**Anexo 2**

**Unifying Modeling Language (UML)**

O desenvolvimento de software é um processo que envolve planejamento e, também, a administração do processo de desenvolvimento. Usualmente esse processo tem uma trajetória abrangente devido às várias fases pelas quais passa desde o seu início, que acontece na fase de levantamento de requisitos do sistema, até a sua fase final, que envolve implantação e testes.

Com o objetivo de promover a qualidade do software a ser desenvolvido e evitar problemas durante o processo de desenvolvimento é mandatório que a modelagem das especificações do software considerado seja realizada sempre acompanhada de uma documentação atualizada.

O uso da linguagem gráfica UML (*Unified Modeling Language*) [Booch *et al.* 1996] vem ao encontro da concretização dos cuidados a serem tomados para que problemas possam ser evitados durante o processo de desenvolvimento de sistemas computacionais orientados a objetos. UML é uma linguagem pesadamente gráfica que dispõe de um conjunto razoavelmente grande de elementos gráficos com diferentes caracterizações semânticas, que usados para representar variados componentes de um sistema computacional.

A representação gráfica de um sistema por meio de UML desempenha um importante papel, auxiliando desenvolvedores a visualizarem as inúmeras perspectivas funcionais das partes que compõem o sistema computacional sendo desenvolvido e, consequentemente, colaboram na promoção de uma compreensão mais apurada, por parte de desenvolvedores, de seu funcionamento. A linguagem gráfica inclui elementos tais como:

* Atividades (Jobs)
* Componentes individuais do sistema e como podem interagir com outros componentes de software
* Como o sistema será executado
* Como entidades interagem com outras(componentes e interfaces)
* Interface de usuário externa

Embora originalmente a linguagem UML tenha sido criada para documentar o projeto de sistemas computacionais orientados a objetos, em várias situações a linguagem tem sido estendida para abranger um conjunto maior de documentação de outros tipos de projetos.

## A2.1 Modelagem

Modelos e Diagramas são os conceitos fundamentais empregados no desenvolvimento da UML, que foi criada com o propósito de representar e documentar o desenvolvimento de aplicações computacionais. Em um ambiente de desenvolvimento de software, modelos colaboram para disponibilizar uma visualização abstrata do sistema em desenvolvimento e diagramas colaboram na representação concreta do sistema.

A UML é bem versátil uma vez que possibilita a modelagem de sistemas computacionais em diferentes níveis de detalhe. A modelagem de um projeto de software pode ser organizada e realizada por meio do uso de diagramas, modelos e de perfis. Diagramas de modelagem podem ser usados para detectar casos de uso de um sistema em modelos de caso de uso.

O uso de modelos pode ser conveniente quando (a) existe interesse em representar visualmente uma proposta de sistema a ser desenvolvido, (b) disponibilizar a representação de um sistema para discussão entre pares e/ou apresentação ao cliente, (c) desenvolver, programar e testar um sistema em construção e (d) usar diagramas UML para a criação de código.

## A2.2 Diagrama

É importante discernir entre o modelo UML e o conjunto de diagramas disponibilizado pela linguagem. Um diagrama é a representação parcial do modelo do sistema. O conjunto de diagramas não necessita refletir completamente o modelo e a remoção de um diagrama não muda o modelo. O modelo pode também conter documentação que direciona os elementos do modelo e os diagramas (tal como escrito em use cases).

A ULM 2 oferece muitos tipos de diagramas que são apresentados divididos em duas categorias. Alguns tipos representam informação estrutural e o resto representa tipos gerais de comportamento, incluindo alguns poucos que representam aspectos diferentes de interações. Esses diagramas são categorizados e organizados hierarquicamente como apresentado na sequência e mostrados nas Figura A2.1, Figura A2.2 e Figura A2.3.

1.View Estrutural (ou Estática) que enfatiza a estrutura estática do sistema usando objetos, atributos, operações e relações. Inclui diagramas de classe e diagramas de estrutura composta.

1.1 Classe: caracteriza um conjunto de informações, comportamentos e relacionamentos que descrevem uma entidade

1.2 Interface: caracteriza a parte visível de uma classe. Normalmente utilizada para descrever as assinaturas públicas de operações em uma classe.

1.3 Componente: uma unidade física de software que pode residir na memória de um processador que implementa um conjunto de interfaces.

2.View Comportamental (ou Dinâmica) que enfatiza o comportamento dinâmico do sistema, evidenciados colaborações entre objetos e mudanças nos estados internos de objetos. Inclui: diagramas de sequencia, diagramas de atividade e diagramas de estado da máquina.

2.1Interações: comunicações entre objetos.

Diagramas

Comportamental

Estrutural

**Figura A2.1** Diagramas são divididos em duas categorias: Comportamental (ver Figura A2.2) e Estrutural (ver Figura A2.3).

Interação

Comportamental

State Machine

Atividade

Caso de Uso

Comunicação

Interação Overview

Sequencia

Timing

**Figura A2.2** Diagramas da categoria Comportamental organizados hierarquicamente.

Componente

Estrutural

Objeto

Classe

Perfil

Estrutura Composta

Deployment

Package

**Figura A2.3** Diagramas da categoria Estrutural.