Unified Modeling Language (UML)

Universidade Federal do Maranhão – UFMA
Pós Graduação de Engenharia de Eletricidade
Grupo de Computação
Assunto: Diagrama de Atividade

Autoria:Aristófanes Corrêa Silva
Adaptação: Alexandre César M de Oliveira

6 Diagrama de Atividade

6.1 Definição

- Capturam ações e seus resultados, focando o trabalho executado na implementação de uma operação (método), e suas atividades numa instância de um objeto.
- Trata-se de uma variação do diagrama de estado com um propósito um pouco diferente do diagrama de estado:

  - Capturar ações (trabalho e atividades que serão executados) e seus resultados em termos das mudanças de estados dos objetos.
  - Os estados no diagrama de atividade mudam para um próximo estágio quando uma ação é executada (sem necessidade de especificação de evento).
  - Outra diferença é que podem ser colocadas como "swimlanes" (agrupamento de atividades com respeito a quem é responsável e onde estas atividades residem na organização), representada por retângulos que englobam todos os objetos que estão ligados a ela.

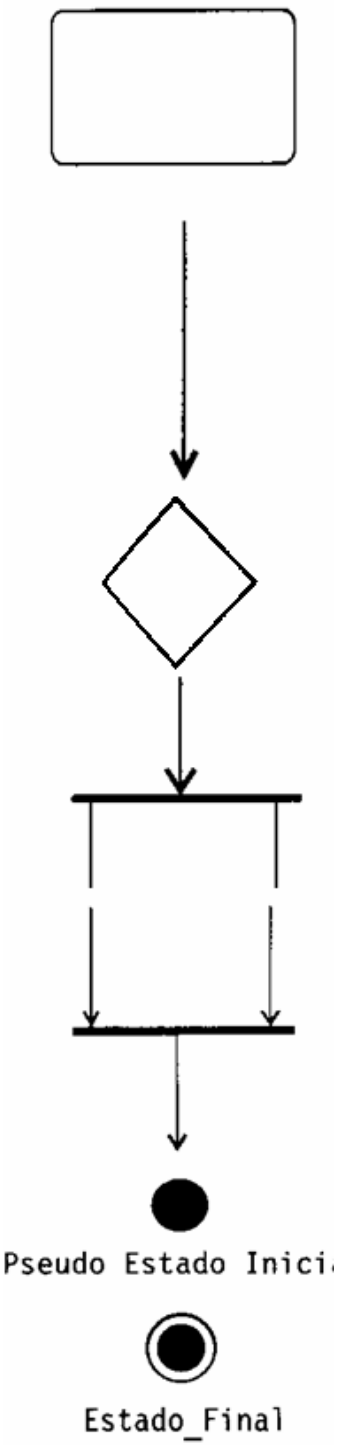
- Forma alternativa de se mostrar interações, com a possibilidade de expressar como as ações são executadas, o que elas fazem (mudanças dos estados dos objetos), quando elas são executadas (seqüência das ações), e onde elas acontecem (swimlanes).
- Mostra o fluxo seqüencial das atividades: atividades executadas por uma operação específica do sistema.
- Consistem em estados de ação, contendo a especificação de uma atividade a ser desempenhada por uma operação do sistema.
- Decisões e condições, como execução paralela, também podem ser mostradas na diagrama de atividade.
- O diagrama também pode conter especificações de mensagens enviadas e recebidas como partes de ações executadas.

6.2 Objetivos

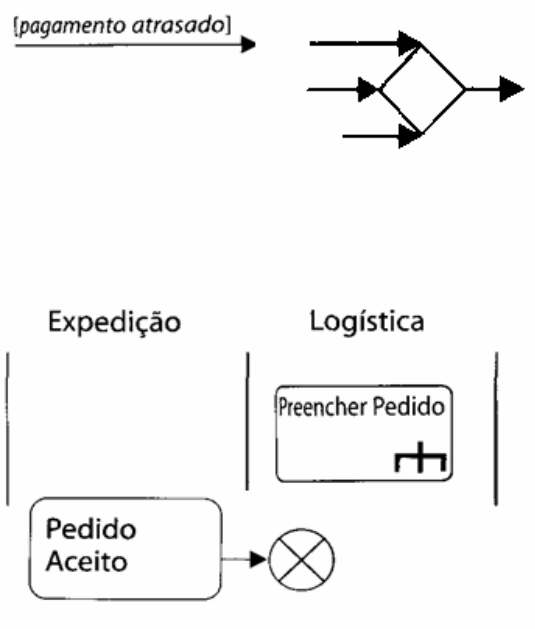
- Capturar as ações que serão executadas quando uma operação é disparada (uso comum) e o trabalho interno em um objeto.
- Mostrar como um grupo de ações relacionadas pode ser executado, e como elas vão afetar os objetos em torno delas.
- Mostrar como uma instância pode ser executada em termos de ações e objetos.
- Mostrar como um negócio funciona em termos de trabalhadores (atores), fluxos de trabalho, organização, e objetos (fatores físicos e intelectuais usados no negócio).

6.3 Notação

- **Atividade**: usado quando o usuário faz alguma coisa ou existe a resposta do sistema, pode ser usado este símbolo.
- **Passagens entre atividades:** (fluxo ou gatilho) Podem ser acrescentados efeitos e resultados.
- **Decisão**: Diversas saídas para o símbolo de decisão
- **Fork**: Significa que uma atividade chegou neste ponto e foi subdividida em mais de uma atividade
- **Join**: Significa que uma atividade chegou num mesmo ponto e criou-se uma nova atividade
- **Entrada/Saída**: Pode haver diversos pontos de saída para um processo

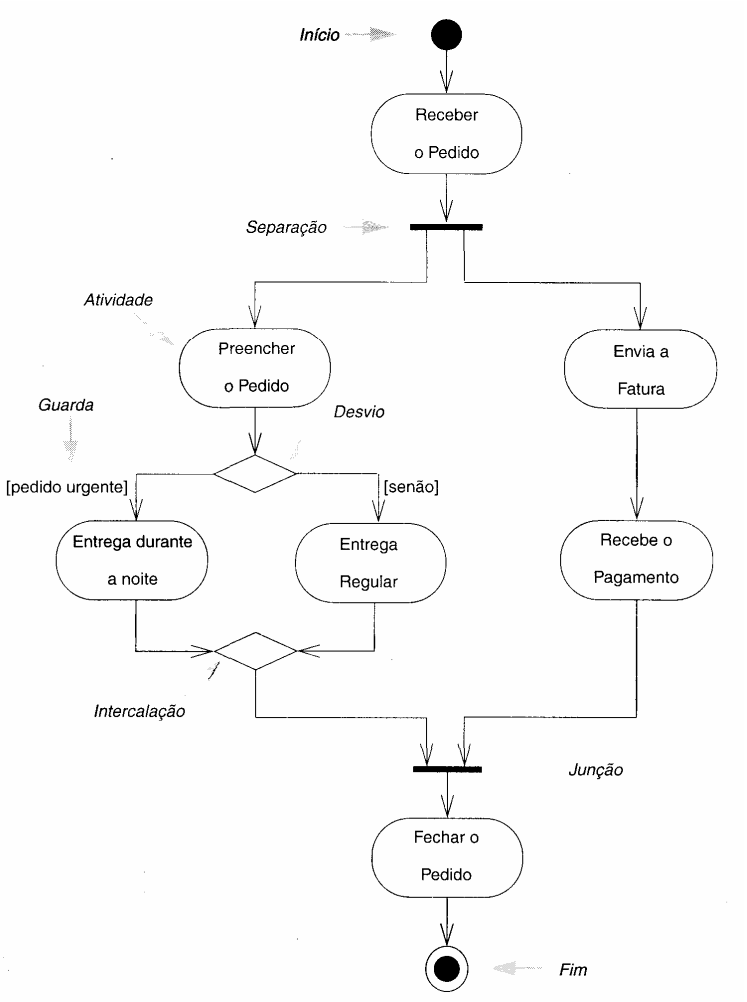


- **Merge**: Fluxos convergentes para um único ponto e existe apenas um saída, o que é diferente do join, onde vários fluxos chegam concorrentemente.
- **Condição**.
- **Swinmlanes.** Reproduz as raias de uma piscina
- **Rake** (rodo - tridente) dentro de uma determinada atividade indica que aquela atividade tem subatividades ou as está invocando
- **Fluxo final**. Indica que as atividades terminaram para aquela situação, ou caso de uso, e vão seguir em outro ponto, possivelmente. O final de um fluxo indica apenas que determinado fluxo de atividades se encerrou naquele ponto.



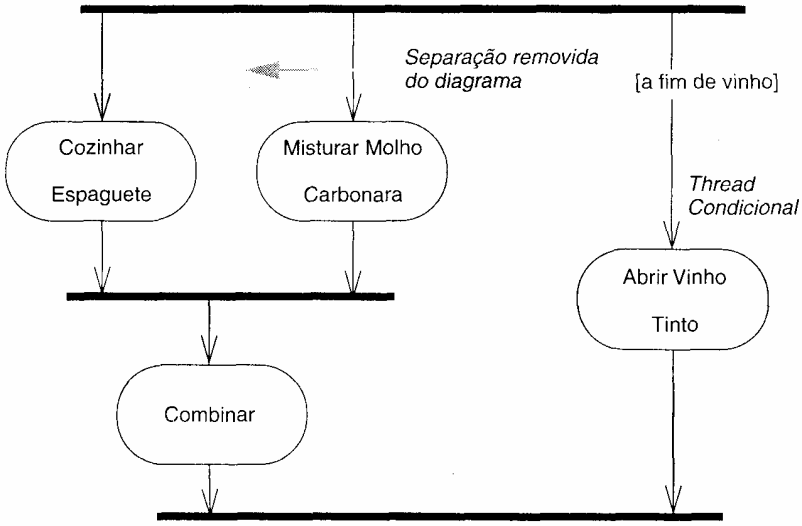
6.4 Comportamento Condicional

- Feito através de desvios e intercalações (*merges*)
- Um desvio é uma transição de entrada única e várias transições de saídas guardadas. Somente uma transição de saída pode ser tomada, de modo que os guardas (guards) devem ser mutuamente exclusivos.
- A utilização de [else] como um guarda indica que a transição "else" deverá ser usada se todos os outros guardas de desvios forem falsas.
- Uma intercalação tem múltiplas transições de entrada e uma única saída. Uma intercalação marca o final de um comportamento condicional iniciado por um desvio.



6.5 Comportamento Paralelo

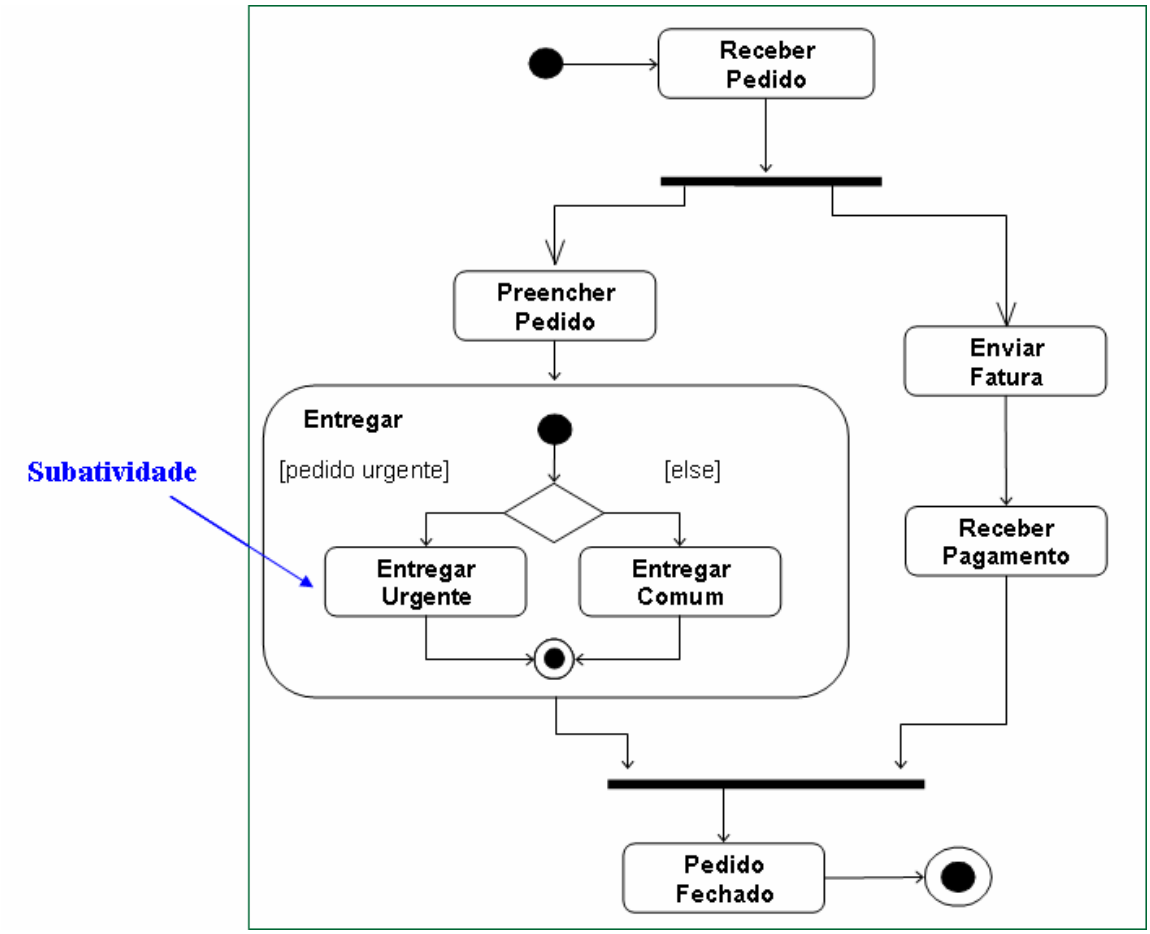
- O comportamento condicional é feito através de *Forks e Joins*
- Uma separação tem uma transição de entrada e várias transações de saída. Quando uma transição de entrada é acionada (triggered), todas as transições de saída são executadas em paralelo.
- Depois de uma separação e realização dos processo é necessário se efetuar a junção
- Separação e junção devem se completar. No caso mais simples, isso significa que todas vez que você tiver uma separação, deve ter uma junção que uma os threads iniciadas por aquelas separações



- **Thread condicional**: existe uma exceção para regra de que todos os estados de entrada em uma junção devem ter terminado suas atividades, antes que a junção possa ser efetuada. Você pode acrescentar uma condição para um thread saindo de uma separação.
- No exemplo, mesmo que você não esteja a fim de vinho, ainda assim seria capaz de comer o espaguete.

6.6 Decomposição

- Uma atividades pode ser dividida em subatividades.



6.7 Raias

- Permite que se documente o que acontece e quem faz acontecer.
- Permite que, em modelagem de domínio, o diagrama de atividade represente pessoas ou departamentos responsáveis por cada atividade.
- Para usar raias, você deve organizar seus diagramas de atividades em zonas verticais separadas por linhas. Cada zona representa as responsabilidades de uma classe específica ou um depto específico.
- Combinam a descrição de lógica do diagrama de atividades com a descrição de responsabilidade do diagrama de interação.
- Podem ser difíceis de serem projetadas em um diagrama complexo.