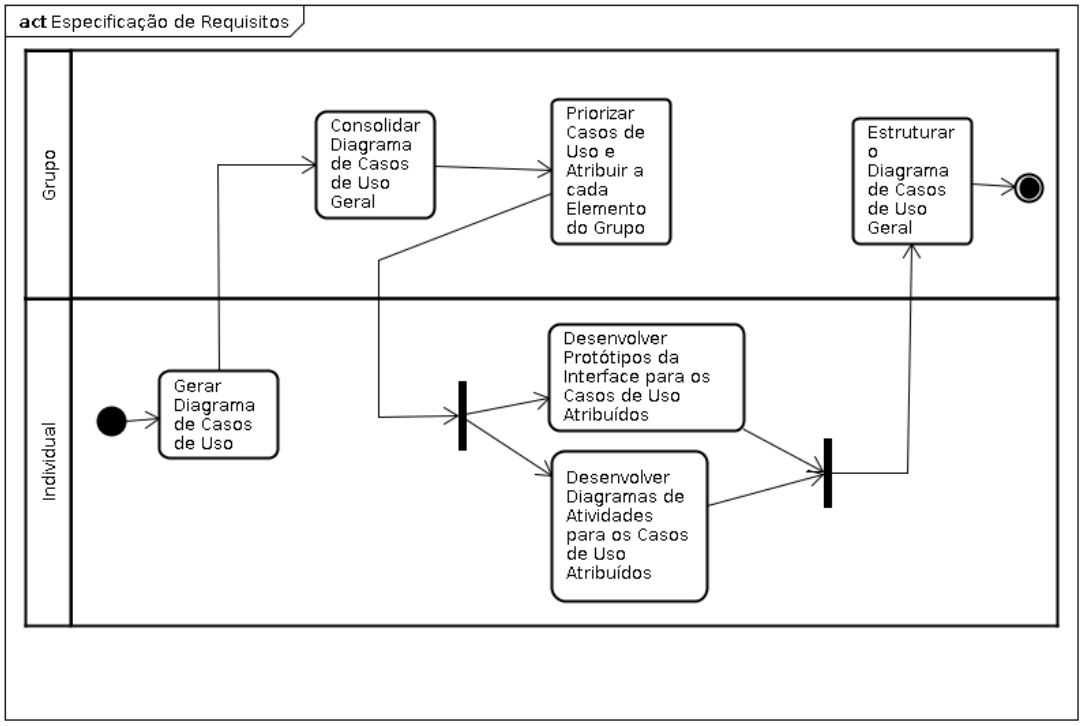
Especificação de Requisitos

Ricardo R. Gudwin
DCA-FEEC-UNICAMP
09/10/2011

A fase de especificação de requisitos é composta por diversas atividades, algumas desempenhadas de maneira individual e outras desempenhadas em grupo, conforme mostrado a seguir:

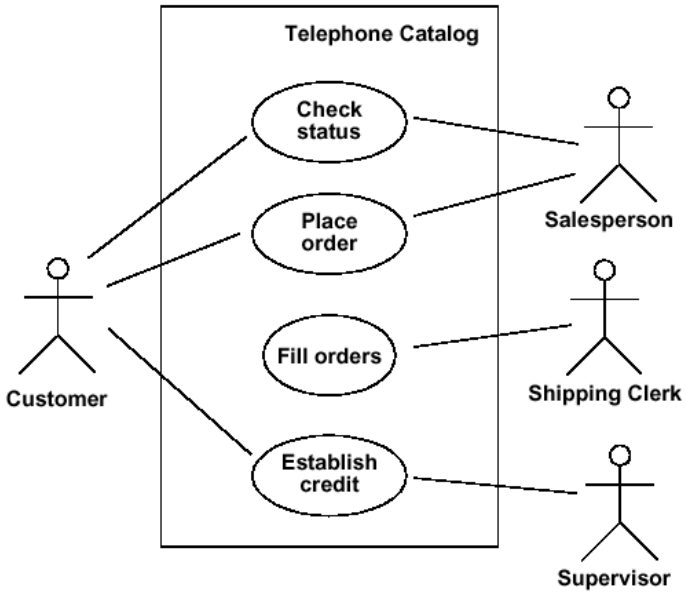


A primeira dessas atividades é a Geração do Diagrama de Casos de Uso inicial

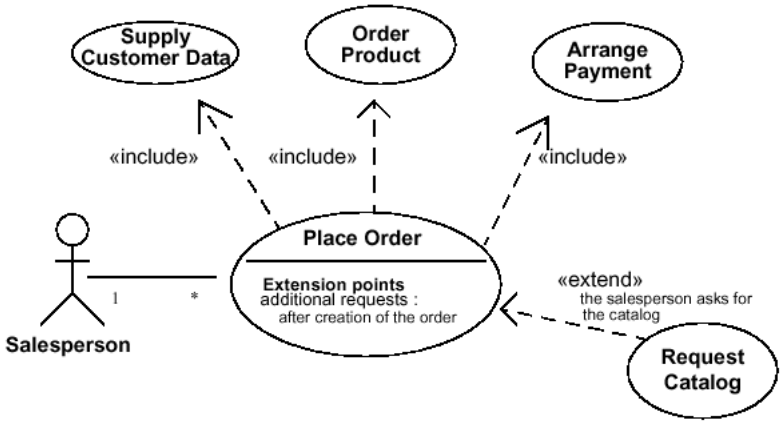
Gerar Diagrama de Casos de Uso

O objetivo dessa atividade é iniciar o processo de especificação dos requisitos, desenvolvendo cenários genéricos descrevendo a interação entre o(s) usuário(s) e o sistema. Nesta atividade, explora-se a descoberta de diferentes possíveis casos de uso. Estes casos de uso devem envolver todos os tipos de interações desejadas entre o sistema e os usuários. O resultado final desta atividade é a criação de um outline do diagrama de casos de uso.

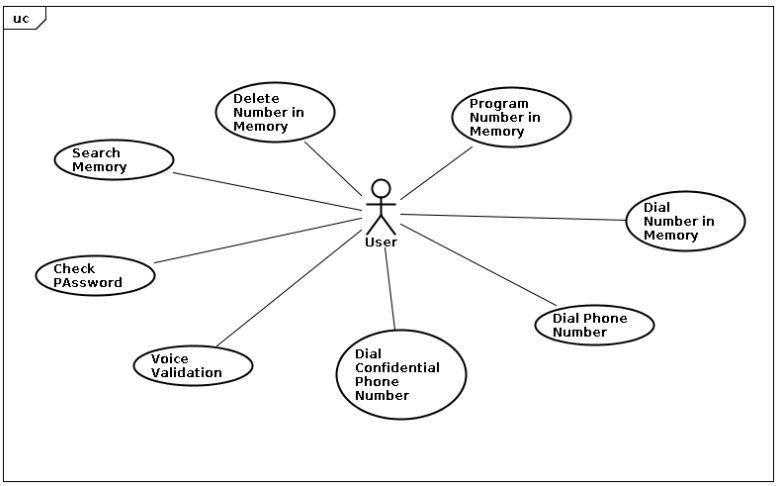
Casos de uso são abstrações de pequenas histórias narrativas envolvendo a interação entre um ou mais usuários (chamados de atores) e o sistema. A idéia é que estes casos de uso representem, por meio dessas pequenas histórias, as funcionalidades de um sistema. Os diagramas de casos de uso mostram atores e casos de uso juntos com seus relacionamentos. Um exemplo de um diagrama de casos de uso pode ser visto na figura a seguir:



Casos de uso podem ter pontos de extensão, ou seja, referências a uma localização dentro de um caso de uso onde sequências de ações de outros casos de uso podem ser inseridas. Cada ponto de extensão tem um único nome dentro de um caso de uso. Um exemplo do uso de casos de uso é mostrado na figura a seguir:



Nessa primeira atividade da especificação dos requisitos, o objetivo é levantarmos o maior número possível de casos de uso e os atores que dele participam. O diagrama criado aqui será somente um outline do modelo final. Esse diagrama será refinado em atividades posteriores. Um exemplo de um diagrama de casos de uso para um sistema de telefonia celular é apresentado na figura a seguir:



Observe que nesta atividade, o mais importante é evidenciarmos o maior número possível de casos de uso, sem maiores preocupações quanto ao possível relacionamento entre eles. Esse relacionamento será analisado posteriormente quando da estruturação do modelo de casos de uso.

Nesta atividade, cada participante da equipe deve tentar desenvolver um diagrama de casos de uso. Neste ponto, os casos de uso não precisam ainda estar estruturados. O mais importante é tentar identificar o maior número de casos de usos

Consolidar Diagrama de Casos de Uso

A etapa anterior visava uma prospecção de diferentes casos de uso, que deve ser executada individualmente por cada membro da equipe participante. Nesta etapa, realizada em grupo, as contribuições individuais de cada membro da equipe devem ser considerada, de forma a resultar um modelo único, incorporando as contribuições de todos os membros da equipe. Durante esta etapa, deve-se também uniformizar os nomes utilizados para os casos de uso, que devem, preferencialmente, reforçar o aspecto de que casos de uso transcorrem no tempo, e não são eventos individuais. Assim, deve-se preferir o nome "Verificação de Password", ao invés de "Verifica Password", "Edição do Texto", ao invés de "Edita Texto", e assim por diante. O resultado desta etapa é um diagrama de casos de uso consolidado (entretanto, ainda não estruturado).

Priorização dos Casos de Uso

O objetivo desta atividade é coletar subsídios para a priorização dos casos de uso, determinando quais deles devem ser desenvolvidos (i.e. analisados, desenhados, implementados, etc.) nas primeiras iterações e quais podem ser desenvolvidos nas iterações posteriores.

Os resultados são então capturados na visão arquitetural do modelo de casos de uso. Esta visão é considerada pelo gerente de projeto e utilizada para o planejamento do que deve ser desenvolvido na iteração corrente. Este planejamento leva em consideração também aspectos não-técnicos, tais como aspectos políticos ou estratégicos. A visão arquitetural deve destacar os casos de uso que descrevem funcionalidades críticas ou importantes (dentre outras, a inicialização do sistema e o encerramento do sistema, sem as quais um protótipo do sistema não poderia ser desenvolvido).

Assim, a escolha dos casos de uso que serão implementados na iteração corrente deve incluir aqueles casos de uso que são mais vitais para a aplicação em questão. Casos de uso referentes a características marginais ou adicionais devem ser postergados para iterações futuras.

Em nosso caso, iremos adotar uma escala de prioridade que vai de 1 a 10, onde 1 corresponde a um caso de uso de baixa prioridade e 10 será um caso de uso de alta prioridade. Atribuiremos portanto uma prioridade a cada um dos casos de uso levantados na atividade anterior, e a seguir definiremos quais serão os casos de uso a serem implementados na corrente iteração, baseados nas prioridades atribuídas.

Como resultado dessa etapa, deve-se atribuir a cada membro da equipe de 1 a 2 casos de uso, de acordo com as prioridades levantadas.

Detalhamento dos Casos de Uso em Diagramas de Atividades

Nesta atividade, o objetivo é descrever de maneira detalhada os casos de uso selecionados na etapa anterior, referenciando de maneira minuciosa o fluxo de eventos entre atores e o sistema, incluindo-se como o caso de uso começa, termina e quais as atividades realizadas tanto pelos atores como pelo sistema.

A descrição de casos de uso pode ser realizada, em princípio, por meio de diferentes tipos de artefatos de software:

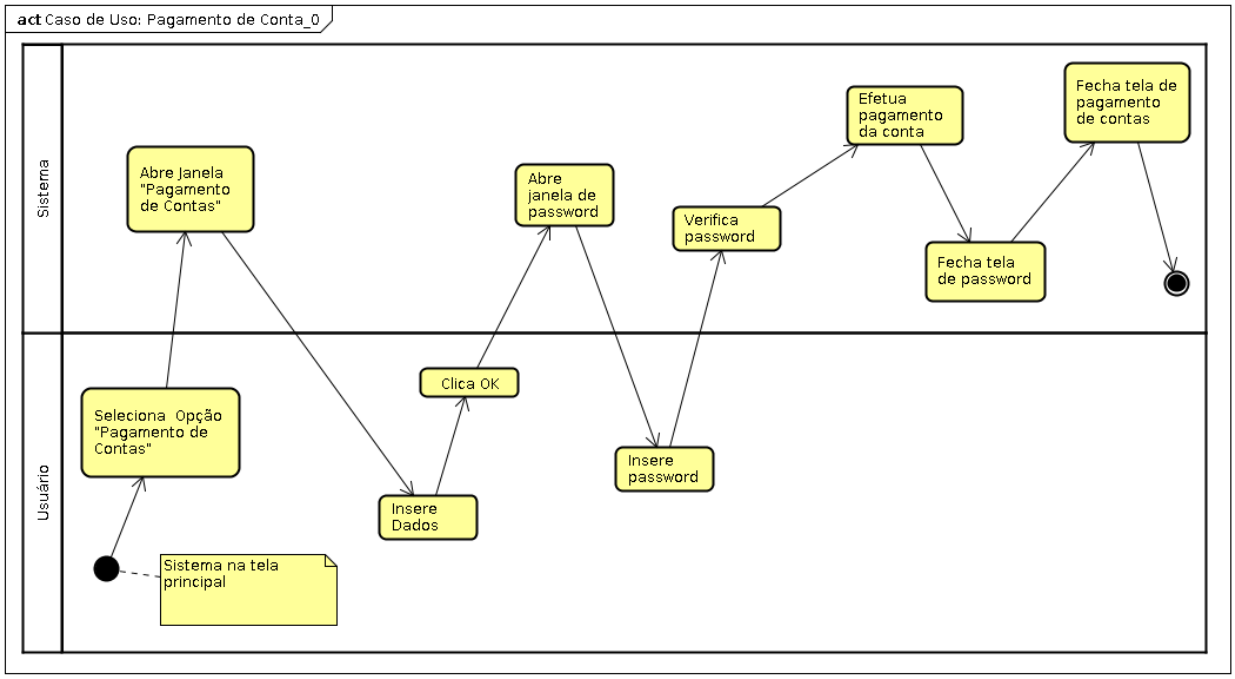
- Texto (sumarizada ou detalhada)
- Diagrama de Sequência
- Diagrama de Atividades

A escolha do artefato ideal depende do grau de complexidade do caso de uso. Quando o caso de uso é detalhado por meio de texto, diversos fluxos de atividade precisam ser descritos. O primeiro deles, chamado de fluxo principal, considera um cenário prototípico em que a funcionalidade desejada é implementada. Os demais fluxos, chamados de fluxos alternativos, consideram cenários alternativos, eventualmente onde o usuário adota outras opções (quando elas são possíveis) ou eventualmente cenários onde ocorrem desistências por parte do usuário, que solicita o cancelamento de decisões tomadas anteriormente.

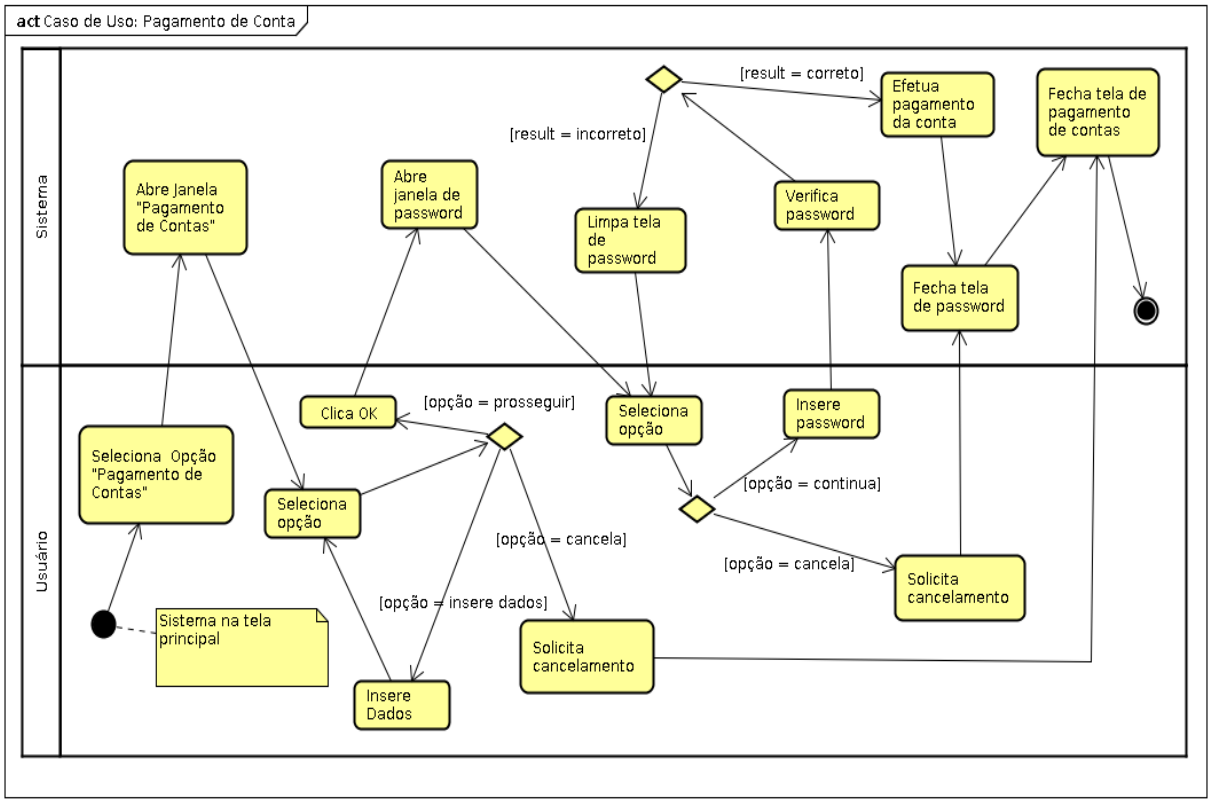
Apesar do detalhamento de casos de uso por meio de texto simples ser bastante popular na comunidade de desenvolvimento de software, uma maneira bastante efetiva de detalhar um caso de uso é por meio de um diagrama de atividades. Durante o detalhamento de um caso de uso, o número de fluxos alternativos pode ser grande e, no caso do detalhamento via texto, existe grande chance de que algum fluxo alternativo seja esquecido de ser especificado, o que acabará resultando em uma especificação deficiente. Uma das vantagens de se detalhar o caso de uso por meio de um diagrama de atividades é que todos os fluxos alternativos podem ser construídos sobre um mesmo diagrama, facilitando a verificação do mesmo quanto a fluxos potencialmente esquecidos. Utilizando um diagrama de atividades para especificar um caso de uso, utilizamos dois *swimlanes*, um para o usuário (ou um para cada usuário, quando o caso de uso possui mais de um usuário) e outro para o sistema. Os diagramas de atividades permitem uma descrição bem precisa das diversas alternativas que um caso de uso pode exibir. O único cuidado que devemos ter é o de não confundir o diagrama de atividades com um fluxograma. O diagrama de atividades deve descrever, de maneira clara e inequívoca, quais as ações realizadas pelo usuário e quais as ações realizadas pelo sistema. Um erro comum quando se constrói um diagrama de atividades é não atentar para esse ponto.

Como devemos fazer o detalhamento de um caso de uso por meio de um diagrama de atividades ? Inicialmente devemos incluir o estado inicial, normalmente acompanhado de uma pré-condição, colocada por meio de uma nota. Essa pré-condição dirá como e quando o caso de uso começa. Em seguida, devemos construir uma sequência básica que descreve o caso de uso, onde o usuário executa certas ações e o sistema responde com ações correspondentes. Na construção dessa sequência básica, chamada de fluxo principal, podemos supor que tudo dá certo e que o sistema reage da maneira correta, de acordo com a funcionalidade que se deseja obter. Por fim, deve-se definir como e quando o caso de uso termina. Essa definição normalmente se dá na forma de pós-condições, também indicadas por meio de uma nota. As pós-condições, entretanto, não são neessárias, podendo ser omitidas em alguns casos onde as mesmas são óbvias.

Um exemplo de um fluxo principal, para um caso de uso de nome “Pagamento de Conta” é descrito na figura a seguir. Neste exemplo, por ser um caso de uso curto, optamos por desenvolver as *swimlanes* horizontalmente. Na maioria das situações, entretanto o uso de *swimlanes* verticais se mostra mais adequado a descrições mais longas. Observe que na descrição do fluxo principal, detalhamos um protocolo de ações em que o usuário faz alguma coisa (ou uma sequência de coisas) e o sistema reage executando também alguma ação ou sequência de ações. É importante lembrarmos sempre de especificar como o sistema deve reagir. É muito comum nos esquecermos de dizer que o sistema precisa criar ou fechar uma janela, no caso de uma aplicação tradicional, ou exibir uma tela em um browser, no caso de uma aplicação web.



Após a definição desse fluxo básico, é necessário incluir em seguida os fluxos alternativos. Esses fluxos alternativos existem porque as vezes o usuário pode ter diferentes opções, ou nem sempre tudo dá certo durante o transcorrer de um caso de uso. O password que foi digitado pode estar incorreto, o usuário pode querer desistir de continuar com o caso de uso e desejar encerrá-lo prematuramente, ou mesmo o caso de uso pode prever múltiplas decisões em alguns pontos do mesmo. Desta maneira, é necessário adaptar o fluxo principal para incorporar os fluxos alternativos. Esses fluxos alternativos são inseridos por meio de nós de decisão, precedidos de ações do tipo "seleciona opção". A opção selecionada é então testada por meio de condições de guarda nos nós de decisão, e os diferentes tipos de caminhos alternativos podem ser inseridos



Ao detalhar um caso de uso por meio de diagramas de atividades, podemos visualizar em um mesmo diagrama dessa forma, não tão somente o fluxo principal, mas todos os possíveis fluxos alternativos. Essa visualização, ao contrário da descrição textual de casos de uso, permite que avaliemos se de fato todos os fluxos alternativos desejados foram inseridos, evitando que o sistema venha a apresentar, mais a frente, situações em que o usuário se veja em dificuldade em virtude de uma especificação mal-feita. Deve-seprestar especial atenção ao fato de que, a cada retorno do sistema, deve-se dar a opção do usuário cancelar a execução do caso de uso e finalizá-lo prematuramente.

Devemos atentar, também, que antes de um nó de decisão, precisa haver uma ação que nos leve ao cálculo de uma variável de decisão. Essa variável de decisão será então testada no nó de decisão, favorecendo o desvio dentre múltiplos caminhos. Da mesma forma, após um nó de decisão, sempre precisará haver uma ação correspondente que ratifique a decisão. Assim, se decidimos prosseguir, devemos efetuar uma ação (clicar no botão OK, por exemplo), que ratifica essa decisão. É muito comum desenvolvedores iniciantes se esquecerem desses detalhes. Verifique no diagrama de exemplo anterior, que a cada nó de decisão sempre existe uma ação anterior que calcula uma variável de decisão. Da mesma forma, sempre existe um teste dessa variável nos arcos que saem do nó de decisão, e sempre existe uma ação posterior que ratifica essa decisão.

Em nosso caso, como devemos proceder ? A cada um dos casos de uso atribuídos ao membro da equipe, o mesmo deve trabalhá-lo individualmente, prestando atenção para incluir todos os fluxos alternativos necessários ao bom desempenho da funcionalidade ensejada. Esta atividade deve ser executada, entretanto, de maneira concomitante com a atividade a seguir: a prototipação da interface com o usuário.

Prototipação da Interface com o Usuário

Nesta atividade, o principal objetivo é construir um protótipo das interfaces com o usuário. Essas interfaces não serão as interfaces definitivas, mas devem conter as informações necessárias para a efetivação dos casos de usos descritos na etapa anterior. Assim, critérios estéticos não são importantes nessa atividade. O importante é a identificação dos elementos de comunicação necessários à interação com o sistema.

Esta atividade deve ser realizada de maneira concomitante com a atividade anterior. Para que o usuário se comunique com o sistema, e este, por sua vez, se comunique com o usuário, é necessário que haja uma interface. Dessa forma, à medida que os diagramas de atividades da atividade anterior são construídos, a cada vez que o sistema reage a uma ação do usuário, essa ação deve ser descrita em termos de uma interface. Essa interface é a que é criada nesta etapa. O resultado final que se espera nesta atividade é uma lista protótipos de interface, devidamente nomeadas, que devem ser referenciadas na descrição dos diagramas de atividades.

Como já ressaltamos, essas interfaces não serão as interfaces definitivas, mas servirão de inspiração posteriormente na fase de design quando elementos não funcionais tais como cores, tamanhos, fontes, ocupação do espaço, etc serão considerados.

Estruturação do Modelo de Casos de Uso

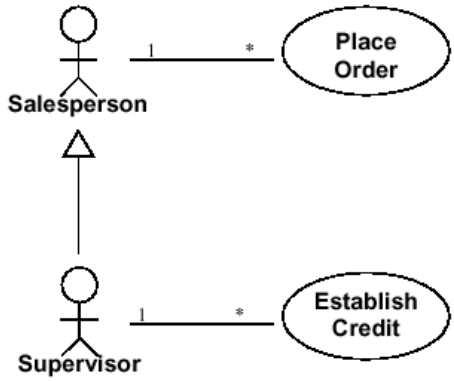
Nesta atividade, o objetivo é reestruturar os elementos do modelo de casos de uso capturados anteriormente (que podem ter sido capturados até de maneira independente entre si) e gerar um modelo que seja homogêneo, consistente e simples de ser interpretado. Durante o detalhamento dos casos de uso, pode ser que algum caso de uso em particular tenha se tornado demasiadamente complexo ou detalhado, quando então se torna útil desmembrá-lo em mais de um casos de uso. Da mesma forma, pode ser que se tenha percebido que muitos casos de uso possuem uma estrutura comum, que se repete diversas vezes nos diagramas de atividades desenvolvidos. Dessa forma, nesta etapa deve-se fazer um refactoring dos casos de uso, de tal forma que os mesmos sejam

repensados e devidamente estruturados, para que a etapa de especificação de requisitos termine definitivamente. Antes de procedermos com a descrição desta atividade em maiores detalhes, vamos ver algumas características dos tipos de relacionamentos que podem surgir em versões estruturadas de diagramas de casos de uso.

Basicamente, 4 tipos de relacionamentos podem ser indicados em diagramas de casos de uso:

- Associações
- Extend
- Include
- Generalização

As **associações** denotam a participação de atores em um caso de uso. É o único tipo de relacionamento entre um ator e um caso de uso. Um relacionamento do tipo **extend** é uma relação entre um caso de uso A para um caso de uso B que indica que uma instância do caso de uso B pode ser aumentada (sujeita a condições específicas de extensão) pelo comportamento especificado por A. Esse comportamento é inserido conforme especificado por um ponto de extensão em B. Um relacionamento do tipo **include** é uma relação entre um caso de uso A para um caso de uso B que indica que uma instância do caso de uso A contém o comportamento especificado por B. Esse comportamento é incluído na localização indicada em A por um ponto de extensão. Um relacionamento do tipo **generalização** entre um caso de uso A e um caso de uso B indica que A é uma especialização de B. Normalmente, essa generalização implica em que B é uma descrição mais abstrata de uma situação, e A é uma descrição mais detalhada. Além dos casos de uso, é possível utilizarmos generalizações para indicarmos o relacionamento entre dois atores. Um exemplo deste tipo é apresentado na figura a seguir:



Observe que nesse caso, o ator Supervisor é um um tipo de Salesperson, ou seja, uma especialização de um Salesperson. Observe ainda pelo exemplo que as associações entre atores e casos de uso podem também conter uma cardinalidade.

As associações entre os atores e os casos de uso podem ainda possuir uma navegabilidade (seta entrando ou saindo do caso de uso). Essa navigabilidade indica quem inicia o caso de uso. Caso a seta seja do ator para o caso de uso, é o ator quem inicia a interação. Caso seja do caso de uso para o ator, é o sistema quem inicia a interação. Veja o exemplo na figura a seguir:

Neste exemplo, que mostra o caso de uso Participação em Conferência Eletrônica, é o professor quem inicia as atividades. O aluno passa a interagir posteriormente, a partir da iniciativa do sistema.

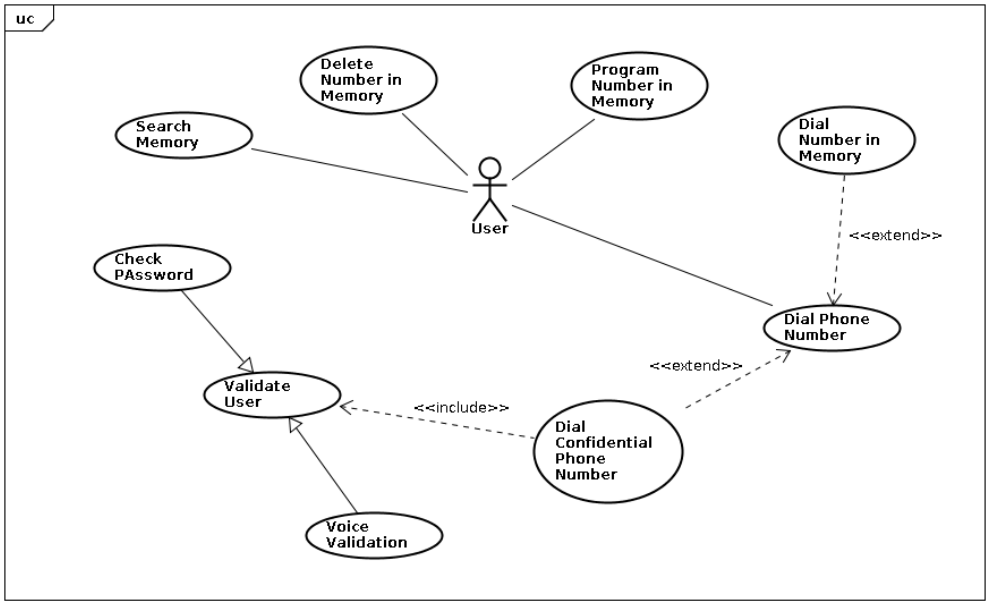
Muito bem ! Entendidos estes relacionamentos, vamos às atividades desta fase. O primeiro passo é

tentar identificar descrições compartilhadas de funcionalidades entre os casos de uso levantados anteriormente. Como isso ocorre ? Durante o levantamento dos casos de uso, podem haver atividades ou partes de atividades que são semelhantes em diversos casos de uso. Pode-se detectar uma situação como essa quando observamos que certas partes dos diagramas de atividades se repetem em mais de um casos de uso. De forma a reduzir a redundância, os casos de uso podem ser então reestruturados para tornar o modelo como um todo mais enxuto. Assim, esses trechos de casos de uso podem se tornar casos de uso independentes, que são reutilizados no modelo por meio da relação de generalização. Essas modificações no diagrama de casos de uso deve ser seguida de modificações equivalentes nos diagramas de atividades, para refletir essas alterações, quando for o caso.

O segundo passo é tentar identificar descrições adicionais ou opcionais de funcionalidade. Sabemos que o mecanismo de extensão permite a inserção de adições ao comportamento básico de casos de uso. Como vimos, esse relacionamento inclui as condições para a extensão e o ponto de extensão, onde o caso de uso deve ser inserido. O que faremos então é tentar identificar situações desse tipo, alterando o diagrama de casos de uso de modo a incluir relacionamentos de extensão. Os diagramas de atividades associados devem ser retrabalhados também, de forma a refletir as mudanças implementadas.

O terceiro passo é tentar identificar descrições repetidas de funcionalidade. Sabemos que o relacionamento de inclusão permite a inserção incondicional e explícita do comportamento de um caso de uso em outros casos de uso. Tentaremos então verificar se situações desse tipo ocorrem, e nesse caso procederemos à inserção de relacionamentos do tipo include em nosso diagrama. Alterações equivalentes devem ser implementadas nos diagramas de atividades referenciados

Nesse ponto, talvez vocês estejam se perguntando … Ahn ? Qual a diferença então entre o relacionamento de generalização e o relacionamento de inclusão. Temos que ter muito cuidado pois esses dois tipos de relacionamentos são muito semelhantes. A grande diferença que existe entre esses dois é que no relacionamento de generalização, o caso de uso que é generalizado é um caso de uso mais abstrato que, apesar de semelhante, será diferente em cada uma de suas especializações. Um bom exemplo é o exemplo mostrado na figura abaixo, fruto da estruturação do modelo de casos de uso que apresentamos anteriormente.



Observe que anteriormente tínhamos dois casos de uso, o caso de uso *CheckPassword* e o *VoiceValidation*, que apresentavam um compartilhamento de funcionalidades. Ambos serviam para efetuar uma validação do usuário. Uma maneira de refinar nosso modelo foi criar um novo caso de

uso abstrato chamado *ValidateUser*, que passou então a ser uma abstração, tanto de *CheckPassword* como de *VoiceValidation*. Para compreendermos a diferença entre o uso de uma generalização e de uma inclusão, observe que a descrição de uma checagem de password não tem nada a ver, em princípio com uma validação vocal. Em termos concretos, eles são casos de uso completamente diferentes. Entretanto, quando abstraímos o que ocorre em cada um desses casos de uso, vemos que ambos nada mais fazem do que validar o usuário. Por isso, usamos uma relação de generalização. O relacionamento do tipo inclusão, ao contrário, inclui totalmente o caso de uso como um sub-trecho de um caso de uso mais complexo. Assim, devemos ter cuidado durante esse refinamento. Em alguns casos, será necessário fazermos abstrações e incluirmos novos casos de uso mais abstratos. Em outros, vamos tentar verificar a mera repetição de situações e utilizaremos inclusões. Observe que no diagrama do exemplo, verificamos que a validação do usuário ocorre somente quando ocorre uma discagem de número confidencial. Por isso, colocamos o caso de uso *ValidateUser* como uma inclusão de *DialConfidentialPhoneNumber*. Por outro lado, tanto *DialConfidentialPhoneNumber* quanto *DialNumberInMemory* extendem a descrição de *DialPhoneNumber*. Por isso, colocamos ambos como extensões de *DialPhoneNumber*.

Repare que após chegarmos ao diagrama de casos de uso final, pode ser necessário que modifiquemos a descrição dos casos de uso feitas anteriormente, como por exemplo a inclusão de novos casos de uso (vide a inclusão de *ValidateUser* no exemplo). Essas modificações devem então ser implementadas, para que a fase de especificação de requisitos se dê por encerrada.

Bibliografia

(Jacobson et.al. 1999) Ivar Jacobson, Grady Booch, James Rumbaugh - **The Unified Software Development Process** - Addison Wesley, 1999.

(Larman 1998) Craig Larman - **Applying UML and Patterns - An Introduction to Object Oriented Analysis and Design** - Prentice Hall Inc., New Jersey 1998.