A história de UML e seus diagramas

Thânia Clair de Souza Vargas

Departamento de Informática e Estatística
Universidade Federal de Santa Catarina (UFSC) – Florianópolis, SC – Brazil
thania@inf.ufsc.br

Abstract. *This paper describes the UML history since the decade of 1990 so far today. It presents the organization of the thirteen UML diagrams, classifying them in structural and behavioral diagrams. The four documents belonging to the specification are cited and explained as well. Finally, each UML 2 diagram is described in detail.*

Resumo. *Este artigo descreve a história de UML desde a década de 1990 até o momento atual. Apresenta-se a organização dos treze diagramas de UML, classificando-os em diagramas estruturais e comportamentais. Os quatro documentos pertencentes à especificação também são mencionados e explicados. Por fim, cada diagrama de UML 2 é descrito em detalhes.*

1. Introdução

Modelagem de software é a atividade de construir modelos que expliquem as características ou o comportamento de um software ou de um sistema de software. Na construção do software os modelos podem ser usados na identificação das características e funcionalidades que o software deverá prover (análise de requisitos), e no planejamento de sua construção. Freqüentemente a modelagem de software usa algum tipo de notação gráfica e são apoiados pelo uso de ferramentas.

A modelagem de software normalmente implica a construção de modelos gráficos que simbolizam os artefatos dos componentes de software utilizados e os seus interrelacionamentos. Uma forma comum de modelagem de programas orientados a objeto é através da linguagem unificada UML.

A UML (Unified Modeling Language) é uma linguagem para especificação, documentação, visualização e desenvolvimento de sistemas orientados a objetos. Sintetiza os principais métodos existentes, sendo considerada uma das linguagens mais expressivas para modelagem de sistemas orientados a objetos. Por meio de seus diagramas é possível representar sistemas de softwares sob diversas perspectivas de visualização. Facilita a comunicação de todas as pessoas envolvidas no processo de desenvolvimento de um sistema - gerentes, coordenadores, analistas, desenvolvedores - por apresentar um vocabulário de fácil entendimento (OMG, 2005a) (OMG,

2005b) (OMG, 2005c) (OMG, 2006).

2. História de UML

No início da utilização do paradigma de orientação a objetos, diversos métodos foram apresentados para a comunidade. Chegaram a mais de cinqüenta entre os anos de 1989 a 1994, porém a maioria deles cometeu o erro de tentar estender os métodos estruturados da época. Com isso os maiores prejudicados foram os usuários que não conseguiam encontrar uma maneira satisfatória de modelar seus sistemas. Foi a partir da década de 90 que começaram a surgir teorias que procuravam trabalhar de forma mais ativa com o paradigma da orientação a objetos. Diversos autores famosos contribuíram com publicações de seus respectivos métodos.

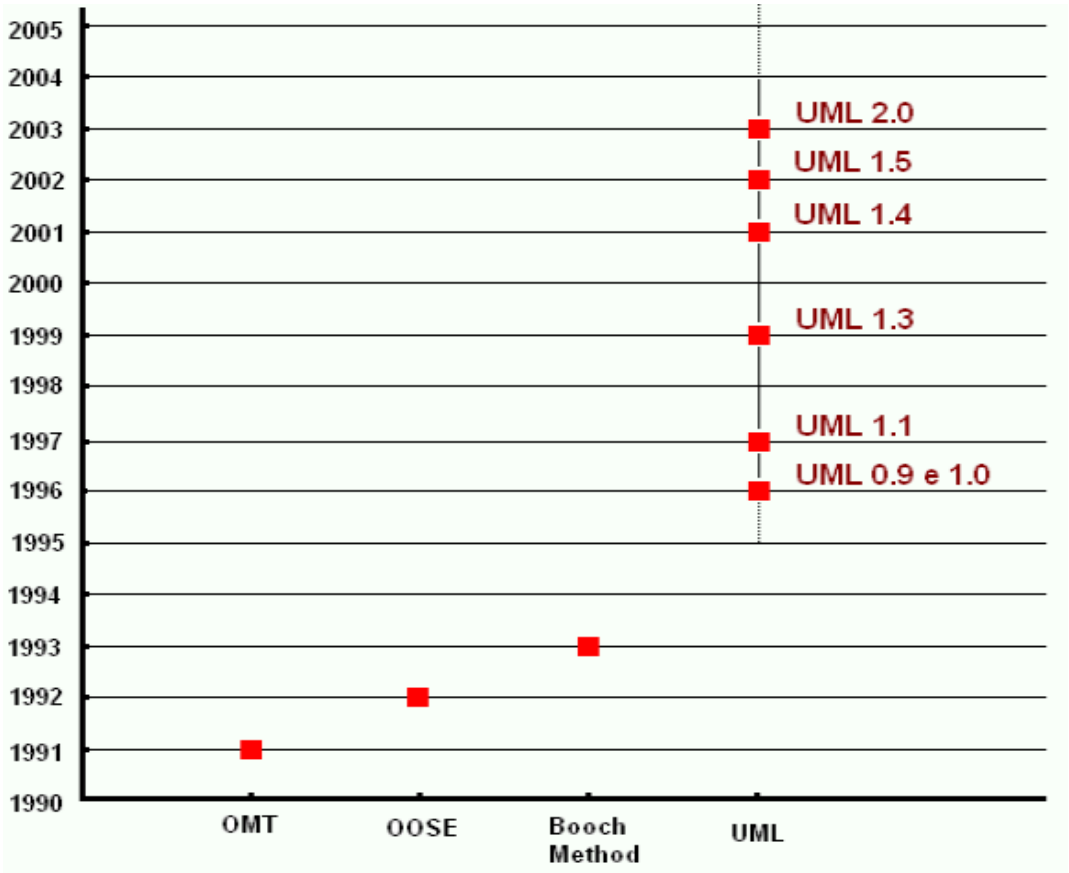

Figure 1. Linha do Tempo de UML

Por volta de 1993 existiam três métodos que mais cresciam no mercado, eram eles: Booch'93 de Grady Booch, OMT-2 de James Rumbaugh e OOSE de Ivar Jacobson. Cada um deles possuía pontos fortes em algum aspecto. O OOSE possuía foco em casos de uso (*use cases*), OMT-2 se destaca na fase de análise de sistemas de informação e Booch'93 era mais forte na fase de projeto. O sucesso desses métodos foi, principalmente, devido ao fato de não terem tentado estender os métodos já existentes.

Seus métodos já convergiam de maneira independente, então seria mais produtivo continuar de forma conjunta (SAMPAIO, 2007).

Em outubro de 1994, começaram os esforços para unificação dos métodos. Já em outubro de 1995, Booch e Rumbaugh lançaram um rascunho do "Método Unificado" unificando o Booch'93 e o OMT-2. Após isso, Jacobson se juntou a equipe do projeto e o "Método Unificado" passou a incorporar o OOSE. Em junho de 1996, os três amigos, como já eram conhecidos, lançaram a primeira versão com os três métodos - a versão 0.9 que foi batizada como UML (FOWLER, 2003). Posteriormente, foram lançadas várias novas versões na qual podemos acompanhar através do gráfico na figura 1.

OMG3 lançou uma RFP (Request for Proposals) para que outras empresas pudessem contribuir com a evolução da UML, chegando à versão 1.1. Após alcançar esta versão, a OMG3 passou a adotá-la como padrão e a se responsabilizar (através da RTF – Revision Task Force) pelas revisões. Essas revisões são, de certa forma, "controladas" a não provocar uma grande mudança no escopo original. Se observarmos as diferenças entre as versões atualmente, veremos que de uma para a outra não houve grande impacto, o que facilitou sua disseminação pelo mundo.

3. Estrutura da Especificação

A especificação de UML é composta por quatro documentos: infra-estrutura de UML(OMG, 2006), superestrutura de UML (OMG, 2005c), *Object Constraint Language* (OCL) (OMG, 2005a) e Intercâmbio de Diagramas (OMG, 2005b).

- *Infra-estrutura de UML:* O conjunto de diagramas de UML constitui uma linguagem definida a partir de outra linguagem que define os elementos construtivos fundamentais. Esta linguagem que suporta a definição dos diagramas é apresentada no documento infra-estrutura de UML.
- *Superestrutura de UML:* Documento que complementa o documento de infra-estrutura e que define os elementos da linguagem no nível do usuário.
- *Linguagem para Restrições de Objetos (OCL):* Documento que apresenta a linguagem usada para descrever expressões em modelos UML, com pré-condições, pós-condições e invariantes.
- *Intercâmbio de diagramas de UML:* Apresenta uma extensão do meta-modelo voltado a informações gráficas. A extensão permite a geração de uma descrição no estilo XMI orientada a aspectos gráficos que, em conjunto com o XMI original permite produzir representações portáveis de especificações UML.

4. Organização dos diagramas de UML 2

A linguagem UML 2 é composta por treze diagramas, classificados em diagramas estruturais e diagramas de comportamento. A figura 2 apresenta a estrutura das categorias utilizando a notação de diagramas de classes (OMG, 2006).

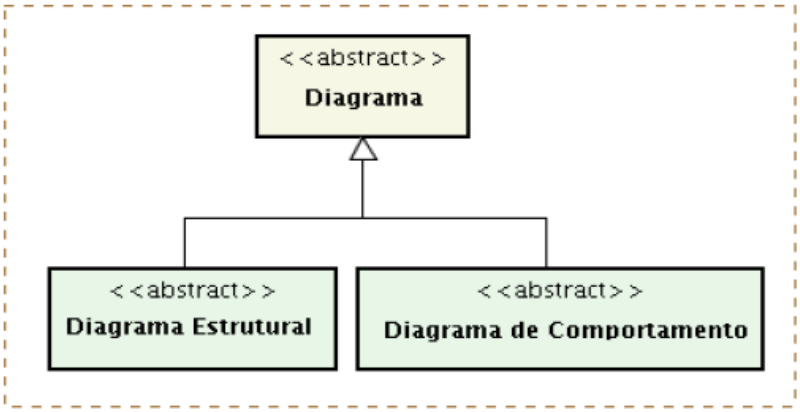

Figura 2. Organização Geral dos Diagramas de UML 2

Os diagramas estruturais, ilustrados na imagem 3 conforme a especificação OMG (OMG, 2006), tratam o aspecto estrutural tanto do ponto de vista do sistema quanto das classes. Existem para visualizar, especificar, construir e documentar os aspectos estáticos de um sistema, ou seja, a representação de seu esqueleto e estruturas "relativamente estáveis". Os aspectos estáticos de um sistema de software abrangem a existência e a colocação de itens como classes, interfaces, colaborações, componentes.

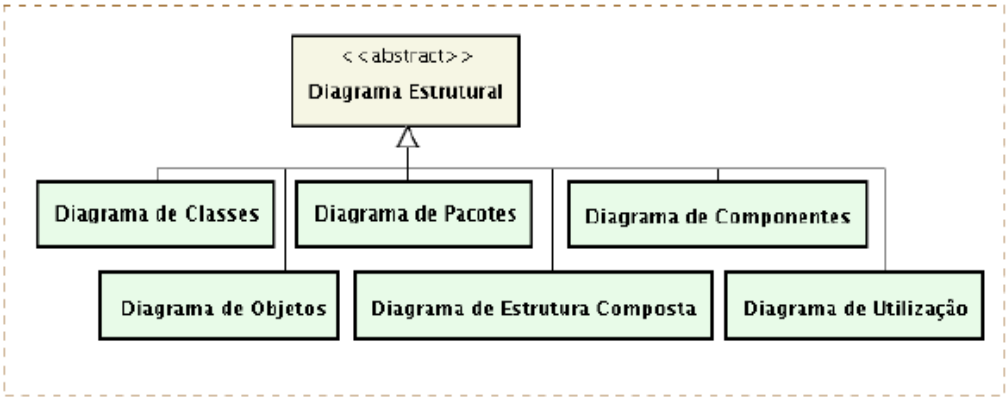

Figura 3. Diagramas Estruturais

Os diagramas de comportamento, ilustrados na imagem 4 conforme a especificação OMG (OMG, 2006), são voltados a descrever o sistema computacional modelado quando em execução, isto é, como a modelagem dinâmica do sistema. São

usados para visualizar, especificar, construir e documentar os aspectos dinâmicos de um sistema que é a representação das partes que “sofrem alterações”, como por exemplo o fluxo de mensagens ao longo do tempo e a movimentação física de componentes em uma rede.

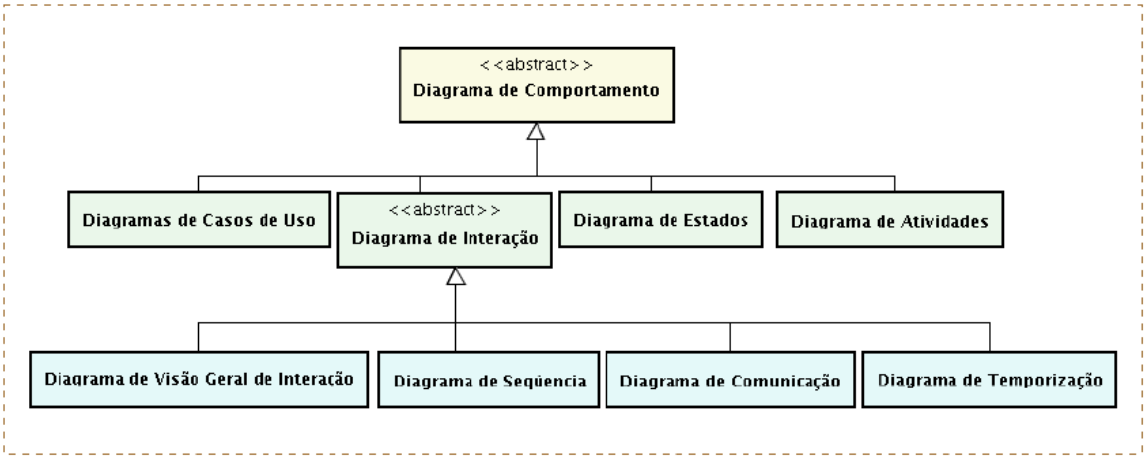

Figura 4. Diagramas Comportamentais

5. Diagramas de UML 2

Um diagrama é uma representação gráfica de um conjunto de elementos (classes, interfaces, colaborações, componentes, nós, etc) e são usados para visualizar o sistema sob diferentes perspectivas. A UML define um número de diagramas que permite dirigir o foco para aspectos diferentes do sistema de maneira independente. Se bem usados, os diagramas facilitam a compreensão do sistema que está sendo desenvolvido.

Nas próximas seções, serão sintetizadamente apresentados os diagramas que compõem a linguagem UML 2.

5.1. Diagrama de Classes

Um diagrama de classes é um modelo fundamental de uma especificação orientada a objetos. Produz a descrição mais próxima da estrutura do código de um programa, ou seja, mostra o conjunto de classes com seus atributos e métodos e os relacionamentos entre classes. Classes e relacionamentos constituem os elementos sintáticos básicos do diagrama de classes (SILVA, 2007).

5.2. Diagrama de Objetos

O diagrama de objetos consiste em uma variação do diagrama de classes em que, em vez de classes, são representadas instâncias e ligações entre instâncias. A finalidade é descrever um conjunto de objetos e seus relacionamentos em um ponto no tempo.

Contém objetos e vínculos e são usados para fazer a modelagem da visão de projeto estática de um sistema a partir da perspectiva de instâncias reais ou prototípicas.

5.3. Diagrama de Pacotes

O pacote é um elemento sintático voltado a conter elementos sintáticos de uma especificação orientada a objetos. Esse elemento foi definido na primeira versão de UML para ser usado nos diagramas então existentes, como diagrama de classes, por exemplo. Na segunda versão da linguagem, foi introduzido um novo diagrama, o diagrama de pacotes, voltado a conter exclusivamente pacotes e relacionamentos entre pacotes (SILVA, 2007). Sua finalidade é tratar a modelagem estrutural do sistema dividindo o modelo em divisões lógicas e descrevendo as interações entre ele em alto nível.

5.4. Diagrama de Estrutura Composta

O diagrama de estrutura composta fornece meios de definir a estrutura de um elemento e de focalizá-la no detalhe, na construção e em relacionamentos internos. É um dos novos diagramas propostos na segunda versão de UML, voltado a detalhar elementos de modelagem estrutural, como classes, pacotes e componentes, descrevendo sua estrutura interna.

O diagrama de estrutura composta introduz a noção de “porto”, um ponto de conexão do elemento modelado, a quem podem ser associadas interfaces. Também utiliza a noção de “colaboração”, que consiste em um conjunto de elementos interligados através de seus portos para a execução de uma funcionalidade específica – recurso útil para a modelagem de padrões de projeto (SILVA, 2007).

5.5. Diagrama de Componentes

O diagrama de componentes é um dos dois diagramas de UML voltados a modelar software baseado em componentes. Tem por finalidade indicar os componentes do software e seus relacionamentos. Este diagrama mostra os artefatos de que os componentes são feitos, como arquivos de codigo fonte, bibliotecas de programação ou tabelas de bancos de dados. As interfaces é que possibilitam as associações entre os componentes.

5.6. Diagrama de Implantação

O diagrama de utilização, também denominado diagrama de implantação, consiste na organização do conjunto de elementos de um sistema para a sua execução. O principal elemento deste diagrama é o nodo, que representa um recurso computacional. Podem ser representados em um diagrama tantos os nodos como instâncias de nodos.

O diagrama de implatação é util em projetos onde há muita interdepedência entre pedaços de hardware e software.

5.7. Diagrama de Casos de Uso

O diagrama de casos de uso especifica um conjunto de funcionalidades, através do elemento sintático “casos de uso”, e os elementos externos que interagem com o sistema, através do elemento sintático “ator” (SILVA, 2007). Além de casos uso e atores, este diagrama contém relacionamentos de dependência, generalização e associação e são basicamente usados para fazer a modelagem de visão estática do caso de uso do sistema. Essa visão proporciona suporte principalmente para o comportamento de um sistema, ou seja, os serviços externamente visíveis que o sistema fornece no contexto de seu ambiente. Neste caso os diagramas de caso de uso são usados para fazer a modelagem do contexto de um sistema e fazer a modelagem dos requisitos de um sistema.

5.8. Diagrama de Seqüência

O diagrama de seqüência mostra a troca de mensagens entre diversos objetos, em uma situação específica e delimitada no tempo. Coloca ênfase especial na ordem e nos momentos nos quais mensagens para os objetos são enviadas.

Em diagramas de seqüência, objetos são representados através de linhas verticais tracejadas (denominadas como linha de existência), com o nome do objeto no topo. O eixo do tempo é também vertical, aumentando para baixo, de modo que as mensagens são enviadas de um objeto para outro na forma de setas com a operação e os nomes dos parâmetros.

5.8. Diagrama de Máquina de Estados

O diagrama de máquina de estados tem como elementos principais o estado, que modela uma situação em que o elemento modelado pode estar ao longo de sua existência, e a transição, que leva o elemento modelado de um estado para o outro. O diagrama de máquina de estados vê os objetos como máquinas de estados ou autümatos finitos que poderão estar em um estado pertencente a uma lista de estados finita e que poderão mudar o seu estado através de um estímulo pertencente a um conjunto finito de estímulos.

5.8. Diagrama de Comunicação

Os elementos de um sistema trabalham em conjunto para cumprir os objetos do sistema e uma linguagem de modelagem precisa poder representar esta característica. O diagrama de comunicação é voltado a descrever objetos interagindo e seus principais

elementos sintáticos são “objeto” e “mensagem”. Corresponde a um formato alternativo para descrever interação entre objetos. Ao contrário do diagrama de seqüência, o tempo não é modelado explicitamente, uma vez que a ordem das mensagens é definida através de enumeração. Vale ressaltar que tanto o diagrama de comunicação como o diagrama de seqüência são diagramas de interação.

5.8. Diagrama de Atividades

O diagrama de atividades representa a execução das ações e as transições que são acionadas pela conclusão de outras ações ou atividades.

Uma atividade pode ser descrita como um conjunto de ações e um conjunto de atividades. A diferença básica entre os dois conceitos que descrevem comportamento é que a ação é atômica, admitindo particionamento, o que não se aplica a atividade, que pode ser detalhada em atividades e ações (SILVA, 2007).

5.8. Diagrama de Visão Geral de Integração

O diagrama de visão geral de interação é uma variação do diagrama de atividades, proposto na versão atual de UML. Seus elementos sintáticos são os mesmos do diagrama de atividades. As interações que fazem parte do diagrama de visão geral de interação podem ser referências a diagramas de interação existentes na especificação tratada.

5.8. Diagrama de Temporização

O diagrama de temporização consiste na modelagem de restrições temporais do sistema. É um diagrama introduzido na segunda versão de UML, classificado como diagrama de interação. Este diagrama modela interação e evolução de estados.

6. Conclusão

Embora a UML defina uma linguagem precisa, ela não é uma barreira para futuros aperfeiçoamentos nos conceitos de modelagem. O desenvolvimento da UML foi baseado em técnicas antigas e marcantes da orientação a objetos, mas muitas outras influenciarão a linguagem em suas próximas versões. Muitas técnicas avançadas de modelagem podem ser definidas usando UML como base, podendo ser estendida sem se fazer necessário redefinir a sua estrutura interna.

A UML está sendo a base para muitas ferramentas de desenvolvimento, incluindo modelagem visual, simulações e ambientes de desenvolvimento. Em breve, ferramentas de integração e padrões de implementação baseados em UML estarão disponíveis para qualquer um.

A UML integrou muitas idéias adversas, e esta integração acelera o uso do desenvolvimento de softwares orientados a objetos.

References

FOWLER, M. UML Distilled: A Brief Guide to the Standard Object Modeling Language. [S.l.: s.n.], 2003.

OMG. OCL 2.0 Specification. 2005.

OMG. Unified Modeling Language: diagram interchange. 2005.

OMG. Unified Modeling Language: superstructure. 2005.

OMG. Unified Modeling Language: infrastructure. 2006.

ROCHA, A. R. C.; MALDONADO, J. C.; WEBER, K. C. Qualidade de Software. São Paulo: a Pretince Hall, 2001.

SAMPAIO, M. C. História de UML. http://www.dsc.ufcg.edu.br/sampaio: [s.n.], 2007.

SILVA, R. P. e. UML 2 em Modelagem Orientada a Objetos. Florian´ polis: Visual Books, 2007.