

Abordagem Y

I. Visão Geral Abordagem Y

Tabela I – Visão Geral Abordagem Y

Abordagem Y			
Item	Sim	Não	Observação
Baseada em UML?	X		
Possui um Perfil UML definido?	X		
Possui um Processo definido?	X		
Utiliza Estereótipos?	X		Estereótipos específicos padrões para todos os modelos.
Possui Diretrizes?	X		Diretrizes específicas para cada modelo.
Permite representação formal de variabilidade?		X	

II. Estereótipos e Diretrizes

Nesta seção são apresentados os estereótipos para aplicação em diagrama de classes, existentes no perfil da abordagem Y por meio da Tabela II, em seguida são apresentados exemplos do uso destes, seguidos pelas diretrizes para cada tipo de modelo.

Tabela II – Estereótipos da Abordagem Y para Classes.

Estereótipos Abordagem Y Para Classes (Aplicados também aos demais modelos da abordagem)		
Estereótipo	Utilização	Exemplo
<<variationPoint>>	Representa o local em que ocorre uma variabilidade. Um ponto de variação está sempre associado a uma ou mais variantes.	Figura 1.
<<mandatory>>	A variante estará obrigatoriamente presente na configuração de qualquer produto da linha de produto.	Figura 1.
<<optional>>	A variante pode ou não estar presente na configuração de um produto da linha de produto. Variantes opcionais também podem ou não estar associadas a um ponto de variação.	Figura 3.
<<alternative_OR>>	Estão sempre associadas aos pontos de variação. Pelo menos uma das variantes deverá ser escolhida para resolver o ponto de variação, ou seja, para estar presente na configuração de um produto da linha de produto.	Figura 1.
<<alternative_XOR>>	Estão sempre associadas aos pontos de variação. Somente uma das variantes deverá ser escolhida para resolver o ponto de variação.	-
<<variability>>	Indica uma variabilidade existente em um modelo UML.	Figura 1 e 2.
<<requires>>	Indica um relacionamento de dependência (em UML) entre variantes no qual a variante dependente (origem da dependência) só existirá em uma configuração se a variante relacionada (destino da dependência) existir.	Figura 2.

<<mutex>>	Indica um relacionamento de dependência (em UML) entre variantes no qual a variante dependente (origem da dependência) só existirá em uma configuração se a variante relacionada (destino da dependência) obrigatoriamente não existir. São conhecidas como variantes mutuamente exclusivas.	Figura 2.
-----------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------

II.1 Exemplos

Classes

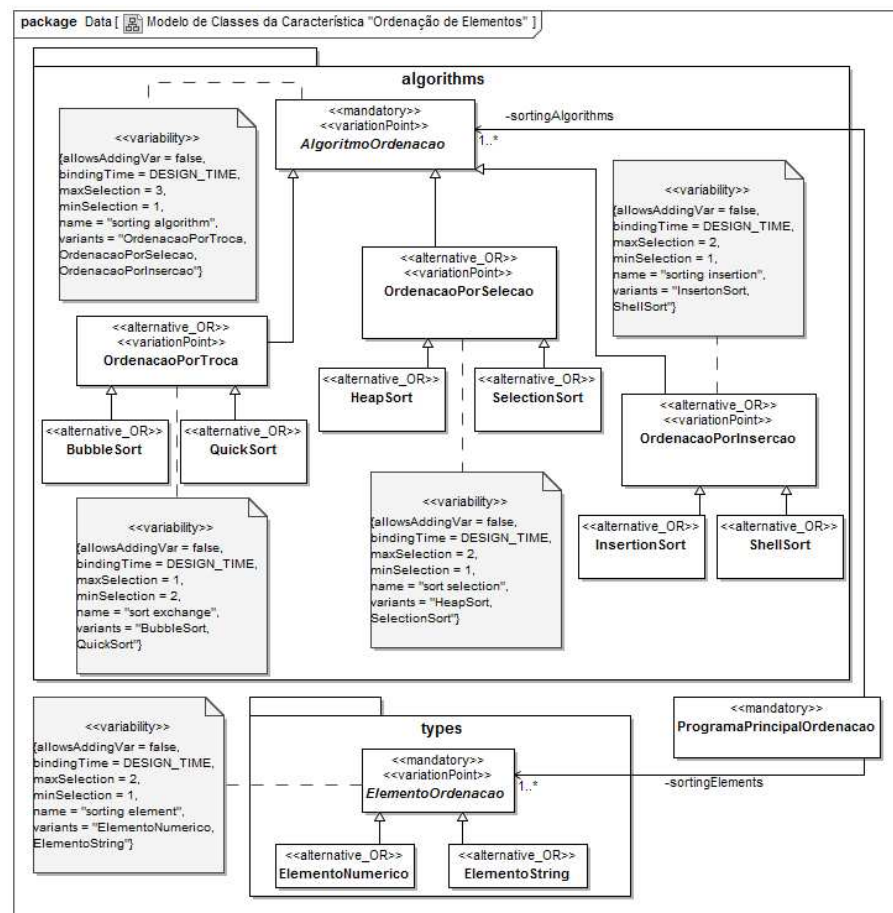


Figura 1 – Exemplo de Modelo de Variabilidade em Diagrama de Classes com a Abordagem Y.

Na Figura 1 observamos a aplicação da abordagem Y, e seus elementos. Passamos a analisar cada um deles, bem como as diretrizes presentes no processo da abordagem Y, que auxiliam sua utilização em outras LPs:

A classe **AlgoritmoOrdenacao** identifica uma classe obrigatória (<<mandatory>>) e representa também um ponto de variação (<<variationPoint>>), com três variantes. Estas variantes estão descritas no elemento comentário, relacionado a classe, por meio do *Tagged Value* (**variants**). As três variantes desta classe são **OrdenacaoPorTroca**, **OrdenacaoPorSelecao** e **OrdenacaoPorInsercao**. Todas estas são estereotipadas como <<alternative_OR>>, o que indica o tipo de restrição para tais variantes, neste caso, significa que ao menos uma ou todas elas podem solucionar o ponto de variação.

OrdenacaoPorTroca, **OrdenacaoPorSelecao** e **OrdenacaoPorInsercao**, além de variantes, são, por sua vez, pontos de variação (<<variationPoint>>), e assim, cada uma delas apresenta um comentário, que descreve as suas variantes (**variants**), bem como o nome da mesma (**name**). Neste caso, todas as variantes são marcadas como <<alternative_OR>> e, como anteriormente, uma delas, ao menos, deve ser selecionada ou todas.

A classe **ProgramaPrincipalOrdenacao**, representa uma classe obrigatória, portanto é marcada como <<mandatory>>, e estará presente em todos os produtos desta LP.

A classe **ElementoOrdenacao**, também é obrigatória (<<mandatory>>) e representa um ponto de variação (<<variationPoint>>), logo possui o elemento comentário ligado a ela, com o estereótipo <<variability>>, que identifica os dados da variabilidade, que é nomeada, por exemplo, de "sorting element" e possui duas classes variantes (**varinats**): **ElementoNumerico** e **ElementoString**, marcadas como variantes alternativas <<alternative_OR>>, onde, ambas podem ser selecionadas, ou ao menos uma.

Desta forma, as variabilidades são identificadas por meio do comentário UML, estereotipada com <<variability>>. **Estas notas são inseridas em todas as variabilidades.**

Diretrizes para Diagrama de Classes - As diretrizes especificadas para auxiliar na identificação das variabilidades em diagramas de classes são expressas abaixo:

CL1. Em modelos de classes, pontos de variação e suas variantes são identificadas nos seguintes relacionamentos:

- a) **generalização**, os classificadores mais gerais sugerem os pontos de variação, enquanto os mais específicos são as variantes;
- b) **realização de interface**, os "suppliers" (especificações) sugerem pontos de variação e as implementações (clientes) são as variantes;
- c) **agregação**, as instâncias tipadas com losangos não preenchidos sugerem pontos de variação e as instâncias associadas são as variantes; e
- d) **composição**, as instâncias tipadas com losangos preenchidos sugerem pontos de variação e as instâncias associadas são as variantes.

CL2. Elementos de modelos de classes, relacionados à associações nas quais os seus atributos *aggregationKind* possuem valor *none*, ou seja, não representam nem agregação nem composição, sugerem variantes obrigatórias ou opcionais. Na Figura 1, a classe **ProgramaPrincipalOrdenacao**, é um exemplo de variante obrigatória.

CL2.1 Elementos de modelos de classes, relacionadas à associações nas quais os seus atributos *aggregationKind* possuem valor * (zero ou vários) ou 0..n onde n é um número inteiro qualquer, diferente de zero, sugerem que tal classe é opcional;

CL3. Variantes que, ao serem selecionadas para fazer parte de um produto, exigem a presença de outra(s) determinada(s) variante(s) devem ter seus relacionamentos de dependência marcados com o estereótipo <<requires>>;

CL4. Variantes mutuamente exclusivas para um determinado produto devem ter seus relacionamentos de dependência marcados com o estereótipo <<mutex>>.

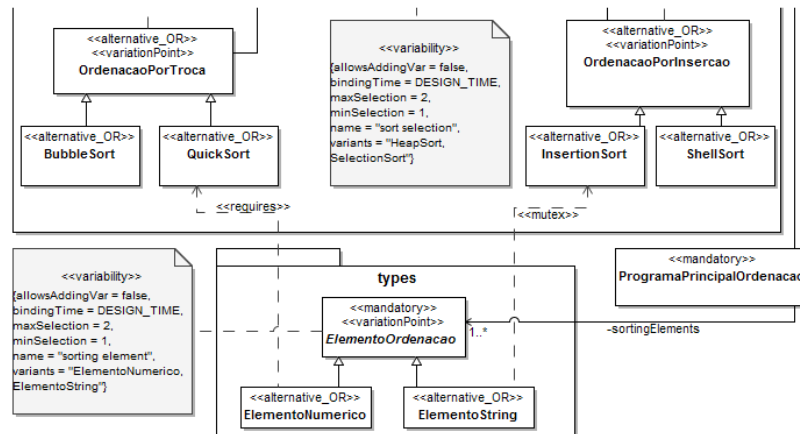


Figura 2 – Exemplo de Identificação de Variabilidade em Classes – *requires* e *mutex*.

Na Figura 2, que representa fragmento de um diagrama de classes, nele notas o uso dos estereótipos para identificar a dependência entre classes, bem como a seleção mutuamente exclusiva.

A classe **ElementoNumerico** requer a presença da classe **QuickSort**, para que possa ser incluída no produto, já a classe **ElementoString** restringe que a classe **InsertionSort** não seja inserida no produto, para que possa ser selecionada, ou seja, se selecionada a classe **ElementoString** como variante para o ponto de variação **ElementoOrdenacao**, a classe **InsertionSort**, que é ponto de variação de outra classe, não pode ser selecionada. No caso da classe **ElementoNumerico**, ser selecionada, a classe **QuickSort**, deverá ser selecionada como variante para o ponto de variação a qual pertence.

A Figura 3 representa a aplicação do estereótipo `<<optional>>`. A classe **SaveGame** é opcional, e assim é marcada como uma variabilidade, pelo comentário da UML – quando opcional a classe pode ou não estar inserida no produto da LP.

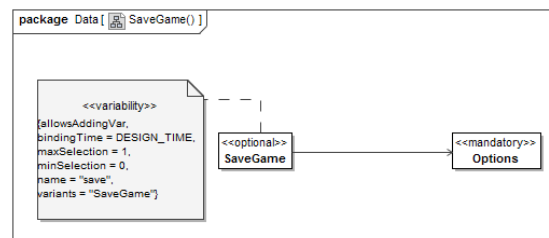


Figura 3 – Exemplo de Identificação de Variabilidade em Classes – *optional*.