MODELAGEM ORIENTADA

PARADIGMA ORIENTADO A OBJETOS - UNIDADE 01

O paradigma orientado a objetos é um paradigma importante que apresentou os conceitos que em seguida viriam a ser englobados por diversas linguagens de programação. Estes conceitos são as classes, os objetos e as relações entre eles. Em oposição ao paradigma de programação estruturada composto por variáveis, funções e tipos abstratos para tratamento de dados, esse paradigma propôs um modelo novo de programação que permite a modelagem dos problemas a serem solucionados, pensando em objetos reais. Segundo sua definição, uma classe é uma abstração que descreve entidades do mundo real e quando instanciadas dão origem a objetos com características parecidas. Essa abstração é composta por atributos e pelos métodos.

MODELAGEM DE SISTEMAS

O paradigma orientado a objetos impacta diretamente na análise de requisitos. Essa fase é importante para que seja possível modelar e projetar corretamente o sistema. Este processo atualmente é chamado de modelagem e consiste no processo de desenvolvimento de modelos abstratos do sistema, sendo que cada um destes modelos apresenta uma perspectiva diferente do mesmo. O desenvolvimento de software apresenta diversos desafios e um dos principais é se certificar de que todos os requisitos solicitados pelo cliente serão atendidos de forma correta. Uma das formas de se obter sucesso com relação a este problema é fazendo uma boa análise dos requisitos para depois, elaborar modelos que o representem de forma simples.

LINGUAGEM DE MODELAGEM UNIFICADA

Ao fazer parte de uma equipe de desenvolvimento de software é necessário resolver diversos problemas para atender todos os requisitos requeridos. Antes de programar, modelamos a solução de alguma forma como por meio de um pseudocódigo. Porém, em uma empresa de desenvolvimento não será possível atender aos requisitos de forma satisfatória se nenhuma ferramenta for usada, tendo em vista que os problemas são consideravelmente maiores. Seria então interessante poder usar uma forma de modelar os problemas que todas as pessoas envolvidas em um projeto pudessem entender sem precisar de explicações, esse é exatamente o objetivo da UML que consiste em uma das principais ferramentas de modelagem usadas em empresas de desenvolvimento de sistemas. O processo de análise orientada a objetos também está relacionado à elaboração de modelos onde, os sistemas devem ser representados com elementos que sugerem o que acontece mais concretamente no mundo real.

SURGIMENTO DA LINGUAGEM DE MODELAGEM UNIFICADA

Diversos métodos voltados para orientação a objetos, mas com problemas de padronização foram desenvolvidos até que, o grupo de padronização chamado OMG percebeu o problema e resolveu apontar uma solução. Um chamado para criação de um padrão unificado de modelagem foi aberto e isto casou com o que estava sendo feito por três pesquisadores da área que haviam apresentado, um ano antes, propostas em relação à padronização. O resultado deste esforço foi a primeira versão da linguagem UML enviada para a OMG. Depois de um certo tempo, algumas outras versões foram lançadas em um processo de modernização, mas apesar de todas essas mudanças, até hoje a linguagem mantém suas características de criação.

MODELOS

O processo modelagem usando a UML é baseado em modelos, portanto é necessário saber em que consiste um modelo. Um modelo pode ser caracterizado, de forma simples e direta, como uma representação de algo de alguma natureza usando algo da mesma ou de outra natureza.

Modelos capturam aspectos importantes e omite o restante das informações. A forma como o modelo é apresentado deve ser selecionada para tornar mais simples tanto sua construção quanto seu entendimento. Em um nível alto de abstração, o objetivo é apresentar diagramas que são construídos no início do desenvolvimento para mostrar aos clientes as alternativas e ajudar na tomada de decisão em pontos que precisam de aprovação. Depois é necessário detalhar um pouco mais essa abstração da estrutura do sistema, apresentando sem muitos detalhes como serão desenvolvidos os objetos, as classes e os relacionamentos. Em seguida, o nível de abstração desce ainda mais com os diagramas e modelos completos para cada módulo do sistema. Com esse nível de abstração, tudo mais que seja necessário para que realmente seja possível desenvolver o sistema por completo, é realizado.

FLUXO DE DESENVOLVIMENTO

Existem diversos fluxos de desenvolvimento, que vão desde o modelo em cascata até as atuais metodologias ágeis, que possuem uma etapa de modelagem prevista. A UML pode ser usada em qualquer um, pois é independente do fluxo de desenvolvimento escolhido. Apesar de todas as vantagens dessa linguagem, algumas questões são usadas como desculpa para o não uso da ferramenta, como a falsa impressão de que ela é complexa. Isso pode se dar pelo pouco conhecimento sobre suas características ou pelo costume do uso algum outro um método pelo desenvolvedor, que aponta ser esse o método mais eficiente. Em relação a esses problemas é importante considerar que a UML é uma ferramenta completa, amplamente conhecida e que foi desenvolvida por um grupo de pessoas, que são especialistas na área de desenvolvimento, para ser simples e eficiente. Sendo assim é necessário que se aprenda essa ferramenta e que seu uso aconteça em diversos projetos antes de se optar por outra ferramenta ou nenhuma.

DIAGRAMAS

Durante o desenvolvimento de diagramas para modelagem de sistemas, questões relacionadas à estrutura e o comportamento, foram levantadas. Dessa forma, se fez necessário segmentar os diagramas UML em estruturais e comportamentais. Os diagramas que apresentam alguma estrutura são classificados como estruturais e os que são focados no comportamento das estruturas do sistema são catalogados como comportamentais. Um diagrama será estático se não apresentar informações que sofram mudanças ao longo do tempo, como o comportamento durante a execução. Os diagramas dinâmicos apresentam informações sobre o comportamento em tempo de execução e mudam à medida que o sistema se desenvolve.

DIAGRAMAS ESTRUTURAIS

CLASSES (CLASES)
PACOTES (SEGMENTOS DE DIAGRAMAS)
COMPONENTES (COMPONENTES USADDOS)
ESTRUTURA COMPOSTA (ELEMENTOS QUE INTERAGEM)
OBJETOS (OBJETOS E SEUS RELACIONAMENTOS)

DIAGRAMAS COMPORTAMENTAIS

CASOS DE USO (FUNCIONALIDADES DO SISTEMA)
ATIVIDADES (FLUXOS QUE OCORREM NO SISTEMA)
GERAL (RELACIONAMENTOS ENTRE OS DIAGRAMAS)
ESTADOS (CICLO DE VIDA DE DETERMINADO OBJETO)
SEQUENCIAL (TROCA DE MENSAGENS ENTRE OBJETOS)
TEMPO (DESEMPENHO DO SISTEMA COMO UM TODO)

MECANISMOS COMUNS ENTRE OS DIAGRAMAS

Associada a cada elemento existe algo que o descreve. Basicamente, a parte visual demonstra um entendimento sobre cada parte do sistema, enquanto essa outra detalha o mesmo. Cada diagrama começa com um símbolo e depois os adornos são adicionados. Um adorno consiste em um elemento que tem uma arte única e torna a representação daquele elemento, mais concreta. Os blocos de construção dos diagramas apresentam, em quase todos os casos, uma dicotomia entre a interface e sua implementação. Os mecanismos de extensão foram criados na para permitir que sejam feitas alterações sem a necessidade de mudar toda a linguagem.

PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

Apesar de todas as vantagens que conhecemos da UML, é necessário compreender como fazer seu uso de modo a obter os resultados esperados. Ela não aponta um processo de sistema padrão, mas usar apenas os modelos não é suficiente para abranger todas as necessidades do desenvolvimento como um todo. Dessa forma, é preciso fazer seu uso em conjunto com algum processo voltado para o desenvolvimento de sistemas.

PROCESSO UNIFICADO

Processos apontam uma sequência a ser seguida durante o processo de desenvolvimento do sistema. Esse modelo consiste em um processo de desenvolvimento interativo e incremental em que, a partir de um conjunto de atividades, os requisitos de clientes são convertidos em um sistema. O fato de ser interativo e incremental está associado ao fato de que as fases ocorrem simultaneamente, mas de forma escalonada e não sequencial, como o clássico modelo cascata.

MODELAGEM DE CASOS DE USO - UNIDADE 02

Normalmente o diagrama de casos de uso é construído após o levantamento dos requisitos dos usuários, pois nesse momento os desenvolvedores possuem todas as informações necessárias para sua elaboração. Quando o sistema apresenta um cenário conhecido e estabelecido, é possível, em uma primeira versão do diagrama, que se construa uma proposta de um sistema mais funcional, ainda que os requisitos do cliente não cubram todas as partes necessárias. Isso é importante, pois o dono às vezes não tem ideia de suas características e o diagrama de casos de uso completo pode ser apresentado a ele. Se faz possível dizer que esse diagrama captura a função e os requisitos do sistema usando atores, casos de uso e os relacionamentos entre eles.

CASOS DE USO

Os casos de uso representam uma função, componente, pacote ou classe de um sistema. Usado também demonstrar como o usuário manipulará o sistema. Segundo as regras da UML, essa estrutura deve, obrigatoriamente, ser nomeada, porém nenhuma norma sobre como isso deve ser realizado é apresentada, deixando a tarefa a cargo do desenvolvedor.

ATORES

Chamados de atores por representarem um papel externo e por interagir com o software. Um usuário seria o melhor exemplo de ator, mas pode ser qualquer outro elemento que acione uma interação com o caso de uso. Um ator pode ser associado a vários casos de uso, mas se deve ter pelo menos um ator associado a cada caso de uso.

RELACIONAMENTOS

Expressam a categoria de ralação existente entre os componentes. Uma associação aponta a comunicação entre um ator e um caso de uso. Uma generalização aponta que um caso de uso ganha as propriedades e o comportamento de outro, mudando seu comportamento.

EXTEND

Comportamento opcional, executado apenas em determinados casos. Especifica como e quando o comportamento apontado no caso de uso extensivo pode ser inserido no comportamento do caso de uso estendido. Vale ressaltar que não existe a criação de dependência.

INCLUDE

Representam comportamentos que são comuns para outros casos de uso. São segmentados em casos de uso menores que compartilham funções. Outro exemplo de uso é quando se deseja decompor um caso de uso complexo, em casos de usos menores e mais simples. Essa relação aponta obrigatoriedade onde, a execução de um primeiro obriga a execução de um segundo caso de uso associado. O comportamento do caso de uso que será incluído deve ser inserido no comportamento do caso de uso que começa.

MODELAGEM DE CLASSES

Quando se programa de forma estruturada, por meio da construção de funções, existe uma separação entre os dados e seu processamento. Com o desenvolvimento orientado a objetos, os dados e seu respectivo processamento, são integrados. Um diagrama de classes consiste em uma representação que tem como objetivo mostrar graficamente a estrutura de um sistema.

ELEMENTOS DO DIAGRAMA DE CLASSES

Um diagrama de classes apresenta três informações principais, o nome da classe, os atributos e os métodos, mostrados respectivamente na primeira e na segunda partição. Esses diagramas apresentam classes comuns e as classes abstratas. Classes comuns possuem nome, atributos e métodos, separados por um segmento. Classes abstratas são igualmente apresentadas, porém com o nome da classe em destaque. Outro componente presente nos diagramas de classe são as interfaces que apresentam um conjunto de elementos, que devem ser implementados por outras classes. Como uma classe abstrata ou interface não instanciam objetos é necessário que alguma outra classe programe o que está descrito na interface ou que alguma classe herde os atributos e métodos da classe abstrata.

RELACIONAMENTOS

Uma relação de herança é uma das bases da orientação a objetos e aponta que uma classe é criada a partir de outra, apenas associando os elementos à sua composição. Assim, podemos afirmar que quando uma classe herda algo de outra, estamos apontando que a nova classe terá tudo que a anterior tinha em adição as características próprias dessa nova classe. Isso remove a necessidade de correção em diversas partes de um código tornando o desenvolvimento um processo mais simples. Uma relação de agregação representa o fato de que uma das classes do relacionamento é parte ou está contida em outra. Nele, a parte existe independentemente do todo, formando um relacionamento fraco. Uma relação de composição apresenta um vínculo mais forte entre as associações, onde o todo não existe sem as partes e as partes não existem sem o todo. Outro importante relacionamento é o encapsulamento, que tem como objetivo garantir a proteção e controle de acesso dos atributos e métodos de uma classe, estabelecendo diferentes níveis de acesso, dependendo de quem acessa.

DIAGRAMA DE OBJETOS

Além do diagrama de classes, temos o diagrama de objetos, que é usado para representar os objetos instanciados de uma classe. Se faz importante ressaltar que um diagrama de objetos deve ser gerado com base em um diagrama de classes e tem como objetivo representar os objetos e seus relacionamentos, sendo considerado um grafo de instâncias. As representações de objetos são feitas de forma parecida com as de classes, porém eles só possuem atributos, e seu nome aponta a classe da qual foram instanciados.

MODELAGEM DE ATIVIDADES

Um diagrama de atividades tem um papel fundamental por mostrar aspectos comportamentais na modelagem de sistemas. Basicamente, ele consiste em um gráfico na forma de fluxo que mostra as características dinâmicas do sistema onde, o fluxo de ações que serão realizadas em determinada atividade relacionada ao sistema que será desenvolvido, pode ser visto. Esta perspectiva sobre o sistema é importante ser considerada, pois, um produto de software pode apresentar melhorias em seu fluxo de ações das atividades. Além disso, a sequência das atividades pode não ter sido implementada de forma otimizada, podendo ter uma melhor performance se realizada de outra forma. Podemos entender o diagrama de atividades como casos especiais do diagrama de casos de uso. Uma vez que uma atividade representa um possível caso de uso do sistema de forma detalhada, os diagramas estão relacionados de certa forma. Se faz possível elaborar diagramas de atividades com base nos casos de uso já descritos para o sistema. Este diagrama é orientado a fluxos de controle, tendo uma similaridade com os fluxogramas de programas. Todavia, essa semelhança está apenas na aparência, devido a uma maior quantidade de informações existentes.

ELEMENTOS DO DIAGRAMA DE ATIVIDADES

ESTADOS

Em um diagrama de atividades temos apenas um símbolo que aponta o começo e outro que aponta o encerramento do processo, porém o símbolo terminal pode ser usado diversas vezes no diagrama. O terminal aponta o encerramento de uma gama de determinados processos, mas não representa o termino de todos os fluxos da atividade. Apenas o símbolo de encerramento pode decretar que a atividade esta encerrada.

PROCEDIMENTOS

Os procedimentos internos são representados por um retângulo de bordas arredondadas, e as setas entre eles representam a mudança de uma atividade para a outra.

FLUXOS

Considere que existe uma condição a ser expressa no diagrama de atividades, que aponta para um teste. Caso seja verdadeiro, o fluxo executará uma ação, mas caso seja falso, outra ação será executada. As estruturas condicionais são expressas nos diagramas de atividades como losangos, os quais tanto podem ser usados para representar a divisão do fluxo devido a uma condição como para unir dois fluxos para resultarem em uma mesma atividade.

BARRAS

Podemos segmentar um fluxo, executando tarefas de forma concorrente. Nesse caso, usamos os componentes FORK e JOIN. O primeiro representa uma divisão dos fluxos de controle e o segundo mostra a união dos mesmos. As operações representadas por esses componentes são paralelas e ocorrem de forma concorrente. Assim, pouco importa se uma operação demorar mais tempo do que outra, o fluxo continuará, somente quando ambas terminarem.

MODELAGEM DE ESTADOS - UNIDADE 03

Apontar o comportamento dos objetos na perspectiva dos estados que eles assumem durante a execução das funções do sistema, é fundamental. Cada objeto no mundo computacional se encontra em algum estado particular, até que algum acontecimento ocasiona as mudanças de estado. Um objeto muda seu estado por causa de algum evento, provocando uma transição de estados e desencadeando determinadas ações. O chamado diagrama de estados representa o comportamento de um elemento por meio de um conjunto de estados e suas transições.

ELEMENTOS DO DIAGRAMA DE ESTADOS

INICIAL STATE (ESTADO DO OBJETO QUANDO CRIADO)
FINAL STATE (ENCERRAMENTO DO CICLO DE VIDA DO OBJETO)
STATE (ESTADO EM QUE UM OBJETO SE ENCONTRA)
TRANSITION (TROCA DE ESTADO DO OBJETO)
CHOICE (TOMADA DE ESCOLHAS DURANTE O PROECESSO)
FORK (MOMENTO EM QUE PROCESSOS PARALELOS OCORREM)
JOIN (MOMENTO EM PROCESSOS PARALELOS SE UNEM)
ENTRY POINT (INGRESSO DE ESTADO OU ESTADO COMPOSTO)
EXIT POINT (EGRESSO DE ESTADO OU ESTADO COMPOSTO)
JUNCTION (ENCADEAMENTO DE DIVERSAS PASSAGENS DE ESTADO)
TERMINADOR (ENCERRAMENTO DAS ATIVIDADES DO OBJETO)

ATIVIDADES INTERNAS

DO (ATIVIDADES FEITAS DURANTE UM ESTADO)
ENTRY (ATIVIDADES FEITAS NO INGRESSO DE UM ESTADO)
EXIT (ATIVIDADES FEITAS NO EGRESSO DE UM ESTADO)

MODELAGEM COMPORTAMENTAL

Usualmente, a modelagem comportamental do sistema, começa com o modelo de casos de uso que, descreve os requisitos funcionais do sistema e os atores. No entanto, esse modelo de não detalha o comportamento interno dessas funções. Para isso, a UML contempla os diagramas comportamentais, que mostram como os objetos do sistema agem internamente, favorecendo assim, o entendimento dos aspectos dinâmicos do sistema. Uma das formas mais usadas para se descrever a interação entre os objetos é a ênfase à ordenação temporal das mensagens.

ELEMENTOS DO DIAGRAMA SEQUENCIAL

Os atores enviam mensagens aos objetos, requestando a execução de alguma operação ou simplesmente o envio de dados. Os objetos podem existir desde o começo ou podem ser criados com o tempo. Os focos de controle representam o período em que um elemento executa uma ação. As mensagens síncronas são representadas por retas horizontais com uma seta na ponta, as quais um emissor aguarda o retorno antes de prosseguir. As mensagens assíncronas são representadas por retas horizontais com uma seta aberta em uma das extrerminadades, as quais os emissores continuam enviando sem aguardar o retorno, fazendo com que o receptor da mensagem não precise responder imediatamente. As mensagens construtoras representam o momento em que um determinado objeto é criado, sendo instanciado por uma mensagem que fora enviada anteriormente. Uma mensagem destrutora aponta a destruição do objeto que não se mostra mais necessário no processo. Uma mensagem de retorno é enviada entre objetos, em resposta à outra mensagem, após a execução de uma determinada ação.

TROCAS DE MENSAGENS

As mensagens representam a demanda que um elemento envia para o outro com o objetivo de executar uma determinada ação, demostrando a ocorrência de eventos. Quem faz o envio de uma mensagem é chamado de emissor, e quem recebe essa mensagem é chamado de receptor.

ESTEREÓTIPOS E FRAGMENTOS

Na representação dos objetos do diagrama sequencial, podemos adotar os estereótipos do diagrama de classes para melhor compreensão das interações entre os objetos. Um estereótipo denominado de classe de fronteira é aquele que representa a interface do sistema, indicando a comunicação entre um ator e os demais objetos que participam da interação. Um estereótipo denominado de classe de controle serve de intermediário entre as classes para tratar as regras de negócio e seu fluxo. Um estereótipo denominado de classe de entidade é aquele que mostra que as classes do sistema também são entidades. Também é possível usar fragmentos, que podem ser fragmentos de interação ou combinados. Cada fragmento representa uma interação independente, formando uma fronteira entre os elementos presentes no diagrama. Assim, é recomendável usar fragmentos apenas se colaborarem para a compreensão. Um fragmento de interação representa a ocorrência de outro diagrama. Um fragmento combinado é usado para determinar o fluxo, correspondendo a uma sequência de mensagens encapsuladas em um fragmento, que compõe um procedimento que pode ser reaproveitado em demais diagramas.

DIAGRAMA GERAL

Consiste em uma abordagem para modularizar a construção de diagramas de interação com o objetivo de dar uma visão geral dos diversos cenários de um caso de uso. Esse diagrama é uma variação do diagrama de atividades, demostrando uma visão de alto nível das interações de um sistema ou processo a partir da integração de vários diagramas sequenciais.

DIAGRAMA DE TEMPO

Demostra a mudança no estado de um objeto, observando o papel que ele assume durante algum tempo. Concentra sua atenção ao tempo exato em que os eventos acontecem.

MODELAGEM COM PROCESSO UNIFICADO - UNIDADE 04

Processos apontam uma sequência a ser seguida durante o processo de desenvolvimento do sistema. Esse modelo consiste em um processo de desenvolvimento interativo e incremental em que, a partir de um conjunto de atividades, os requisitos de clientes são convertidos em um sistema. O fato de ser interativo e incremental está associado ao fato de que as fases ocorrem concorrentemente, mas de forma escalonada e não sequencial, como o clássico modelo cascata.

FASES DO PROCESSO UNIFICADO

CONCEPÇÃO (ESCOPO E OS CASOS DE USO DO PROJETO)
ELABORAÇÃO (FORMA COMO O SISTEMA SERÁ CONSTRUÍDO)
CONSTRUÇÃO (COMPONENTES DO SISTEMA)
TRANSICIÇÃO (TESTES E ENTREGA O PRODUTO)

ATIVIDADE DE REQUISITOS

Nesse modelo, a atividade de requisitos é a primeira ação de cada fase do processo. Consiste em uma das tarefas mais complexas da engenharia de software, pois fundamenta e sustenta todo o restante do processo de desenvolvimento. Das técnicas de modelagem da UML, podemos adotar o diagrama de casos de uso para modelar requisitos funcionais, que guiarão o processo de desenvolvimento. O diagrama de atividades para representar o comportamento de cada requisito funcional. Ou o diagrama sequencial para apontar o cenário de cada função.

REQUISITOS FUNCIONAIS (COISAS QUE O SISTEMA DEVE FORNECER)
REQUISITOS NÃO FUNCIONAIS (QUALIDADES DO SISTEMA)

ATIVIDADE DE ANÁLISE E PROJETO

Consiste na segunda ação de cada fase do processo. O ato de analisar consiste em determinar o que o sistema deve fazer em uma visão lógica do negócio. O ato de projetar consiste em apontar como acontecerá o desenvolvimento do software, em consonância com as tecnologias de programação e banco de dados que serão adotados. Dessa forma, a atividade de análise precede a atividade de projeto, mas elas também acontecer concorrentemente.

DOCUMENTOS E MODELOS DE CASOS DE USO

Não existe um formato certo para a documentação dos casos de uso. Dessa forma, é possível que se documente da forma que se considerar melhor. Mas, a forma sugerida, é chamada de formato de cenários. Nele existe o cenário fundamental que relata a descrição de uma tarefa em um mundo ideal, e o cenário alternativo que relata qualquer situação que represente uma exceção ao cenário fundamental. No contexto da modelagem de sistemas, um modelo aponta para um conjunto de diagramas, sendo recomendado agrupar os casos de uso por assunto.

DIAGRAMA DE PACOTES

Demostra os elementos do sistema agrupados e organizados em pacotes, com o objetivo de representar os componentes ou módulos que integram um sistema e suas dependências. Um pacote é um mecanismo usado para agrupar elementos semanticamente relacionados. Esse diagrama tem uma notação gráfica simples, que demostra como os elementos do sistema estão organizados em pacotes e suas dependências, partindo da categorização dos elementos em grupos que representam as partes do sistema.

DIAGRAMA DE ESTRUTURA COMPOSTA

Detecta a forma de um conjunto de elementos que interagem, durante o processo de execução de um determinado sistema, de forma a associar os mesmos. Contudo, o comportamento dessa associação não é detalhado, pois isso é o objetivo de diagramas comportamentais. Representa a estrutura interna de um componente, com base na descrição das partes que o compõem e se comunicam. Tem uma notação gráfica simples, explanando exatamente os objetos das classes que assistem um caso de uso ou integram uma colaboração.