

Buscando novos caminhos por meio da UML

Por Ana Cristina Melo

Vivemos uma grande evolução tecnológica e, com ela, um aumento exponencial na demanda de novos softwares. Precisamos desenvolver mais — e em menos tempo. Nossos softwares precisam carregar um "algo a mais". Assim, é hora de repensar antigos modelos de desenvolvimento e gerência.

Este artigo trata de como a **orientação a objetos** e a **UML** podem melhorar o quadro atual de desenvolvimento de sistemas.

Por que Orientação a Objetos?

Atualmente, será que conseguimos imaginar instituições financeiras, grandes lojas de departamentos, companhias aéreas, entre tantos outros setores, sem associá-las à tecnologia?

A resposta a esta pergunta nos leva ao quadro atual de desenvolvimento de sistemas. Precisamos, cada vez mais, oferecer serviços de melhor qualidade, pois muitos setores dependem de nossos sistemas. Todavia, nos deparamos com um problema antigo: como desenvolver aplicações que atendam às novas ondas da Tecnologia de Informação, se ainda se houve falar em **crise de software**?

Para resolver essa questão, temos que levar em conta que não podemos desenvolver como há 15 anos, pois a nossa demanda e necessidades tecnológicas são muito superiores àquela época. Precisamos de sistemas desenvolvidos mais rapidamente, com custos menores, que permitam às organizações gerenciar melhor e competir com armas poderosas. Sabemos que a Análise Estruturada e Essencial não conseguem suprir essas necessidades.

A área de Hardware há muito tempo encontrou seu 'caminho das pedras', quando aprenderam a componentizar e reutilizar esforços. E é esse o caminho que devemos trilhar no software. Devemos componentizar, reutilizar esforços e para isso, necessitamos da orientação a objetos.

Com a orientação a objetos, deixamos de abstrair dados e funções como elementos independentes. Focalizamos personagens de nosso mundo real, que são os objetos. A partir deles, idealizamos nossos sistemas. Passamos, então, a pensar da mesma forma que visualizamos o mundo real.

O que é UML ?

Sabemos da importância da orientação a objetos, mas não basta termos uma Mercedes e não sabermos dirigi-la; ou usá-la como quem usa um Fusca (sem desmerecer o pobrezinho!).

Assim, não adianta o paradigma da orientação a objetos se não fizermos uma boa análise a partir dele. E foi tentando acertar e criar o melhor método, que, no início da década de 90, chegamos a ter mais de 50 deles no mercado, gerando a então “guerra dos métodos”. Todavia, a grande maioria pecava por tentar criar adaptações da análise estruturada.

No início da década de 90, métodos que buscavam uma nova forma de pensar começaram a sobressair. Assim, os métodos Booch (de Grady Booch), OMT (de James Rumbaugh) e OOSE (de Ivar Jacobson) passaram a ter a maior fatia do mercado. Percebendo as semelhanças existentes em seus métodos, Booch e Rumbaugh decidiram unificá-los. Surgiu o Método Unificado 0.8. Logo a seguir, a eles uniu-se Jacobson, surgindo a versão 0.9. Contribuições de outros metodologistas e de diversas empresas da área (como IBM, HP, Microsoft e Oracle, entre outras), fizeram com que a idéia evoluísse e os conceitos fossem melhorados. Surgiu a UML 1.0 - Unified Modeling Language (Linguagem de Modelagem Unificada).

Assim, ganhávamos uma linguagem de modelagem — e não mais um método. Uma linguagem que pudesse ser usada com qualquer processo de desenvolvimento. A UML nos diz o que é possível fazer sem dizer como e em que ordem – esta tarefa fica a cargo do processo. A UML, na sua versão 1.1, foi padronizada pelo **OMG** (Object Management Group) e já sofreu algumas atualizações (mais documentais do que estruturais), estando atualmente na sua versão 1.4.

Por que usar UML ?

Uma das grandes vantagens da UML é o fato dela ser totalmente extensível e adaptável. Você não adapta sua modelagem à UML. Você seleciona os elementos da UML que melhor expressarão sua modelagem. E se para isto for necessário estender os modelos da UML, você o faz sem perder compreensão. Qualquer um que leia seu modelo, entenderá que foi feita uma extensão.

Além disso, acabam-se as fronteiras entre as fases de análise e projeto. Um mesmo diagrama é utilizado em todas as fases, mudando-se, apenas, sua visão.

O mapeamento direto dos modelos para as linguagens de programação orientadas a objeto e vice-versa também é um dos grandes ganhos da UML.

Esses são alguns dos inúmeros benefícios que a UML nos fornece, sem que percamos a liberdade de criar.

Estrutura da UML

A UML tem sua estrutura basicamente dividida em: elementos de modelo (por exemplo: classes, casos de uso, componentes, etc), diagramas e relacionamentos. Possuímos, ainda, os mecanismos de extensão, que dão ao desenvolvedor a flexibilidade de estender seu modelo.

Os **diagramas da UML** são: Diagrama de Casos de Uso, Diagrama de Classes, Diagrama de Objetos, Diagrama de Gráfico de Estados, Diagrama de Atividades, Diagrama de Seqüências, Diagrama de Colaborações, Diagrama de Componentes e Diagrama de Implantação.

Esses diagramas permitem a modelagem de todas as fases do projeto. Neles modelamos nossos elementos, como classes e casos de uso, apresentando seus relacionamentos.

Relembro que temos a liberdade de utilizar os diagramas conforme as necessidades de nosso projeto.

Modelando Requisitos

A maior dificuldade em modelarmos um sistema não está nos diagramas que temos que desenhar, no código que devemos criar ou nas bases de dados que devemos projetar. Na realidade, está nos requisitos que devemos gerenciar.

Lidamos com a expectativa humana. Temos que entender o que o usuário quer e, muito mais importante, demonstrar que realmente entendemos seus desejos.

O **caso de uso** veio solucionar esse problema, pois oferece ao desenvolvedor e usuário um padrão único, compreensível para ambos, e não ambíguo, que descreve com precisão as funcionalidades do sistema. O processo de validação de requisitos tornou-se mais preciso com os casos de uso.

Mais do que modelar requisitos, hoje o caso de uso é a base sólida da modelagem em UML. O caso de uso não serve apenas para organizar as funcionalidades do sistema. Ele serve de base para a modelagem dos objetos, além de ser a base para os casos de teste e validação de todo projeto.

Entre tantos outros ganhos, o caso de uso é um bom exemplo de que, repensando o processo, podemos chegar a modelos de melhor qualidade.

Conclusão

Muito se fala em modelos de qualidade. Todavia, precisamos ajustar as engrenagens de um processo de desenvolvimento, para que tenhamos um produto final de qualidade. Uma das engrenagens mais importantes é o paradigma de desenvolvimento de sistemas. Precisamos produzir mais, em menos tempo, e sem erros. Precisamos acompanhar a evolução tecnológica, sem que para isso tenhamos que nos sacrificar.

Um bom começo é voltar a atenção para as tecnologias que o mercado já aprovou. A orientação a objetos e a UML não podem mais ser enxergadas como modismo (até porque nunca foram). É hora de repensarmos nossos conceitos e, enfim, estarmos um passo à frente do desenvolvimento.