Diagramas de Atividade e Diagramas de Estado

Ricardo R. Gudwin
DCA-FEEC-UNICAMP

Introdução

Neste texto, apresentamos um sumários de dois tipos de diagramas UML: o diagrama de atividades e o diagrama de estados.

Diagrama de Atividades

O diagrama de atividades é um diagrama UML utilizado para modelar o aspecto comportamental de processos. É um dos diagramas que mais sofreu mudanças em seu meta-modelo, desde seu surgimento no UML 1.0. Neste diagrama, uma **atividade** é modelada como uma sequência estruturada de **ações**, controladas potencialmente por nós de decisão e sincronismo. Em seu aspecto mais simples, um diagrama de atividades pode ser confundido com um fluxograma. Entretanto, ao contrário de fluxogramas, os diagramas de atividades UML suportam diversos outros recursos, tais como as **partições** e os nós do tipo *fork* e *merge*, além da definição de **regiões de interrupção**, que permitem uma modelagem bem mais rica do que simplesmente um fluxograma.

Um exemplo de um diagramas de atividade simples é mostrado na figura 1 a seguir.

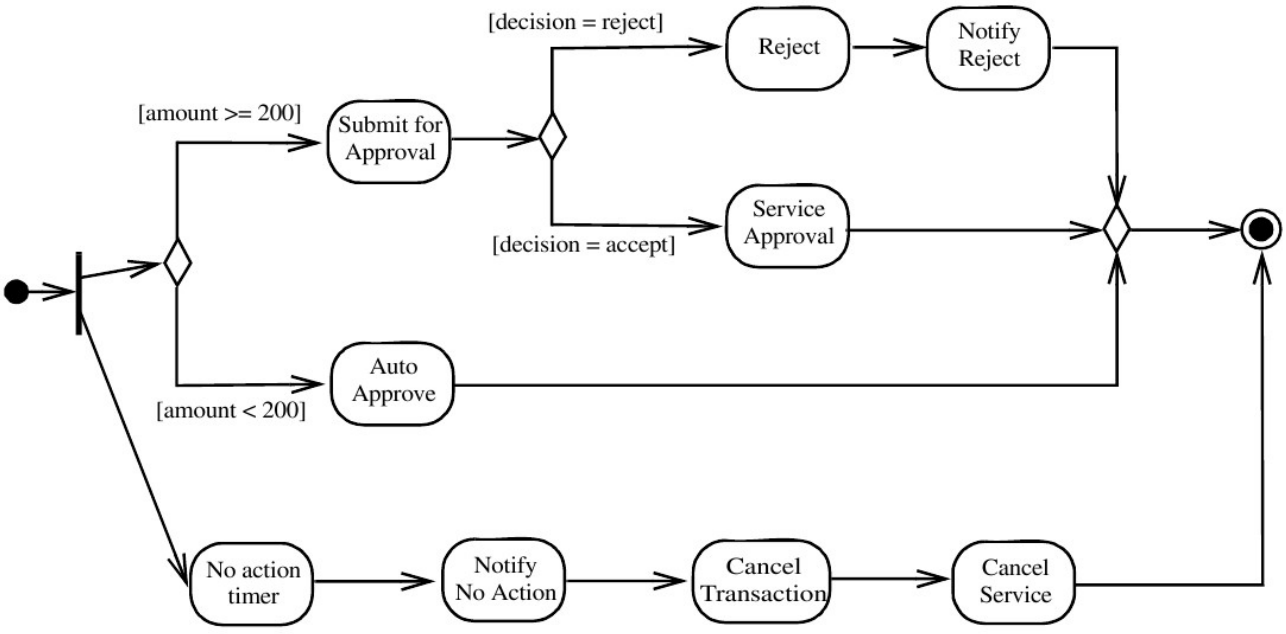


Figura 1: Exemplo de um Diagrama de Atividades Simples

Neste diagrama, o ponto preto à esquerda indica o início da atividade, as caixas com cantos arredondados representam ações, os pequenos losangos são nós de decisão e a barra vertical preta é um nó de sincronização do tipo *fork*. Os arcos conectando as ações representam a sequência em que as ações devem ser executadas, sendo que nos arcos que saem dos nós de decisão existem **condições de guarda**, que decidem qual será a próximo ação a ser executada. Essas condições de guarda devem ser proposições mutuamente exclusivas, de tal forma que para n arcos saindo de um nó de decisão, somente um deles pode ser verdadeiro a cada instante de tempo. Por fim, no canto direito do diagrama encontra-se um nó de finalização, que denota o final da atividade.

A figura 2 a seguir mostra um diagrama de atividades com partições. Diagramas com partições permitem que processos onde múltiplos agentes atuam, potencialmente em paralelo, possam ser representados.

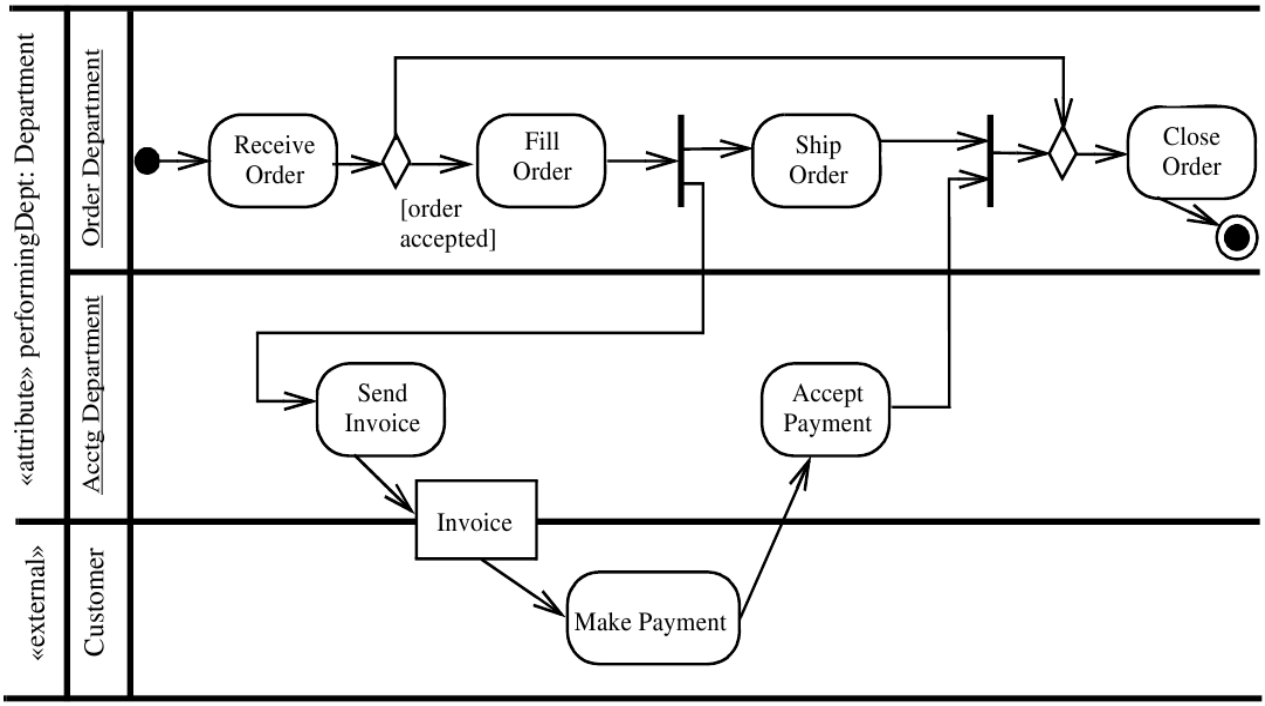


Figura 2: Diagrama de Atividades com Partições

Ação

Uma ação representa um passo elementar de uma atividade, ou seja, um passo que não pode ser decomposto dentro de uma atividade. Uma atividade representa um comportamento que pode ser composto por ações ou outras sub-atividades. Uma ação pode ter um conjunto de arcos de entrada e de saída, que especificam o fluxo de controle e de dados para outros nós. Uma ação não inicia sua execução até que todas as suas condições de entrada sejam satisfeitas. Somente quando uma ação é terminada que a ação subsequente fica habilitada.

Uma ação é representada conforme a figura 3 a seguir:

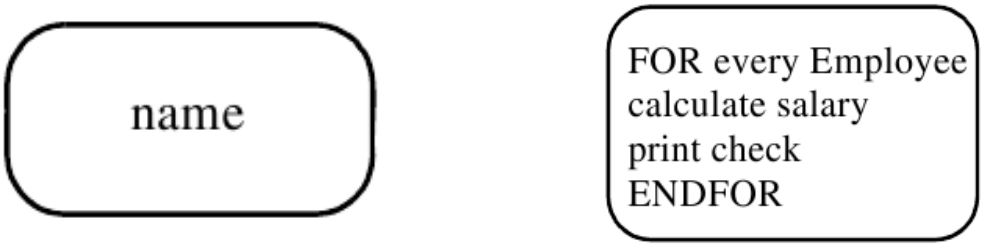


Figura 3: Exemplos de Ações

Alternativamente, ações podem ser definidas com pré-condições, que definem as condições necessárias para que a ação possa ser executada, e pós-condições, que definem o estado depois que a ação é executada. Exemplos de situações com essa podem ser vistos na figura 4 a seguir.

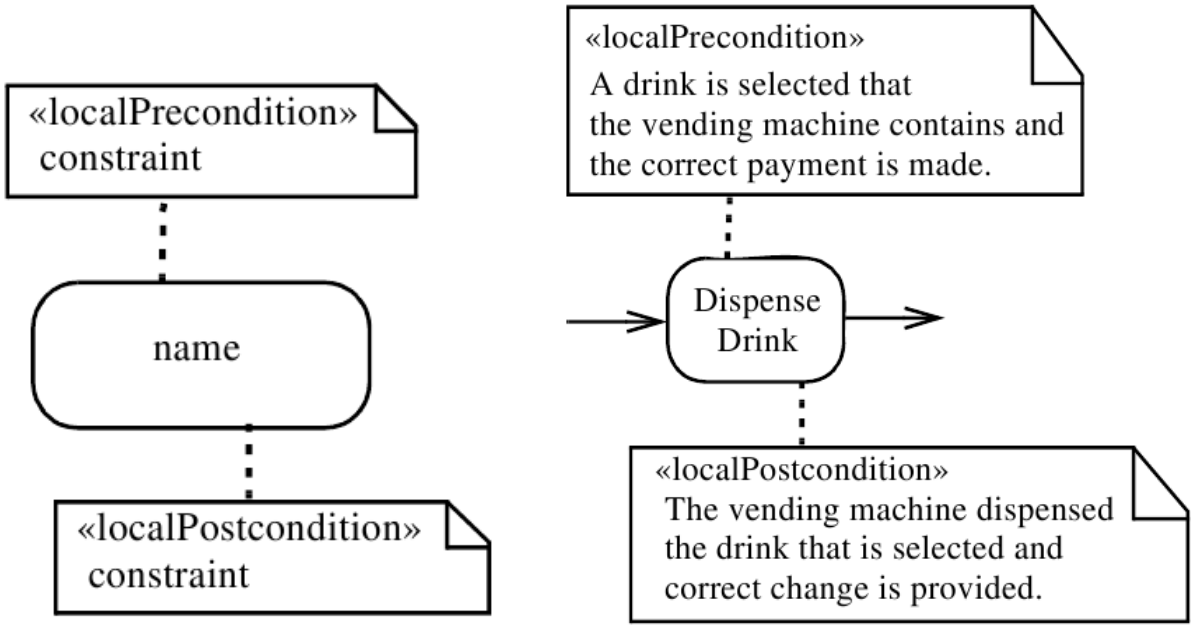


Figura 4: Ações definidas com Pré-condições e Pós-condições

Atividades

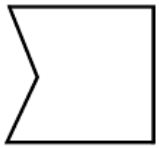
Atividades podem ser representadas por sequências de ações e também de sub-atividades. Dessa forma, para representar uma sub-atividade dentro de uma atividade (ou seja, todo um conjunto de ações ou sub-atividades), utiliza-se uma representação semelhante a de uma ação, com um pequeno ícone no canto direito inferior. A notação para uma atividade pode ser vista na figura 5 a seguir.

Figura 5: Exemplo de Atividade

Do ponto de vista formal, uma atividade conforme representada na figura 5 não é exatamente uma atividade, mas uma ação especial, chamada de CallBehaviorAction, que de maneira atômica invoca a execução de toda uma atividade. Entretanto, para efeitos práticos, podemos entendê-la como uma atividade de-per-si.

Eventos

Outros elementos que podem aparecer em um diagrama de atividades correspondem a eventos. Eventos são mudanças de estado instantâneas que propiciam o início de uma outra ação. Existem basicamente três representações para eventos. Para representar um evento único que, caso aconteça, propicia o início de uma ação subsequente, utiliza-se a ação especial *AcceptEventAction*, representada pela notação indicada na figura 6.



Accept event action

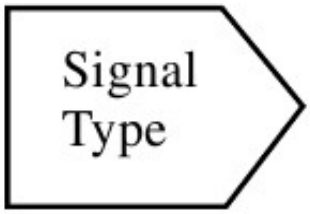
Figura 6: Evento que Propicia o início de uma Ação subsequente

Para representar um evento periódico, que acontece de tempos em tempos, e a cada vez que aconteça favoreça o início de uma ação subsequente, utiliza-se a notação apresentada na figura 7 a seguir.

Accept time event action

Figura 7: Evento Temporizado

Para representar a geração de um evento deliberado, ao final de uma ação, utiliza-se a notação indicada na figura 8 a seguir.



Send signal action

Figura 8: Geração de um Evento

As figuras 9, 10 e 11 a seguir ilustram o uso de eventos junto com ações.

Figura 9: Evento e Ação correspondente

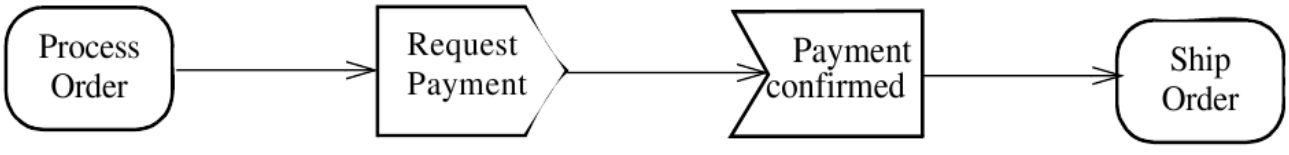


Figura 10: Geração e Recepção de Evento

Figura 11: Exemplo de Evento Temporizado e Ação

Objetos

Além do fluxo de controle, que especifica uma sequência de ações que definem um processo, um diagrama de atividades também pode representar o fluxo de dados acontecendo em um processo. Esse fluxo de dados pode ser representado definindo-se explicitamente os objetos necessários para que uma ação possa ser realizada, bem como os objetos gerados após a finalização de uma ação. Um objeto é representado da mesma maneira que em um diagrama de classes, entretanto sem a necessidade de estar sublinhado. Alguns exemplos de uso de objetos podem ser vistos na figura 12 a seguir.

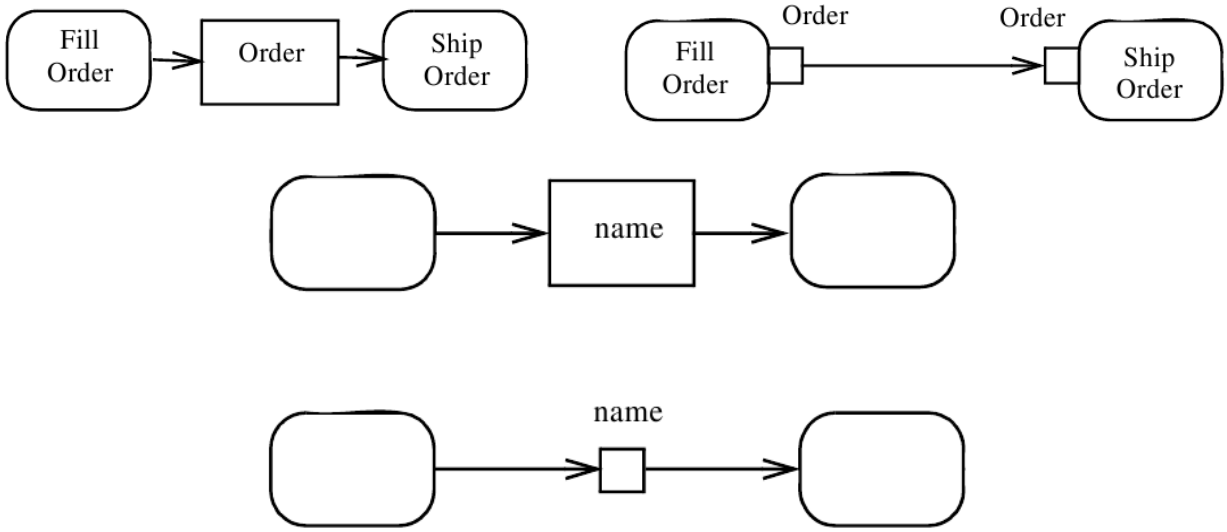


Figura 12: Exemplos do Uso de Objetos entre Ações

Objetos podem também ser passados como parâmetros para atividades completas. Veja o exemplo da figura 13.

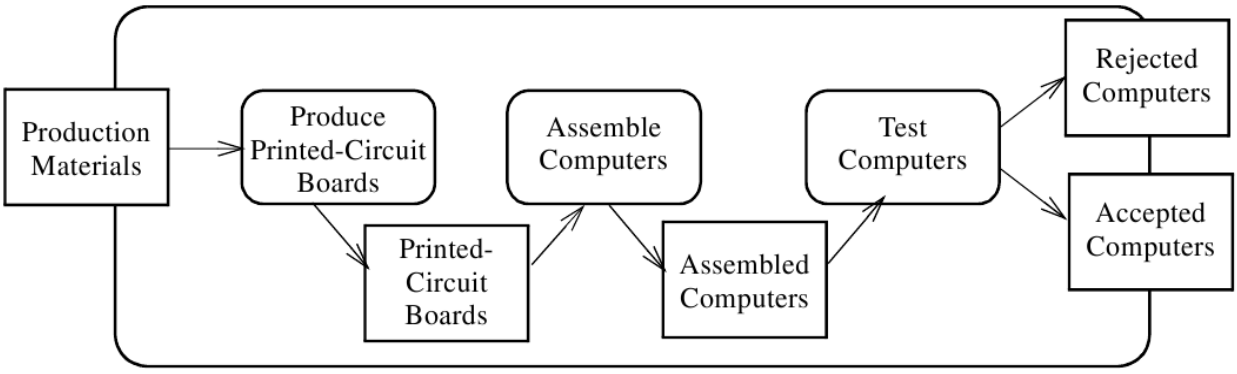


Figura 13: Objetos como Parâmetro para Atividades

Nós de Controle

Além de ações, atividades, eventos e objetos, os diagramas de atividade admitem um conjunto de assim chamados nós de controle, pois controlam o fluxo do processso. Esses nós podem ser visualizados na figura 14 a seguir.

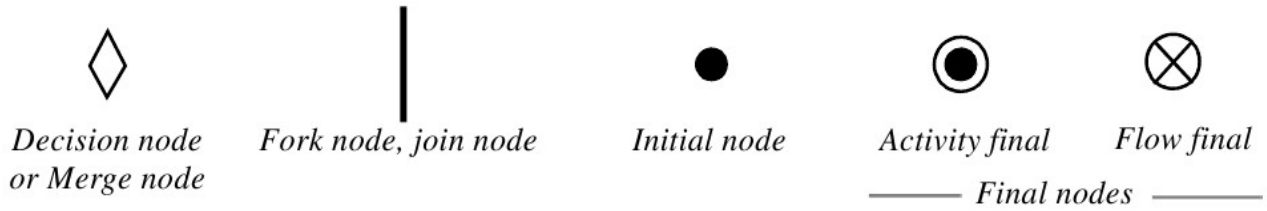


Figura 14: Nós de Controle

O primeiro nó de controle é o **nó de decisão**. Observe que um losango pode representar tanto um nó de decisão como um nó do tipo *merge*. Quando existe um único arco de entrada no nó e diversos arcos de saída, ele representa um nó de decisão. Nesse caso, cada arco de saída deve indicar uma condição de guarda, entre colchetes, condição que deve ser satisfeita para caracterizar o respectivo arco como a sequência de controle. Quando há diversos arcos de entrada e um único nó de saída, tem-se um nó do tipo *merge*. Esse nó é utilizado para agregar diversos fluxos de controle em um só.

O segundo tipo de nó de controle é o nó de sincronização. Esse nó é utilizado para representar fluxos de controle que ocorrem em paralelo, ou seja, ações que devem acontecer em paralelo. Nós de sincronização podem ser de dois tipos diferentes. O primeiro deles é um nó do tipo fork, que ocorre quando se quer indicar que, a partir daquele instante, as ações subsequentes devem acontecer em paralelo. Um exemplo de um nó do tipo fork é indicado na figura 15.

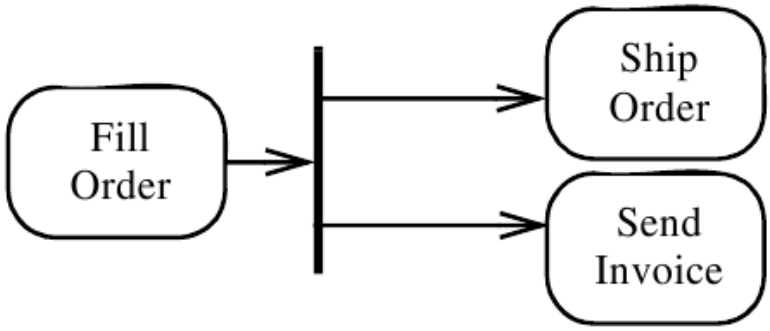


Figura 15: Exemplo de Nó do tipo fork

Na figura 16 temos o outro caso de sincronização, o nó do tipo *join*.

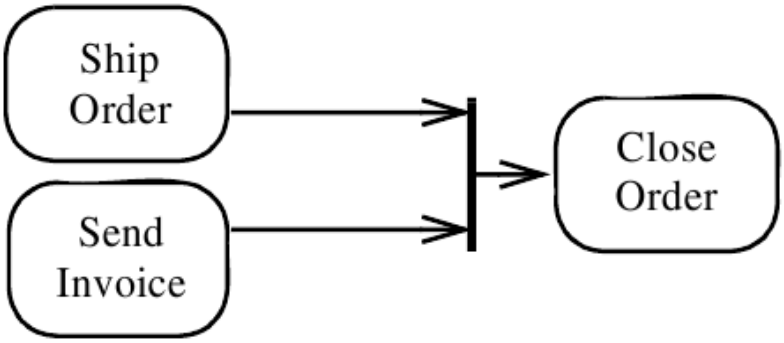


Figura 16: Exemplo de Nó do tipo join

Os nós do tipo join servem para indicar um ponto de sincronização entre fluxos de ação que estão acontecendo em paralelo. Somente quando todas as ações que chegam a um join estiverem concluídas é que a ação consecutiva pode ser executada.

O nó de início e de final já foram apresentados anteriormente. É importante, entretanto, ressaltar a diferença que existe entre um nó do tipo final de atividade e um nó do tipo final de fluxo. Quando o final de uma ação alcança um nó do tipo final de atividade, isso significa que a atividade como um todo, representada pelo diagrama termina. Quando o final de uma ação alcança um nó do tipo final de fluxo, somente o fluxo em questão é que termina, mas a atividade continua, em outros fluxos que estejam ainda em funcionamento.

Interrupções e Regiões de Interrupção

Em diagramas de atividade UML 2.0, introduziu-se o conceito de interrupção e regiões de interrupção. As interrupções (ou excessões) são notacionadas como arcos em forma de raio, conforme pode ser ilustrado na figura 17 a seguir.

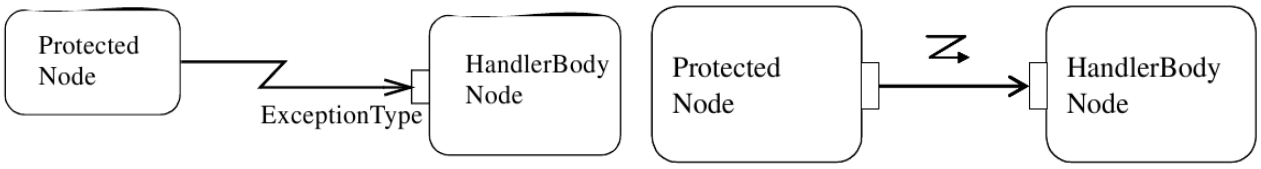


Figura 17: Exemplos de Notação para Interrupções

Pode ser interessante indicar o conjunto de ações onde as interrupções podem ocorrer. Quando isso for interessante, essas regiões podem ser demarcadas por meio de retângulos com linhas tracejadas. Um exemplo desta notação encontra-se na figura 18 a seguir.

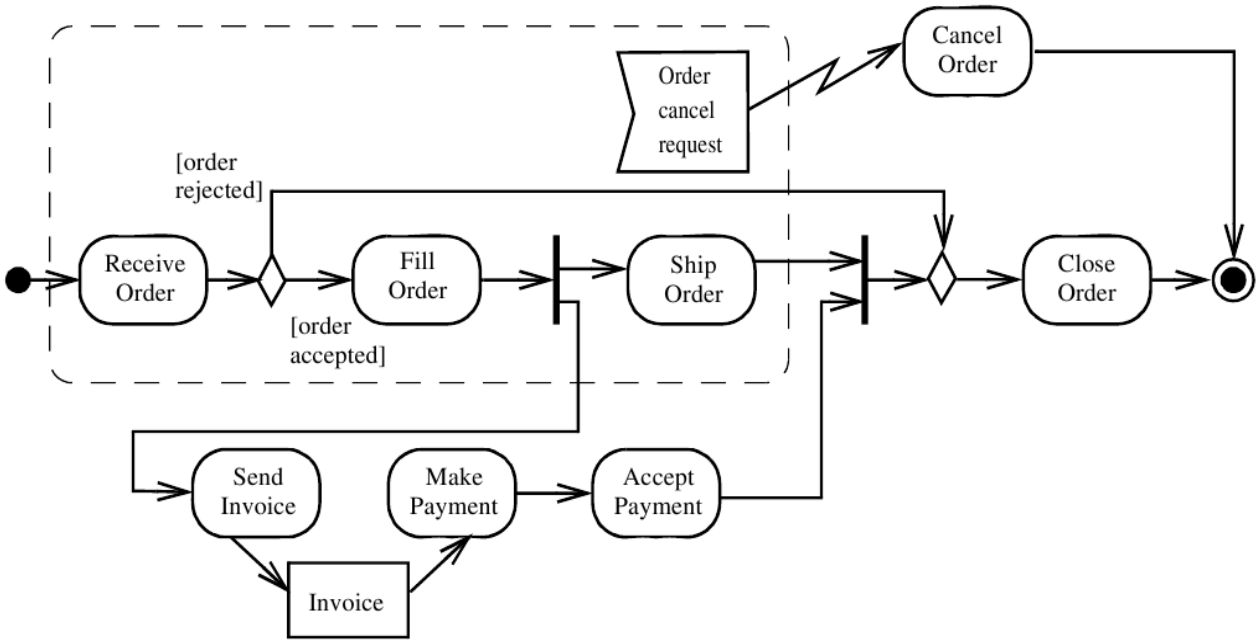


Figura 18: Regiões de Interrupção

Na figura 18 acima, representa-se que o evento de requisição de cancelamento de pedido só pode acontecer na região demarcada, ou seja, enquanto alguma das ações dentro da região estiverem sendo executadas. For a da região de interrupção, o evento não deve ser considerado.

Pontos de Extensão

Para evitar que linhas longas conectando pontos extremos de um diagrama tornem o diagrama muito poluído, podem ser utilizados pontos de extensão. Um exemplo de um ponto de extensão pode ser visto na figura 19 a seguir:

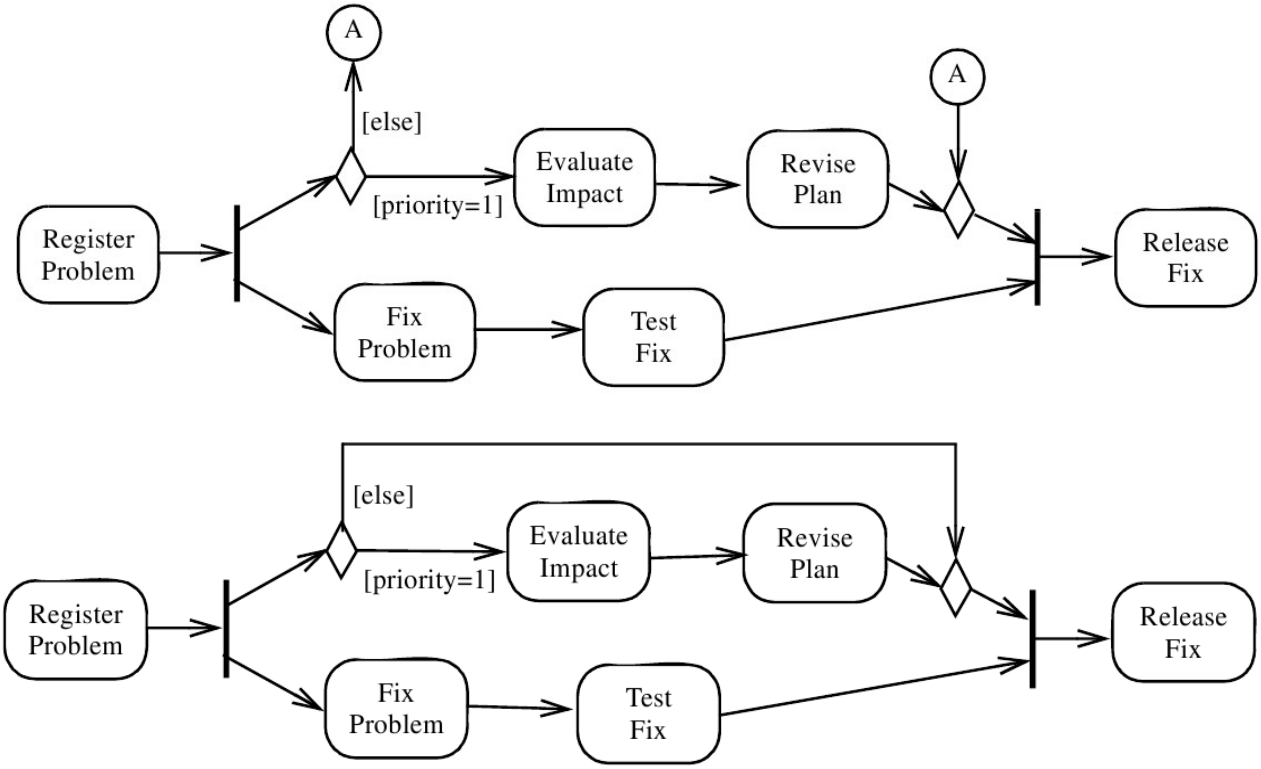


Figura 19: Pontos de Extensão

Partições

Sem o conceito de partições, os diagramas de atividade UML funcionam meramente como extensões de fluxogramas. A introdução do conceito de partição traz toda uma rica gama de representações para os diagramas UML. As partições podem ser representadas como indicado na figura 20 a seguir.

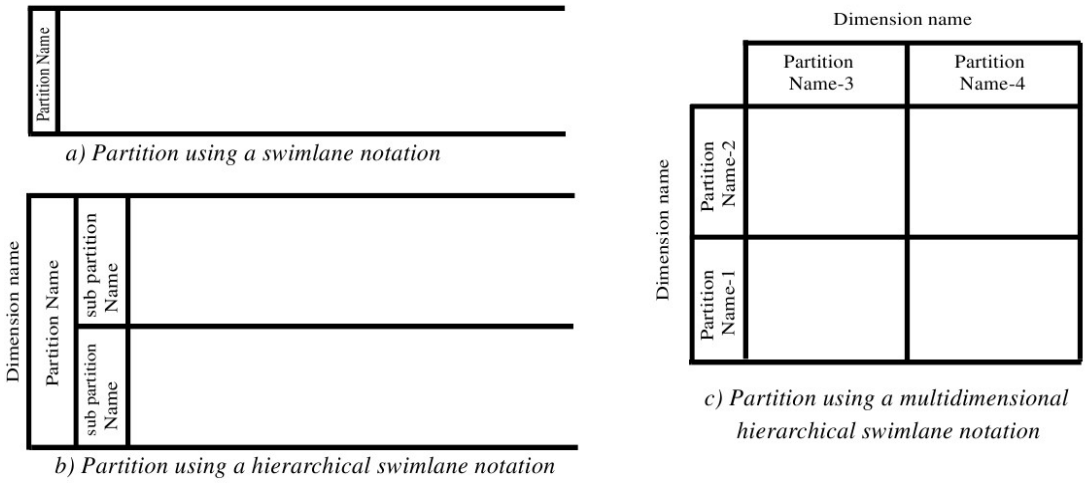


Figura 20: Exemplos de Partições

As partições servem para indicar que diferentes ações são executadas por diferentes agentes dentro de um processo. O caso mais interessante e que é dos mais importantes para nós é utilizar diagramas de atividade com partições para representar casos de uso. Nesse caso, uma partição é utilizada para representar o usuário e outra para representar o sistema. Um exemplo é apresentado na figura 21 a seguir.

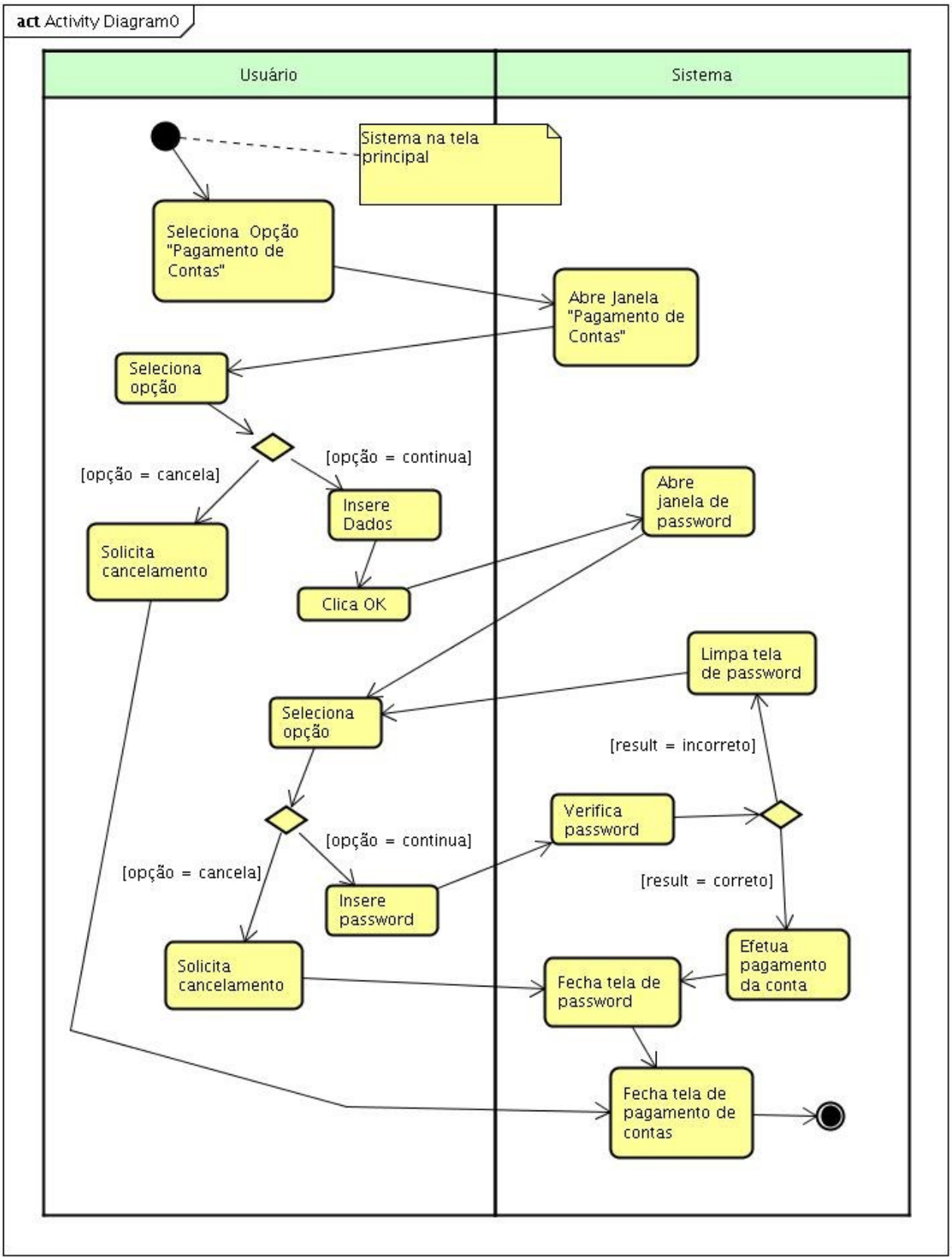


Figura 21: Uso de Diagrama de Atividades para Representar Casos de Uso

Diagramas de atividade como os da figura 21 são utilizados para detalhar os casos de uso levantados durante a especificação do sistema. Nesses diagramas, assume-se que o usuário do sistema realiza certas ações e o sistema, em resposta, reage realizando certas tarefas. Com isso, o comportamento do sistema pode ser especificado.

Outros Recursos em Diagramas de Atividades

Os diagramas de atividades possuem ainda, de acordo com a versão 2.3 da norma UML, outros recursos não apresentados aqui neste texto, que podem elevar sobremaneira a complexidade do diagrama. Dentre eles, o recurso de regiões de expansão, os pinos de entrada e saída, os nós estruturados, os conjuntos de parâmetros e outros, podem ser consultados diretamente no texto da norma.

Diagramas de Estado (Máquinas de Estado)

Os diagramas de estado (ou máquinas de estado, como aparecem na versão 2.3 da norma UML) são utilizadas para modelar um comportamento discreto em sistemas de transição entre estados finitos. Existem basicamente dois usos para máquinas de estado: máquinas de estado comportamentais e máquinas de estado para protocolos.

Máquinas de estado comportamentais podem ser utilizadas para especificar o comportamento de vários tipos de elementos. Por exemplo, podem ser utilizadas para modelar o comportamento de entidades individuais (objetos), por meio da modificação dos valores de seus atributos. O formalismo de máquina de estados neste caso é uma variante orientada a objetos dos *Statecharts* de Harel.

Máquinas de estado para representar protocolos expressam as transições legais que um objeto pode desenvolver. Com seu uso, pode-se definir o ciclo de vida de objetos, ou uma determina ordem na invocação de suas operações. Para este tipo de máquina de estado, interfaces e portas podem estar associados.

Os diagramas de estado UML possuem diversos detalhes que o tornam uma poderosa ferramenta para a modelagem de transição entre estados. Entretanto, neste texto, não iremos tratar de todos seus recursos, nos limitando a abordar suas características básicas. Leitores interessados nos detalhes mais sofisticados devem se remeter diretamente à norma.

Estado

Um estado modela uma situação durante a qual alguma condição (usualmente implícita) se mantém. Essa invariância tanto pode representar uma situação estática, como um objeto aguardando que um evento externo ocorra, como condições dinâmicas, como um processo apresentando um determinado comportamento. A notação para um estado pode ser visualizada na figura 22 abaixo.

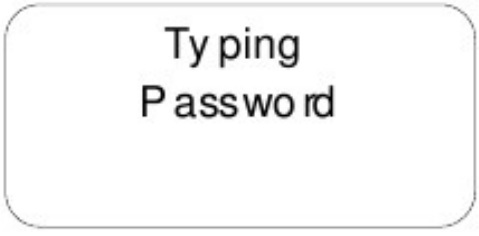


Figura 22: Representação para um Estado

Alternativamente, um estado pode ter seu espaço interno subdividido em compartimentos, conforme ilustra a figura 23.

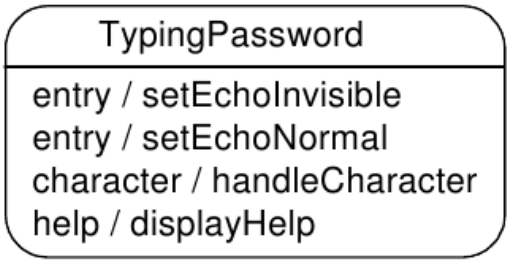


Figura 23: Exemplo de Estado com Compartimento Interno

O compartimento interno pode abrigar uma lista de ações ou atividades que podem ser realizadas quando se entra (*entry*), sai (*exit*) ou permanece (*do*) no estado. Outros tipos de comportamentos podem também ser definidos. A figura 24 mostra um exemplo de um pequeno diagrama de estados que usa esses recursos.

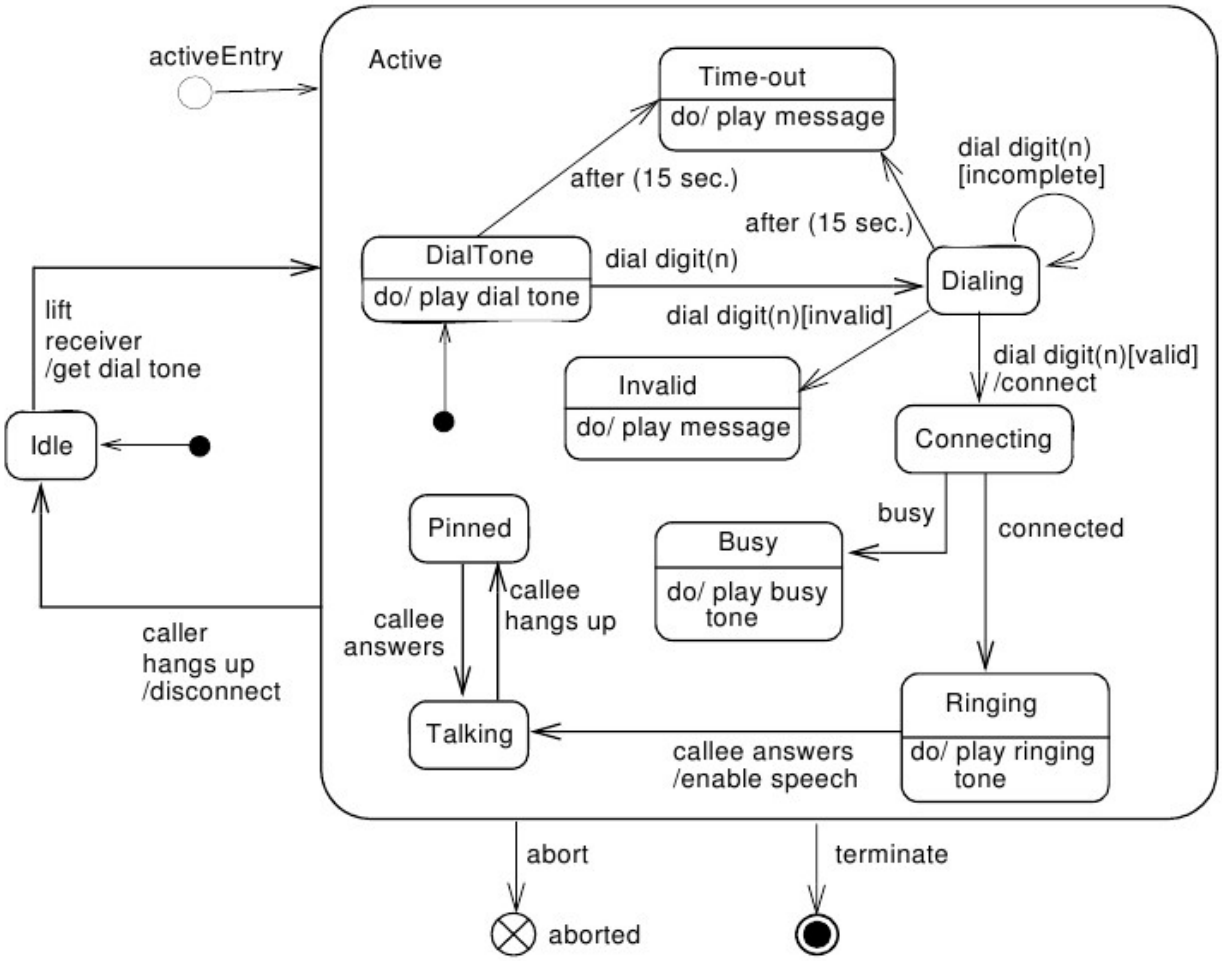


Figura 24: Exemplo de um Diagrama de Estado

A notação para os arcos que ligam os estados pode assumir a seguinte sintaxe, visualizada na figura 25.

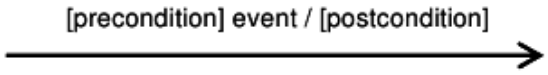


Figura 25: Notação para os Arcos entre Estados

Os estados não necessariamente precisam utilizar os compartimentos internos. O diagrama mostrado a seguir na figura 26 também é um diagrama de estados.

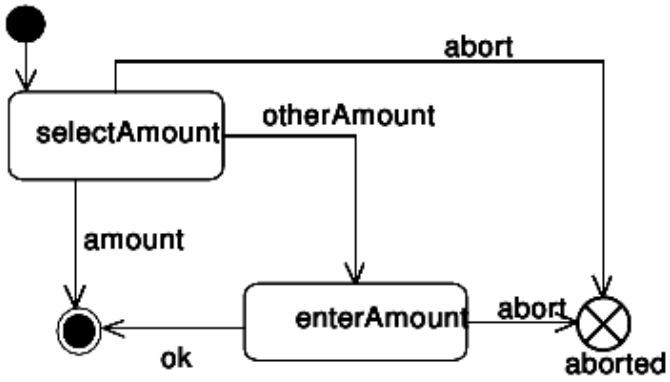


Figura 26: Estados sem Compartimentos Internos

Os diagramas de estado UML podem adquirir níveis bastante sofisticados de complexidade. A figura 27 a seguir ilustra um pouco da complexidade possível

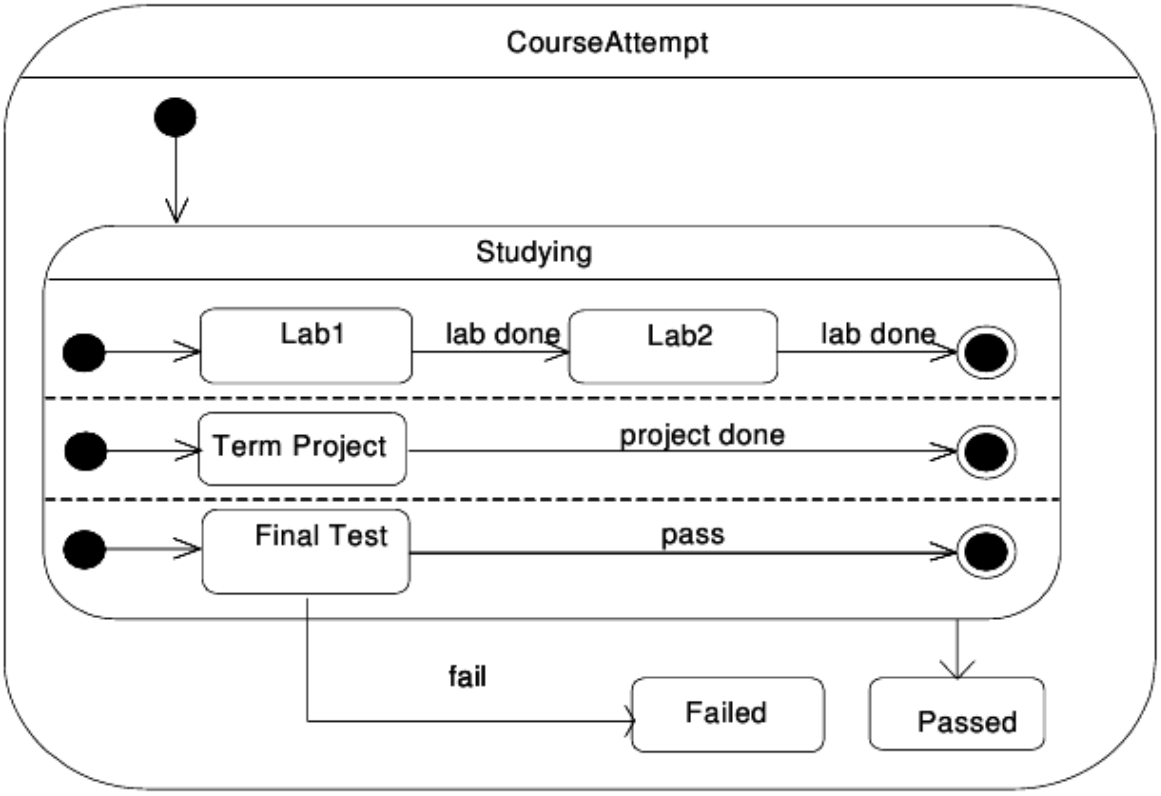


Figura 27: Diagrama de Estados mais Complexo

Como os estados de um diagrama de estados não demandam o uso dos compartimentos internos, observa-se que a notação é bastante semelhante à notação de uma ação em um diagrama

de atividades. É necessário, entretanto, deixar claro que os dois diagramas têm uma semântica bastante diferente entre si. Nos diagramas de atividades, o comportamento está expresso fundamentalmente nos nós do diagrama. Cada nó representa um pedaço de comportamento. No diagrama de estados, ao contrário, todo o comportamento se encontra nos arcos do diagrama, sendo que os nós do diagrama de estados representa o que está nos arcos do diagrama de atividades, e os nós dos diagramas de atividades representam o que está nos arcos dos diagramas de estado. Assim, apesar de visualmente bastante similares, do ponto de vista semântico, o que é representado em cada diagrama é exatamente o oposto um do outro.

Em termos notacionais, já houve uma grande mudança nos diagramas de atividades e diagramas de estado, passando-se da norma UML 1 para o UML 2. As ações do diagrama de atividades no UML 1 tinha uma notação ligeiramente diferente da atual, que se faz muito mais próxima dos estados do diagrama de estados. Este autor supõe que no futuro, novas modificações poderão ocorrer nestes diagramas, para evitar ambiguidades.