Diagrama de casos de uso

Este documento existe em versão eletronica, na página web :
`http://www.cos.ufrj.br/~anquetil/UML/`

Nessa aula vamos estudar o primeiro diagrama para modelar o comportamento de um sistema: os casos de uso. Vamos discutir várias noções incluindo: caso de uso, atores, extensão e inclusão de casos de uso, cenário, ...

Esta aula é baseada nos capítulos 16 e 17 do "UML User Guide" e o livro "Applying Use Cases – A Practical Guide" (c.f. referência no capítulo "Introdução" da disciplina).

Descrição geral

Este diagrama mostra como o sistema a ser desenvolvido vai interagir com seu ambiente (usuários ou outros sistemas). Ele é bastante importante porque vai ser a base do processo de desenvolvimento do sistema. O diagrama de classes especifica a estrutura do domínio e do sistema, os casos de usos vão ser a entrada para formalizar as funcionalidades que o sistema deve cumprir.

Um caso de uso descreve as operações que o sistema deve cumprir para cada usuário. Ele vai ajudar a formalizar as funções que o sistema precisa fazer. Vamos definir um caso de uso para cada tarefa que o sistema deve cumprir para um usuário.

Por exemplo, para o sistema de reservas, vamos ter um caso de uso para fazer uma reserva, consultar as reservas, suprimir uma reserva, imprimir relatórios, etc.

Um *caso de uso* se apresenta como uma lista completa das interações entre um usuário e o sistema para cumprir uma tarefa. Lista completa significa que o caso de uso descreve as interações desde o início da tarefa, até o fim.

Um exemplo de caso de uso é proposto na figura 10.

Casos de uso descrevem interações entre o sistema e *atores*. Um ator é "alguma coisa" (usuário, outro sistema, ...) que não faz parte do sistema e interage com ele. Por exemplo para um sistema de regulagem da temperatura, o termostato vai ser um ator, ele interage com o sistema e não faz parte dele.

Um usuário real pode cumprir vários papeis para o sistema (i.e. ser representado por vários atores), por exemplo num banco, o gerente pode ser um ator quando ele coloca dinheiro no caixa (ator: operador) e um outro ator quando ele tira dinheiro de sua própria conta (ator: cliente). Um ator pode também representar várias pessoas, quando falamos de um "cliente" tirando dinheiro do caixa eletrônico, é claro que esse ator representa qualquer pessoa.

É importante identificar todos os atores que vão interagir com o sistema. E para cada

Tirar dinheiro de um caixa eletrônico

1. O usuário introduz o cartão no caixa eletrônico
2. O caixa eletrônico propõe varias operações
3. O usuário aperta o botão "saque"
4. O usuário escolhe a conta (ex.: "conta corrente")
5. O usuário entra a valor do saque
6. O usuário entra sua senha
7. O caixa eletrônico verifica a senha com o banco e o saldo da conta
8. O caixa eletrônico da o dinheiro para o usuário
9. O caixa eletrônico imprime um recibo

Figura 10: Um exemplo de caso de uso

ator, é importante identificar que operações ele vai precisar. A pesquisa de caso de usos se faz a partir dos atores, não a partir das operações. Isso quer dizer que não vamos procurar qualquer funcionalidade abstrata que seria interessante para o sistema cumprir, mas vamos procurar funcionalidades concretas que um usuário realmente precisa.

Geralmente, um caso de uso é iniciado por um ator, não pelo sistema. Por exemplo, na figura 10, o primeiro passo é "O usuário introduz o cartão".

Um caso de uso só descreve as interações entre o ator e o sistema, não descreve como essas funcionalidades vão ser implementadas. Por exemplo na figura 10 não descrevemos como o sistema vai verificar a senha ou o saldo da conta. Isso será explicitado mais tarde, com outros diagramas.

Nota : No desenvolvimento de um sistema, é muito importante fazer a distinção entre o "que" e o "como".

Pensar no "como" cedo demais poderia influenciar o "que" de uma maneira útil para o programador mas não para o usuário. É importante no início do desenvolvimento tentar não pensar na implementação e só concentrar nos desejos dos usuários. Poderemos modificar esses desejos depois se tiver um problema com o projeto, mas desse modo, a modificação será explicita. O que significa que depois de alguns anos, saberemos porque tomamos esta decisão.

Nota : De maneira geral, no desenvolvimento de um software, é importante sempre gravar as razões de todas as decisões para poder reutilizá-las depois. A ausência de tal informação é um problema muito comum e importante na manutenção de sistemas.

Também é muito difícil gravar todas as decisões porque muitas delas são feitas de maneira inconsciente.

Casos de uso têm vários objetivos importantes:

- Ser compreensíveis para usuários que provavelmente não entendem informática.
- Incentivar a análise do sistema especificando as funcionalidades necessárias.
- Delimitar o sistema.
- Servir de base para os casos de teste.

Casos de uso têm que ser compreensíveis por usuários por que só eles sabem o que o sistema precisa fazer. Os casos de uso permitem verificar se o desenvolvedor e o usuário concordam sobre o que o sistema deve fazer. Isso é um problema importante no desenvolvimento de software. No mesmo tempo, casos de uso podem servir de "contratos" entre os usúarios e a equipe de desenvolvimento.

No "Unified Software Developement Process", os casos de uso vão, também, incentivar o desenvolvimento do sistema porque eles descrevem as funcionalidades necessárias. Tudo o que o sistema precisa fazer vai ser descrito em um dos casos de uso.

Já falamos que um ator é uma "coisa" que não faz parte do sistema, nesse sentido, quando identificarmos atores e o que eles devem fazer, nós estamos definindo os limites do sistema. Para especificar uma interação, devemos definir o que o sistema e os atores fazem, por exemplo quando entrar um número de identificação, quem vai verificar que o número é correto, o sistema ou o ator? Pode, também, ser difícil identificar atores quando eles foram outros sistemas (ou sub-sistema).

Não é sempre claro o que pertence ao sistema e o que é fora dele. Especificar o caso de uso nos obriga a tomar essa decisão explicitamente. (De novo, o fato da decisão ser explícita é muito importante).

Finalmente, os casos de uso podem também ser usados como base para criar casos de teste. Testar o sistema para verificar se ele faz o que precisa e sem bugs é uma tarefa fundamental do desenvolvimento de software. Mas ao mesmo tempo, é difícil estabelecer testes bons. Os casos de uso são muito úteis nesse sentido, já que um caso de uso descreve uma interação completa e real com o sistema.

Os casos de uso são descrições muito abstrata das funcionalidades necessárias. Quando vamos afinar mais a especificação do sistema, cada caso de uso poderá ser formalizado com diagramas de seqüência e/ou de colaboração. Esses diagramas são mais formais, e então mais úteis para especificar precisamente (sem ambigüidades) o funcionamento do sistema. Mas, ao mesmo tempo, esses diagramas podem ser mais difíceis de entender pelo usuário. Normalmente, tem vários desses diagramas para um só caso de uso.

O diagrama

O nome de um caso de uso pode ser qualquer sentencia, mas a UML recomenda usar uma frase ativa curta (verbo + substantivo), por exemplo: "entrar reserva", "tirar dinheiro", ...

Esse diagrama é diferente dos outros porque contem poucos gráficos. Um caso de uso é principalmente composto de texto livre, ou pseudo-código que descreve cada interação.

Os elementos principais do diagrama são uma elipse para representar um caso de uso e um pequeno boneco para representar um ator (figura 11). O nome do caso de uso pode ser dentro da elipse ou abaixo dele.

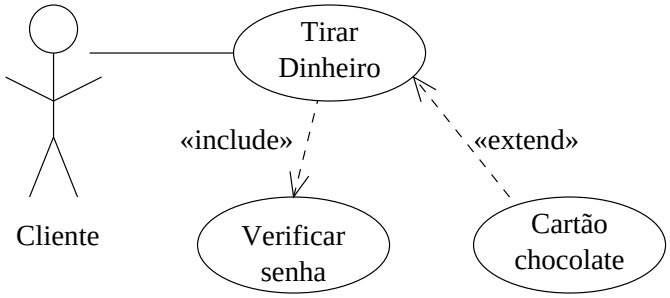

Figura 11: A representação do diagrama de caso de uso em UML

O diagrama não especifica os casos de uso, mas só apresenta qual ator interage com qual caso e organiza os casos entre eles (ex.: qual herda de um outro). Os casos de uso vão ser espificados em outros documentos. A UML não define precisamente a forma dessa descrição. Ela é um texto livre.

Casos de usos podem ser organizados em pacotes, ligados com a relação de generalização ou a relação de dependência. Tal organização de casos de uso será usada só para o desenvolvimento de sistemas amplos. Ela não é considerada importante nessa disciplina.

O livro "Applying Use Cases" (c.f. referências no capítulo "Introdução" da disciplina) propõe uma estrutura um pouco mais formal de especificação dos casos de uso (p.84 e anexo B). Vocês podem também procurar essa especificação na internet: http://books.txt.com.

Vamos estudar aqui uma versão simplificada dessa estrutura (figura 12). Notem que essa estrutura introduz várias noções que serão descritas na próxima seção.

Esta simplificação é ainda muito rica e não será usada inteiramente na maioria dos casos. Um caso mínimo vai ter um nome e o(s) fluxo(s) de eventos (principal e possivelmente alternativos). De maneira geral, a descrição abstrata ao inicio é uma boa idéia. Pontos de extensão e casos de uso incluídos (ver próxima seção) são opcionais sendo utilizando quando necessário.

Noções avançadas

As noções mais avançadas que vamos discutir nessa seção são: fluxo de eventos e cenário, pre- e pós-condições, interfaces, a relação de generalização no diagrama de caso de uso, inclusão e extensão de casos de uso, atores de um sub-sistema.

Fluxo de eventos, cenário

Um caso de uso descreve um fluxo de eventos. Normalmente, para cumprir uma operação, pode ter vários fluxos de eventos alternativos. Por exemplo, para o caso de uso "tirar

Nome do caso de uso
Descrição do caso de uso (um parágrafo).
Atores
Lista dos nomes dos atores com descrição curta.
Prioridade
Seja este caso de uso muito importante no projeto ou acessório?
Estado
Qual é o estado desse caso de uso (ainda muito abstrato, bem especificado, fechado, ...) ?
Pre-Condições
Lista de condições que têm que ser verificadas antes que o caso de uso começa.
Fluxo de eventos
Fluxo principal
1. Primeiro passo no caso de uso.
2. Segundo passo no caso de uso.
3. ...
Fluxos alternativos
Descrever os fluxos alternativos.
Pós-Condições
Lista de condições que têm que ser verificadas depois do fim do caso de uso.
Pontos de extensão
Lista dos pontos de extensão no caso de uso.
Casos de uso incluídos
Lista dos nomes dos casos de usos incluídos.
Outros requisitos
Lista de outros requisitos que afetam o caso de uso.

Figura 12: Especificação de um caso de uso

dinheiro", tem o fluxo de eventos principal descrito na figura 10. Mas podem existir também outros fluxos que se referem ao usuário errar na entrada da sua senha, querer cancelar a operação, ou não ter dinheiro suficiente na sua conta, etc.

Cada fluxo de eventos, cada caminho de interação desde o início da tarefa até o fim é chamado de *cenário*.

A maioria das operações são amplas demais para nós descrevermos todos os cenários simultaneamente. Para simplificar o trabalho, vamos procurar definir independentemente cada cenário. Isto deve também nos ajudar achar todos os cenário possíveis.

Vamos sempre definir primeiro o "fluxo principal" de eventos que é o cenário "normal", sem erros, nem cancelamento, etc. É o cenário que descrevemos no caso de uso exemplo da figura 10. Em geral, esse caso será aquele que um usuário vai descrever em primeiro lugar também. É o caso básico a partir do qual poderemos acrescentar variações, nuanças e casos "anormais" (erros, cancelamentos, etc.)

Todas essas variações vão ser descritas depois nos fluxos alternativos. Tem vários tipos de cenários alternativos:

- Caminhos alternativos (menos freqüentes) para chegar ao mesmo ponto (cliente pode tirar dinheiro da poupança em vez da conta corrente).
- Caso de erro (por exemplo a senha do cliente é errada).
- Caso de cancelamento (o cliente desistiu de tirar dinheiro).

Pré-condições, pós-condições

A descrição do caso de uso pode conter condições que tem que ser verificadas (verdadeiras) antes o caso ser executado (pré-condições) ou depois que for executado (pós-condições).

As pré-condições especificam qual é o estado do sistema antes do caso começar. As pós-condições indicam em qual estado o caso de uso vai deixar o sistema. Essas condições tem que ser verdadeiras independentemente do cenário que for executado (incluindo um possível cancelamaneto).

Por exemplo uma pre-condição para nosso caso de uso seria que o caixa eletrônico estivesse ligado e não seja usado por outra pessoa nesse momento. A pós-condição, nesse caso, poderia ser a mesma. Depois do cliente tirar dinheiro, a maquina tem que estiver de novo pronta para atender um novo cliente (ou o mesmo para uma outra operação). Não importa qual cenário o cliente executou exatamente —se ele tirou dinheiro, se ele cancelou ou se ele não tinha dinheiro suficiente na conta— a pós-condição tem que ser verificada.

Pré-condições não podem depender de informações que vão ser entradas no caso de uso mesmo. Por exemplo, no caso Tirar dinheiro, uma pré-condição não pode ser que o cliente tiver conta nesse banco, porque antes do caso (é uma PRÉ-condição), não sabemos ainda qual é o banco dele, ele ainda não introduziu o cartão.

Interfaces

O livro "Applying Use Cases" propõe de associar interfaces aos casos de uso e atores. Como já vimos, as interfaces vão especificar quais operações o ator (ou o caso de uso) realizam. No caso de um ator usúario, é claro que as operações são virtuais e só servem para descrever as ações que ele pode fazer. Essas interfaces vão permitir especificar um pouco mais as interações. Interfaces podem também ser úteis para reutilização de casos de uso (ou parte de casos).

Com nosso caso de uso, a interface do ator cliente vai conter as operações "entra operação escolhida", "entra valor do saque" e "entra senha".

A representação das interfaces é indicada na figura 13. A linha simples indica quem realiza a interface, a seta tracejada indica quem usa a interface.

Generalização

O livro "Applying Use Cases" diz que existe generalização só entre atores, não entre caso de usos. O "UML User Guide" diz que existe generalização entre caso de usos. Essa segunda

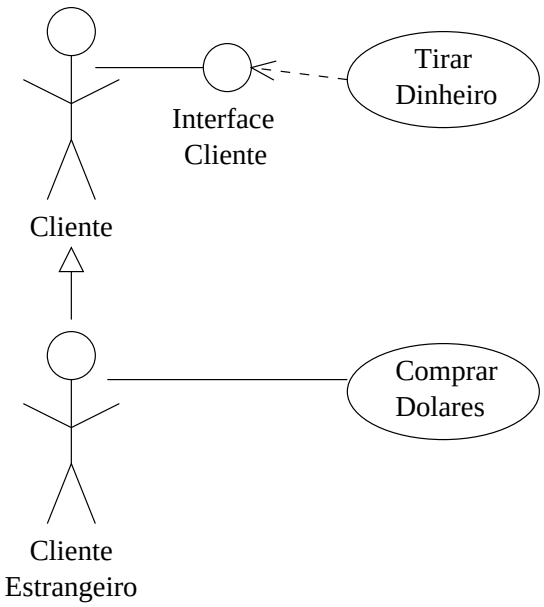

Figura 13: Interfaces e generalização para atores e casos de uso.

interpretação parece a melhor interpretação.

Entre atores, a generalização significa que o ator mais especifico pode ser usado em vez de o ator mais geral. O ator que herda interage com os mesmos casos de uso que o ator mais geral, e pode também interagir com outros casos de uso que o mais geral não tem.

Entre casos de uso, a generalização significa a mesma coisa: o caso de uso mais especifico pode ser usado em vez de o mais geral. Por exemplo, o caso de uso "Verificar senha" pode ser usado em vez de um caso de uso mais geral "Identificar usuário". Outros casos de uso específicos podem também ser usados em vez de "Identificar usuário".

Essas relações de generalização talvez sejam mais fácil de entender quando pensarmos nas interfaces: um ator ou um caso de uso "realizam" uma interface e o "sub-ator" ou o "sub-caso" têm que realizar a mesma interface.

Na figura 13, podemos ver uma especialização do ator Cliente que pode efetuar o caso de uso "tirar dinheiro" mas também um outro caso especifico a esse tipo de Cliente.

Inclusão

A UML propõe também duas relações de dependência entre casos de uso: ≪estende≫ e ≪inclui≫.

A dependência ≪inclui≫ é usada para decompor um caso de uso complexo em sub-partes. O caso de uso complexo inclui um outro que cumpre uma sub-parte dele (exemplo). Esse mecanismo permite não repetir a mesma sub-parte várias vezes. Nesse caso, os dois casos de uso (o complexo e a sub-parte) não podem ser encontrados só porque o caso complexo não é completo sem a sub-parte, e a sub-parte não é um caso completo e não significa nada fora do contexto do caso complexo.

Extensão

A dependência ≪estende≫ indica uma extensão possível de um caso de uso básico. No caso de uso básico, tem um ponto de extensão onde se pode usar o caso de uso extensão. Isso quer dizer que nos casos normais, apenas o caso de uso básico será usado, mas em alguns casos, ele será estendido pelo caso de uso extensão que vai acrescentar alguns interações. Ao contrário da inclusão, o caso de uso que é extendido não sabe da existência do caso de uso extensão.

Por exemplo, o caso de uso descrito na figura 10 poderia conter um "ponto de extensão" antes do ponto 8 para os clientes com cartão "chocolate" que indique que o cliente deve deixar pelo menos 500$Reais na conta. O caso de uso extensão é proposto na figura 14.

Caso de uso extensão "Cartão chocolate"
Se o cartão for um cartão chocolate, então extensão "cliente especial":

1. Caixa verifica com o banco que o saldo da conta será superior a 500$Reais depois do saque

Figura 14: Um exemplo de caso de uso

O caso de uso estendido (figura 10) deve especificar quais são seus pontos de extensão, por exemplo assim: Ponto de extensão "Cliente especial" antes do passo 8.

Num diagrama, os pontos de extensão podem ser especificados também: a elipse do caso de uso é separada em dois, na parte superior se coloca o nome do caso de uso, na parte inferior se coloca os pontos de extensão.

O caso extensão deve conter uma condição ao inicio. Ele é "executado" somente se a condição for verdade. Esse caso de uso tem que especificar também qual ponto de extensão ele usa, porque um caso de uso pode ter mais de um.

Um ponto de extensão único pode ser usado por vários casos extensão.

Sub-sistema

Em muitos casos, o sistema é amplo demais para ser entendido inteiramente. Por isso, decompomos tais sistemas em sub-sistemas de tamanho menor que podemos entender completamente até um nível suficiente de detalhes para implementá-los.

Um exemplo possível de decomposição em sub-sistemas seria de ter um sub-sistema gerando uma base de dados, um gerando as interações com os usúarios (telas, impressões, etc.), e um intermédio fazendo computações. Esses sub-sistemas vão ter que comunicar entre eles, por exemplo o sub-sistema de computação pede dados ao sub-sistema do banco de dados, faz algumas computações sobre esses dados e passa os resultados ao terceiro sub-sistema para imprimir.

Para cada sub-sistema, os outros sub-sistemas são atores com quem ele vai interagir. A noção de interface nesse caso é mais clara, ela corresponde à interface do sub-sistema: quais operações ele exporta para outros sub-sistema.

"Truques"

Procurem primeiro os atores que interagem com o sistema. Para isso, procurem responder perguntas como: Quem usa o sistema? Quem o instala? Quem o inicia ou o pára? Quem faz a manutenção? Quem entra dados no sistema? E quem precisa/usa dados do sistema?

É bom definir um pequeno dicionário dos atores, que descreve sucintamente cada ator. Isso permite verificar que temos uma idéia clara de o que o ator representa e permite também verificar com os usúarios as definições dos atores.

Depois, para cada ator, procurem definir quais operações ele precisa fazer. Uma interação é uma troca de dados: Quem fornece dados para quem (o sistema para o ator ou o ator para o sistema ou os dois)? Quem cria, consulta, modifica ou destrui os dados?

Sempre definam em primeiro o fluxo de eventos principal de um caso de uso sem se preocupar com os fluxos alternativos.

Exercícios

Especifiquem um caso de uso completo para um sistema de telefone quando:

- O ramal XXX tenta ligar o ramal YYY.
- Mesmo, mas o ramal YYY rederecionou os telefonemas para um outro ramal (ZZZ).

Quais são os casos de uso para o sistema de reserva da nossa empresa aérea ? Especifiquem cada um deles.