , DLL´s , arquivos EXE . Além disto a visão de componentes mostra o comportamento e dependência entre os componentes. O diagrama de componentes é composto por :

1. Pacotes de componentes - representa os grupos de componentes relacionados
2. Componentes - representa os módulos de programa : DLL , EXE , etc
3. Interfaces - define os métodos visíveis de uma classe ou componente
4. Relacionamento de dependência - Exibe quando um componente usa serviços de outro componente.

Para criar um novo componente você deve clicar com o botão direito do mouse sobre o item - **Component View** - e selecionar **New -> Component Diagram**

Para excluir , renomear um diagrama repita a operação acima e selecione - **Rename ou Delete**



Visão de distribuição

A visão de distribuição exibe um diagrama de distribuição que mostra os nós e as conexões entre os nós. Onde um nó pode ser um hardware e uma conexão um cabo entre o hardware e a interface.

Gerando o código para as classes automaticamente

O **Visual Modeler** tem um assistente que gera automaticamente o código para as classes que você criou no Visual Modeler . Vejamos um exemplo:

Para poder usar este recurso você vai ter que ter criado um componente no VM e ter associado ao componente as classes pertinente. Eu vou criar um componente e uma classe - teste - usando o - **Class Wizard** - no menu Tools.(*Basta você clicar e seguir as orientações para criar as propriedades e métodos da classe*)

Após isto , para gerar o código para classe , selecione a classe criada e clique com o botão direito do mouse , selecionando a seguir a opção - **Generate Code.**(fig.1). Na janela - **Assign to new Component -** selecione Visual Basic EXE e clique em OK.(fig.2)

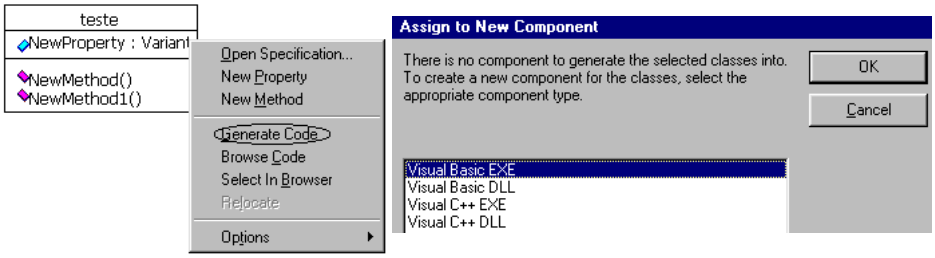
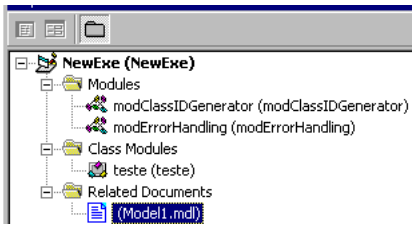


Fig.1 Fig.2

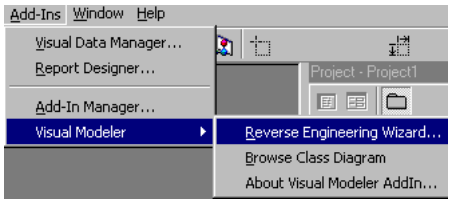
A seguir o assistente de geração de código irá entrar em ação , você deve ir confirmando as opções e no final terá o código gerado para a sua classe.



Fazendo engenharia Reversa

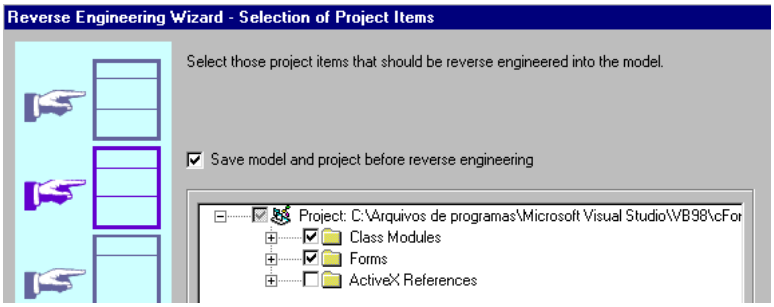
Você pode gerar o diagrama para uma aplicação feita em VB. Para fazer isto você deve carregar os add-ins : *Visual Modeler Add-ins e Visual Modeler Menus*. Vejamos como fazer o serviço:

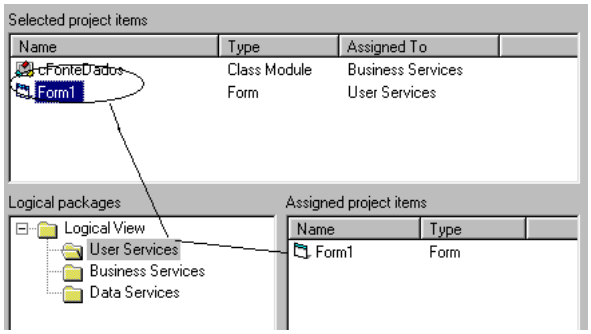
1-) Inicie o Visual Basic e carregue o projeto para o qual deseja realizar a engenharia reversa. Eu vou carregar um projeto que contém uma classe e um formulário.



2-) A seguir selecione a opção - **New** - e na janela - **Reverse Engineering Wizard - Welcome** - clique em **Next**

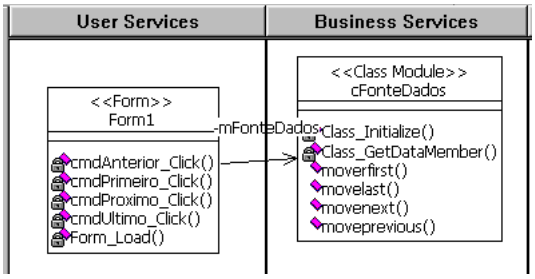
3-) Na janela **Reverse Engineering Wizard - Welcome - Selection of Project Items** selecione os items para o qual que deseja o codigo e clique em **Next**





4- Na próxima janela arraste os itens do projeto para uma dos pacotes lógicos e clique no botão - **Next**

Clique no botão - **Finish**- e a seguir no botão - **Close** . Pronto o seu diagrama já esta quentinho !



Embora simples o **Visual Modeler** permite realizar algumas tarefas que vão fazer você ganhar tempo e ... dinheiro !

Gostou ? até o próximo artigo ...

Referências:

- Seção VB .NET do Site Macoratti.net
- Super DVD .NET - A sua porta de entrada na plataforma .NET
- Super DVD Vídeo Aulas - Vídeo Aula sobre VB .NET, ASP .NET e C#
- Seção C# do site Macoratti.net
- Super DVD C#
- Super DVD Visual Basic
- Curso Básico VB .NET - Vídeo Aulas
- Curso C# Básico - Vídeo Aulas
- Seção de Jogos do site Macoratti .net
- Seção UML do site Macoratti.net
- UML - Diagrama de classes - Macoratti.net
- UML - Casos de Uso - Conceitos (revisão) - Macoratti.net
- UML - Conceitos Básicos III - Macoratti.net
- Modelando sistemas com UML - Macoratti.net
- Implementando Soluções OOP I

José Carlos Macoratti