

Prefácio

A motivação para escrever este texto, foi a reiterada solicitação de meus alunos que pediam a criação de um texto que abordasse a UML dentro da perspectiva da construção de um software.

Alunos reclamavam que ou os textos sobre UML eram extensos e dispersos demais, ou resumidos demais, deixando muitas questões obscuras. Exemplos eram dados utilizando-se softwares complicados de se entender e ou vários softwares em diferentes momentos.

Minha resposta sempre foi: - Meus queridos alunos, não é possível em um único livro se abordar tudo a respeito de processo de construção de software e a UML inserida nele. Vocês têm uma gama de cerca de dez livros a ler! Sinto muito.

Como dou aulas e consultoria, isso me parecia preguiça da parte dos alunos, porém as solicitações continuaram e percebi que tinham sentido. Isso aconteceu quando vi dois programadores (*já treinados!*), cada um com um livro na mão, tentando acompanhar a seqüência de suas atividades.

É claro que o trabalho de *mentoring*, elimina muitas dúvidas aparecidas em encruzilhadas do desenvolvimento, como, por exemplo: "Que artefato devo produzir agora?"

Ocorre que muitos são desenvolvedores solitários ou que trabalham em pequenos grupos que não podem pagar um mentor e, apesar de já serem treinados, as dúvidas persistem.

Um ano se passou desde a primeira solicitação até a conclusão final deste trabalho.

Aviso que este texto aborda a UML 2.0 de forma prática, já testada na vida real. Não apresenta teorias obscuras sobre aspectos que o grosso do público de análise e de programação jamais verá.

Todos os diagramas apresentam exemplos de um único software, um site de aluguel de DVDs na Web. O único momento em que não uso esse exemplo é quando falo de teoria de classes. Isso ocorre porque não é possível que todos os conceitos da teoria de classes sejam apresentados dentro de uma única aplicação. Esta teria de ser enorme! Então, somente neste capítulo eu uso exemplos diferentes do software adotado. Depois retomo o software de aluguel de DVDs na Web até o banco de dados.

Outro aspecto a salientar é que este livro não tem glossário; se um conceito é necessário, forneço-o na hora em que o estamos usando. Faz parte do texto o significado, a notação e a semântica. Além disso, procuro mostrar qual a finalidade do conceito e, finalmente, como usá-lo.

Este trabalho é um norte para você se debruçar e descobrir o caminho a ser adotado na sua empresa. Procuro neste texto deixar claro, a tudo a que me refiro: O que é, para que serve e como se usa.

Page-Jones [10], dizia que, tempos após escrever um livro, mudou de idéia sobre várias coisas. Isso pode acontecer comigo. Nesse caso, prometo alterar o trabalho. Porém, além dessa situação, é possível que você encontre alguma forma de colaborar com este texto brasileiro sobre UML 2.0 e como usá-la. Sinta-se à vontade para entrar em contato comigo e expor suas opiniões. Farei o possível

2 Desenvolvendo software com UML 2.0 – Definitivo

para responder a todos os e-mails. Apenas tenha em mente o caráter prático deste texto. Prático, no sentido de que é de uso comum no dia-a-dia de analistas de negócios e programadores.

? **O que este livro não é?**

Este livro não é um tratado minucioso sobre UML 2.0. A documentação *Superstructure* da OMG tem cerca de 650 páginas e não é o único documento da OMG sobre o assunto. Nosso texto, aborda a parte técnica, a notação e a semântica, se estas são necessárias ao exemplo em foco.

Este livro não é um tratado sobre o Processo Unificado, seria, no mínimo, irresponsável dizer isso. Recomendo, para um estudo aprofundado, as literaturas referenciadas ^{4,5,45}. Principalmente o autor Kendall Scott, que fez um ótimo resumo do Processo Unificado.

Quando me perguntam qual processo é o melhor, respondo que uma mescla é muito bem-vinda. É assim que o processo ICONIX propõe um diagrama de seqüência para cada Caso de Uso. Acho perfeito o *approach*, você verá isso no capítulo que trata de diagrama de seqüência. O Processo Unificado propõe um meio iterativo e incremental. Iterativo porque passamos por iterações (1,2,3,4,...) em cada fase da estrutura. Incremental porque a última iteração deve agregar valor à iteração anterior. Perfeito. É assim que fugimos do antigo processo 'Queda D'Água'. A *Extreme Programming - XP* sugere a simplicidade, ou seja, vamos pensar na parte crítica mas na coisa mais simples que possa funcionar e resolver os problemas de agora. Perfeito. O RUP pensa primeiro em definir a arquitetura e depois substanciar essa arquitetura com modelos e solução. Ou seja, pensamos nos elementos de: desempenho, escalabilidade, reuso e restrições econômicas e tecnológicas, iniciando-se pela "Visão" do software. Perfeito.

Poderíamos continuar com a relação, informando cada processo existente e suas qualidades. Mas não se engane, por trás das qualidades existem... **Defeitos**. As reclamações são as mais variadas e dependem do caso. Nessas situações, usamos a criatividade brasileira; que nos coloca entre os melhores desenvolvedores de software do mundo. Descubra o que cada processo tem de melhor e separe o que cada um tem de ruim. Fique atento aos momentos de quando usar um e evitar o outro, mas principalmente use sempre a UML seja qual for o processo. Por exemplo a XP não recomenda o uso da UML, **use-a!**

Este livro não fala de padrões de desenvolvimento de software. Esse assunto poderá ser abordado em futuras edições.

? **O que este livro tem?**

Sugiro a leitura deste livro de forma seqüencial, capítulo a capítulo pois em todos procuro colocar segredos aqui e ali, mas fique à vontade para uma leitura salteada, já que o texto também pode ser usado como referência.

Vamos dar um passeio pelos capítulos deste livro, assim:

- ☞ Capítulo 1 – Conhecimentos Iniciais. Informa, de forma abreviada, sobre a história da UML e suas correlações com o Processo Unificado. A palavra

3 Prefácio

informa aqui é precisa, isto quer dizer que a preocupação não é formação mas sim informação sobre esses temas, a fim de levar o aluno a esclarecimentos sobre o que esses termos significam. Também são relacionados todos os diagramas da UML 2.0 e a ordenação em que serão abordados neste texto;

- ✍ Capítulo 2 – Documentos iniciais de um software. Aborda os documentos iniciais de software e suas correlações com a UML. Principalmente, a criação do documento visão e seu uso;
- ✍ Capítulo 3 – Caso de Uso. Aborda os princípios dos Casos de Uso e dá significação para o documento Visão nos primeiros diagramas;
- ✍ Capítulo 4 – Diagrama de Atividades e Descrição dos Casos de Uso. Aborda o possível uso do Diagrama de Atividades para a descrição dos Casos de Uso; adicionalmente as últimas modificações introduzidas nesse diagrama pela UML 2.0;
- ✍ Capítulo 5 – Teoria de Classes. Aborda a Teoria de Classes. Prefiro chamar de teoria de classes à teoria de orientação a objetos, pois abordo o uso, notação, semântica, jargão, visibilidade e modificadores com exemplos em Java, visando o diagrama de classes;
- ✍ Capítulo 6 – Diagrama de Classes. Aborda a construção do diagrama de classes, como criá-lo à luz dos Casos de Uso, armadilhas e soluções;
- ✍ Capítulo 7 – Tecnologias. Apresenta uma visão geral das tecnologias atuais, para que os analistas de negócios UML, possam conhecer a capacidade das tecnologias disponíveis, mesmo que não saibam programar para todas estas tecnologias. Saber o uso horizontalmente é mais importante para um analista de negócios do que saber verticalmente sobre dado assunto. Ao participar de reuniões e conversas sobre importantes decisões de software o analista deve estar preparado para esses aspectos;
- ✍ Capítulo 8 – Diagrama de Seqüência. Aborda o diagrama de seqüência, seu início e o porquê de usá-lo à luz dos cenários de um Caso de Uso; adicionalmente as últimas modificações introduzidas neste diagrama pela UML 2.0;
- ✍ Capítulo 9 – Diagrama de Estados. Aborda sua importância, uso e construção;
- ✍ Capítulo 10 – Banco de Dados. Esse capítulo aborda o Modelo de Entidade e Relacionamentos, informando sobre Entidades, dado, informação, construção do modelo MER, classes *versus* tabelas no banco de dados e situa o MER dentro do contexto dos diagramas da UML;
- ✍ Capítulo 11 – Diagrama de Pacotes. Aborda o Diagrama de Pacotes, que pode ser usado estereotipado com Casos de Uso e sem estereótipos, com classes;
- ✍ Capítulo 12 – Diagrama de Componentes e Implantação. Aborda o Diagrama de Componentes e Implantação, seu uso, sua notação e sua semântica, mas principalmente o momento de seu uso;

4 Desenvolvendo software com UML 2.0 – Definitivo

- ✍ Capítulo 13 – Diagrama de Objetos. Aborda o Diagrama de Objetos, sua notação, sua semântica e seu uso;
- ✍ Capítulo 14 – Diagrama de Comunicação, antigo diagrama de Colaboração. Aborda seu uso, sua notação e sua semântica;
- ✍ Capítulo 15 – Timing Diagram. Apresenta possíveis usos, semântica e notação;
- ✍ Capítulo 16 – Diagrama de Interação – Visão Geral. Aborda usos, notação, semântica e quais diagramas podem fazer parte deste novo diagrama da UML 2.0.
- ✍ Capítulo 17 – Composite Structure Diagram, novo na UML 2.0 Aborda possíveis usos, notação e semântica.
- ✍ Capítulo 18 – Métrica e casos de uso. Aborda o uso de métrica de estimativa de esforço na construção de software, usando Pontos de Casos de Uso;
- ✍ Apêndice A – Relata alguns dos estereótipos padrões, predefinidos na UML 2.0 e aqueles que deixaram de ser suportados.
- ✍ Finalmente existe uma seção onde você encontra as respostas para todos os exercícios dos capítulos. E mais: No site Web http://wpslive.pearsoncmg.com/br_medeiros_uml_1 os professores encontrarão mais de cem exercícios destinados ao uso em sala de aula; todos com respostas comentadas pelo autor.

Alguns poderão perguntar por que utilizei Java, quando precisei dar exemplos de código. A resposta é que este livro não é de programação, porém, pode ser necessário a alguém, com boa visão de programação, saber como a notação da orientação a objetos se traduz para código. Havendo essa necessidade você pode observar os exemplos dados. Os exemplos estão em Java porque é a linguagem que conheço.

Vamos imaginar um site de locação de DVDs pela Internet.

Nosso curso não exige ferramentas, porque existem mais de 300 ferramentas visuais para se modelar UML e precisaríamos ensinar o uso de uma ferramenta junto com os conceitos de UML, o que inviabilizaria este texto. Você pode estar em uma empresa que exige *skill* em uma ferramenta ou outra, como saber? Assim, focamos nos conceitos da UML e não na ferramenta.

Ajuda muito saber que as ferramentas têm um aspecto visual muito semelhante, que um diagrama de classes ou de casos de uso, em uma é muito semelhante ao apresentado em outra.

Para saber o que uma ferramenta de modelagem UML deve ter, antes de escolher, vá até o link Internet:

http://www.objectsbydesign.com/tools/modeling_tools.html.

No link http://www.objectsbydesign.com/tools/umltools_byCompany.html, você vai encontrar uma lista com muitas ferramentas de UML, com a descrição dos fabricantes, nome do produto, versão, data do último release, plataforma e preço.

5 Prefácio

Existem ferramentas gratuitas, pagas com preços razoáveis e pagas com preços altos, as diferenças ficam por conta do que a ferramenta oferece, além permitir a geração de gráficos e geração de códigos, porque todas essas características as boas ferramentas têm.

Ao escolher uma ferramenta, procure uma que atenda a usuários de vários sistemas operacionais, garanto que isso vai evitar uma série de problemas.

Devido ao fato de atuarmos em consultoria e treinamento de soluções orientadas a objetos, sempre nos perguntam como entrar em contato para trocar idéias, tirar dúvidas e solicitar treinamentos, entre outras coisas. Vá até o link da Fábrica Web ou me envie um e-mail, que terei um imenso prazer em atendê-lo.

Ernani Medeiros

www.fabricaweb.com.br

esales@fabricaweb.com.br

Desenvolvendo software com UML 2.0 : definitivo – Ernani Sales de Medeiros – São Paulo – Pearson Makron Books, 2004. – ISBN 85-346-1529-2