Modelo de Casos de Uso (MUC)

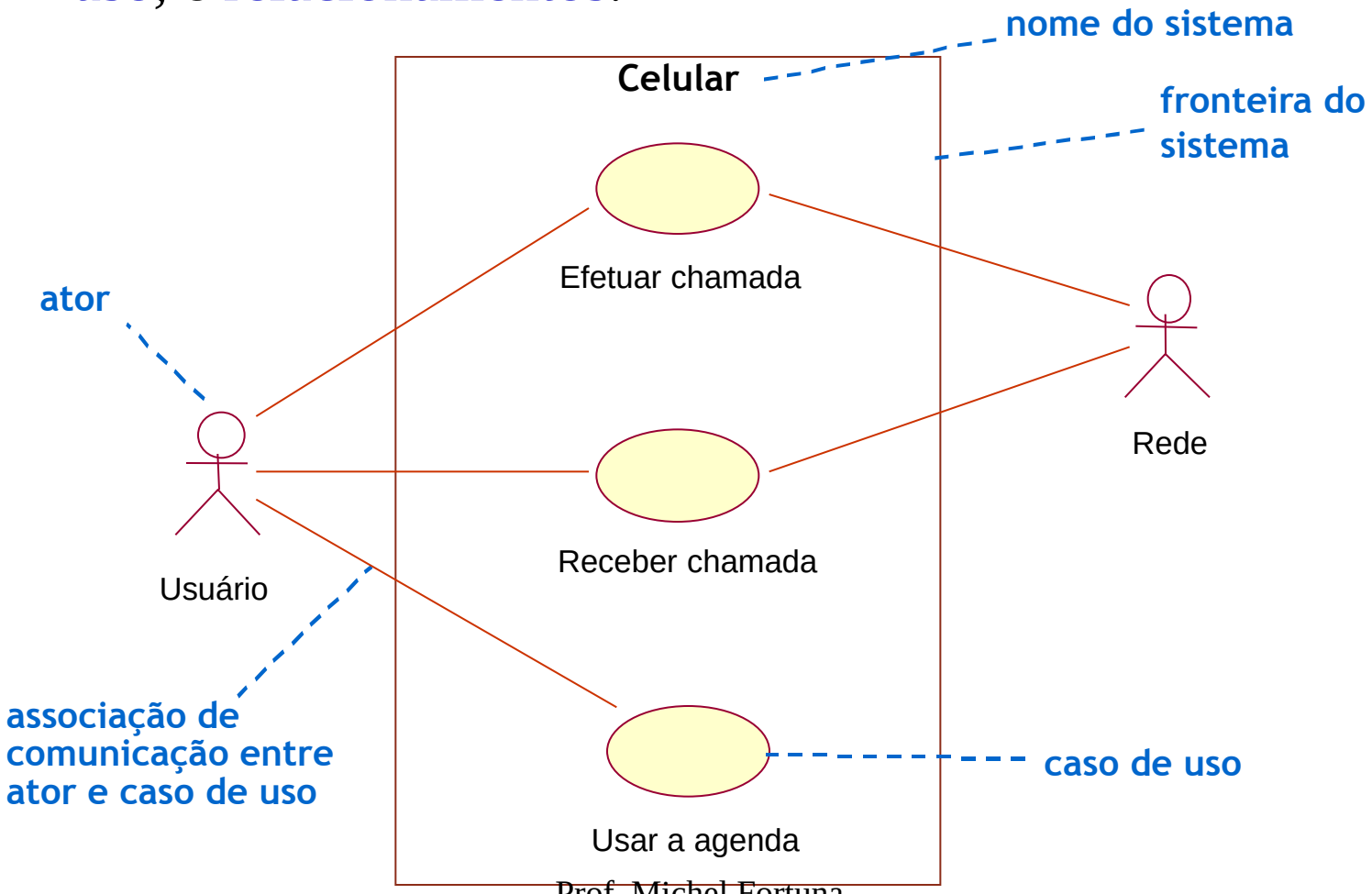
- Serve para a especificação dos requisitos funcionais de um sistema, ou seja:
  - O que o sistema deve fazer;
  - Que funções os atores poderão executar com o uso do sistema (casos de uso).

- É composto de 3 partes:

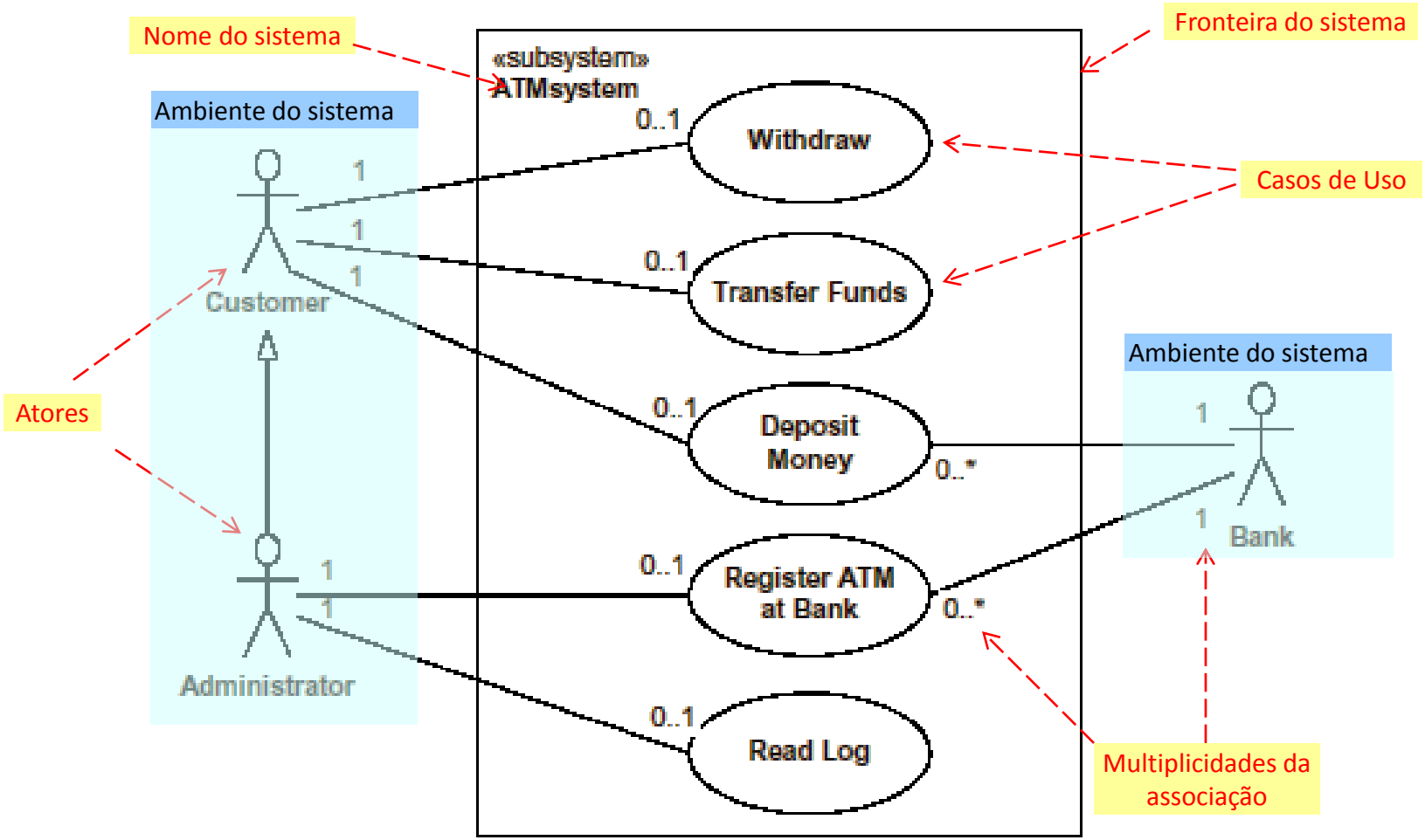
- Diagrama de Casos de Uso
- Glossário de Termos
- Descrição dos Casos de Uso

Diagrama de UCs

- Elementos do diagrama de UCs: sistema, atores, casos de uso, e relacionamentos.



Outro Exemplo de Diagrama de UCs



Sistema de terminal eletrônico de banco

Como Descobrir (e Nomear) os Atores

- **Atores** são usuários e quaisquer outros sistemas (de HW ou SW) que poderão interagir (trocando sinais e dados) com o sistema que se está modelando. Representam o ambiente do sistema.

- Para **encontrar os atores** identifique as pessoas que utilizarão o sistema, e então, identifique os papéis que elas desempenham. Esses papéis são os atores.
  - Exemplo: **José** (pessoa) × **Gerente** (papel de José, no contexto do sistema)

- O **nome do ator** deve descrever o papel que ele desempenha no sistema ou, dito de outra forma, descrever as responsabilidades do ator perante o sistema.
  No exemplo acima, o nome do ator seria Gerente (e não José)

Exercício

Para os sistemas abaixo indicados, nomeie seus atores:

- Sistema de gerência de um restaurante
- Sistema de PDV (Ponto-de-Venda) (exemplo: caixa de supermercado)

Como Nomear os UCs

- Use **nomes ativos**, ou seja, que implicam uma ação, como por exemplo, “Abrir pedido” ou “Emitir extrato” (em vez dos nomes passivos: “Abertura de pedido” e “Emissão de extrato”, respectivamente).
- Nomes ativos implicam algo sendo feito, e reforçam a idéia de que UCs fazem algo útil para o ator.

Como Descobrir os UCs

Todo UC deve satisfazer os seguintes critérios:

Critério 1: Permitir que algum ator ou *stakeholder*[1] alcance, através do sistema, um objetivo seu

- Traduzir valor para o ator (sentimento de "**missão cumprida**").

Critério 2: Deixar o sistema em um estado estável

- **Estado estável**: estado em que fica descartada qualquer necessidade de retorno (*rollback*) a um estado anterior (estado consistente, que pode persistir indefinidamente);
- ***Traduz uma condição de suficiência***: apenas estados estavéis são relevantes.

Critério 3: Precisar de um único ***evento autônomo*** para ser ativado e completar a sua execução

- **Evento autônomo:** evento externo (gerado fora do sistema, pelos atores), que:
  1) É um *evento inicial*[2] do sistema; ou
  2) É um evento para o qual não se pode embutir (no código do) sistema qualquer suposição sobre que evento autônomo o antecede imediatamente.
- ***Traduz uma condição de completude***: todos os estados estáveis são relevantes.

[1] ***Stakeholder***: toda pessoa que tem algum tipo de interesse nos resultados do sistema. Ator é um *stakeholder* que interage com o sistema.
[2] Um evento externo é **inicial** em um sistema quando ele pode ser o primeiro evento disparado logo após a ativação do sistema.

Exemplo de Utilização dos Critérios

Os próximos slides mostram como utilizar os critérios anteriores para decidir quais são os UCs de um sistema. Para isso, são tomados como exemplo dois sistemas: um sistema de caixa de supermercado (sistema *PDV* – Ponto de Venda) e outro de controle de refeições em um restaurante (sistema *Restaurante*).

Esses sistemas tem as seguintes funções (entre outras):

- ***Registrar*** *(*itens da) ***compra*** e ***Pagar conta*** (no sistema *PDV*)
- ***Registrar pedido*** (de refeição) e ***Pagar conta*** (no sistema *Restaurante*)

Cada uma dessas duas funções em um sistema é semelhante à correspondente função no outro sistema. Inicialmente, vamos pressupor que cada função constitui seu próprio UC. O objetivo desse exemplo é mostrar, com base na análise dos critérios anteriores, que isso é correto apenas no sistema *Restaurante;* no sistema *PDV*, o correto seria colocar as suas duas funções acima em um único UC.

Acompanhe o desenvolvimento dessa análise nos slides a seguir, lendo os textos na ordem dada pela numeração indicativa (1º, 2º, ...).

Obs: Existe também uma versão mais dinâmica (e, portanto, mais didática) desses slides no site Moodle da disciplina (utilize o *MS Powerpoint* ou o *Powerpoint Viewer*, para uma melhor visualização dos seus elementos dinâmicos).

Efeitos da Utilização dos Critérios (1)

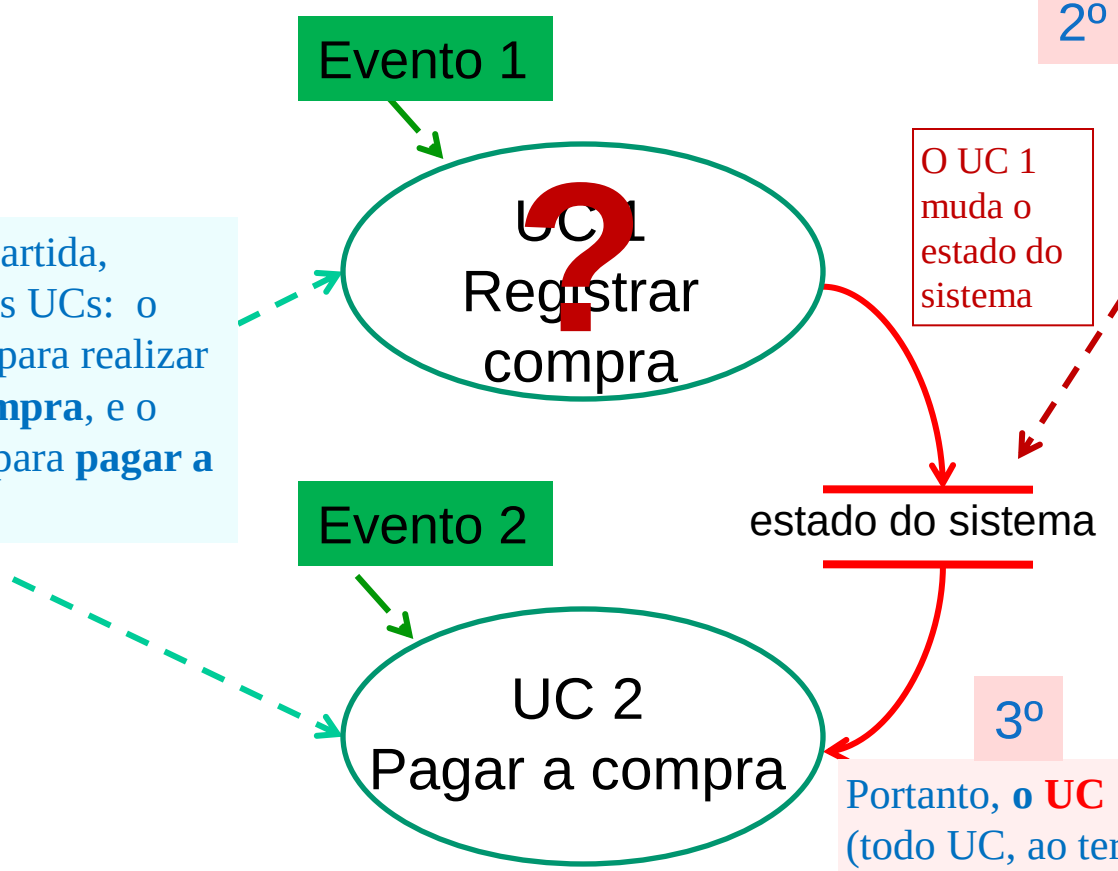
Sistema de Ponto-de-Venda (supermercado)

Funções: 1) Registro de uma compra;
2) Pagamento da compra.

Quantos UCs devemos ter? **Dois?**

1º

Como ponto de partida, vamos adotar dois UCs: o primeiro (**UC 1**) para realizar o **registro da compra**, e o segundo (**UC 2**) para **pagar a compra**.



2º Mas o **estado** em que o UC 1 deixa o sistema **não é estável**, pois se a compra não for paga, o registro da mesma precisa ser desfeito (já que os itens voltam para a prateleira do supermercado). Ou seja, o sistema deve retornar ao estado anterior ao registro.

3º

Portanto, **o UC 1 infringe o Critério 2** (todo UC, ao terminar, deve deixar o sistema em um estado estável).

Efeitos da Utilização dos Critérios (1)

Sistema de Ponto-de-Venda (supermercado)

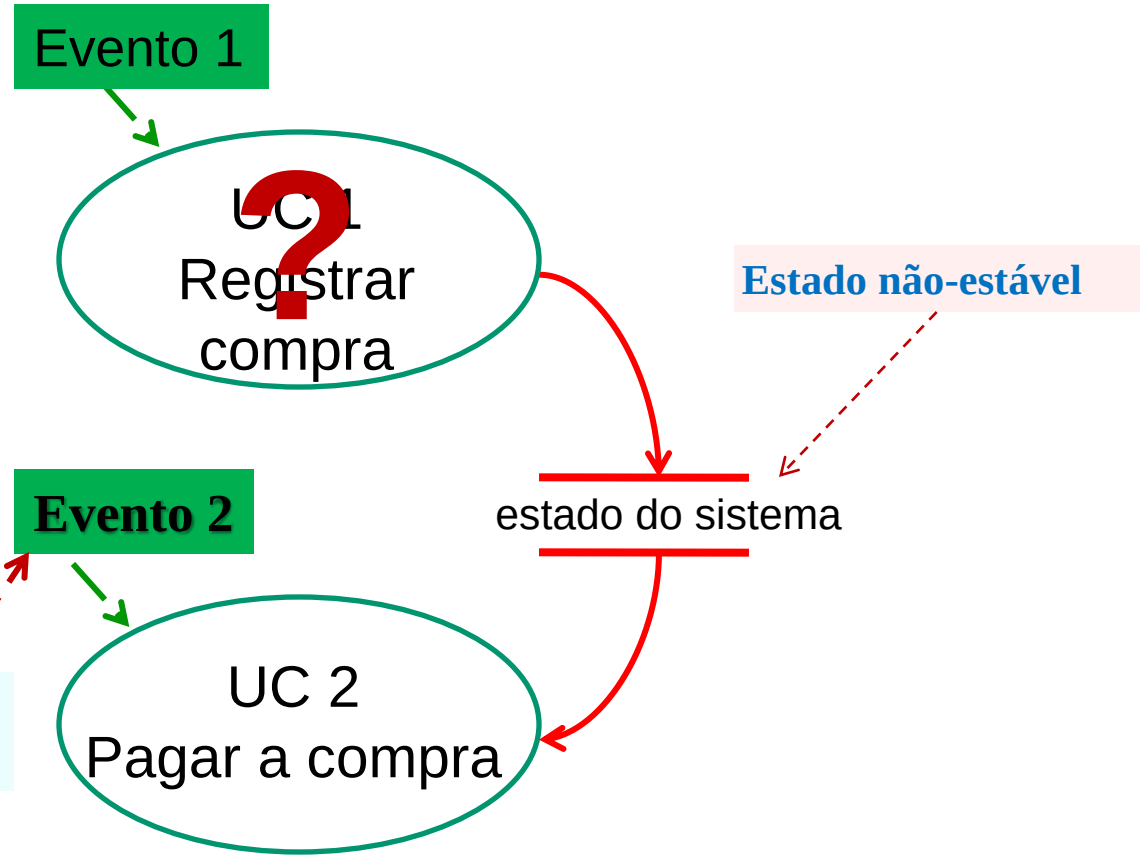
Funções: 1) Registro de uma compra;
2) Pagamento da compra.

Quantos UCs devemos ter? **Dois?**

4º

Uma análise do **Evento 2** mostra que, embora o momento exato da sua ocorrência dependa da decisão do ator, ele deve, obrigatoriamente, ocorrer logo após o primeiro evento.

Ou seja, **ele é um evento externo, mas não autônomo**.



Efeitos da Utilização dos Critérios (1)

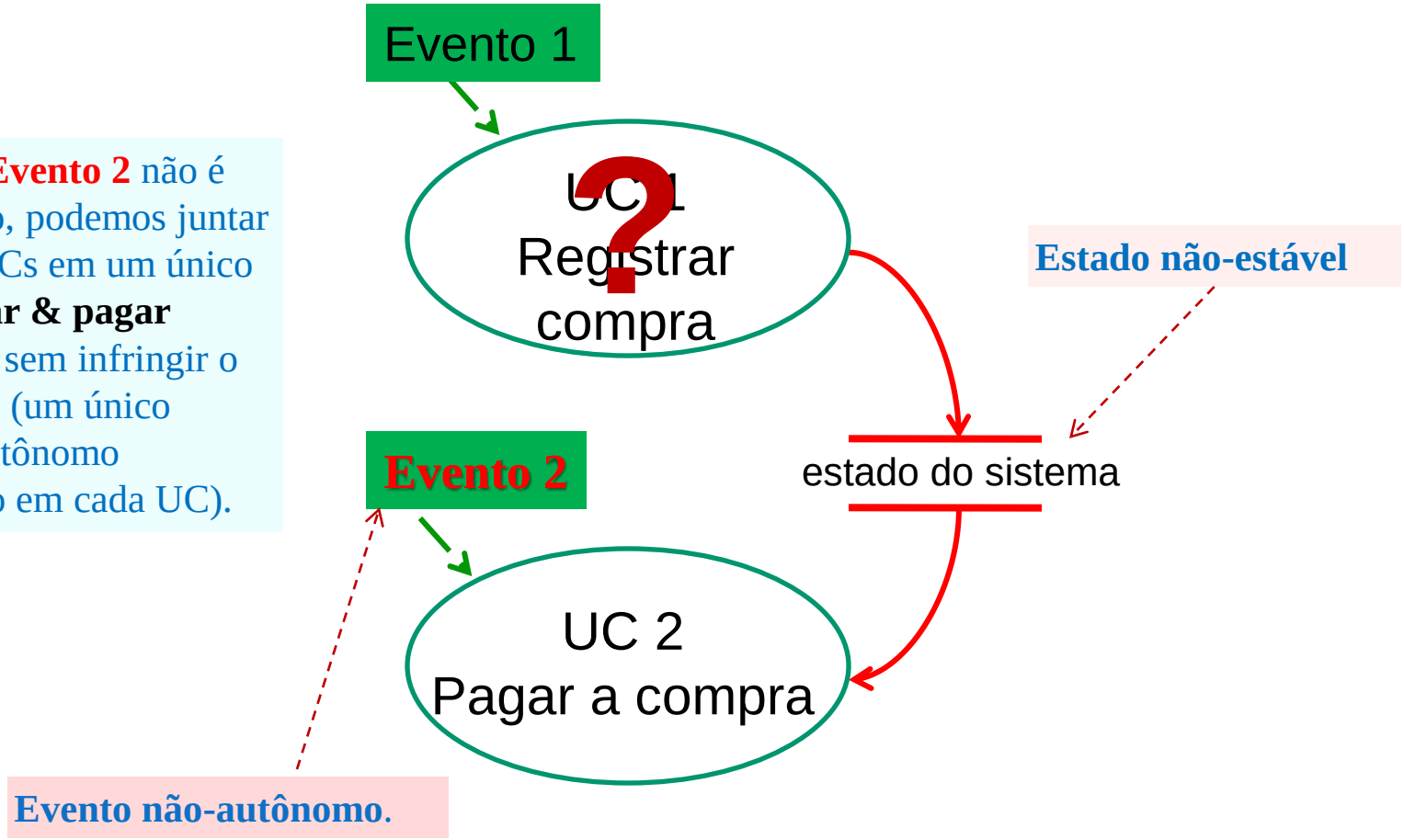
Sistema de Ponto-de-Venda (supermercado)

Funções: 1) Registro de uma compra;
2) Pagamento da compra.

Quantos UCs devemos ter? **Dois?**

5º

Como o **Evento 2** não é autônomo, podemos juntar os dois UCs em um único (**Registrar & pagar refeição**) sem infringir o Critério 3 (um único evento autônomo envolvido em cada UC).



Efeitos da Utilização dos Critérios (1)

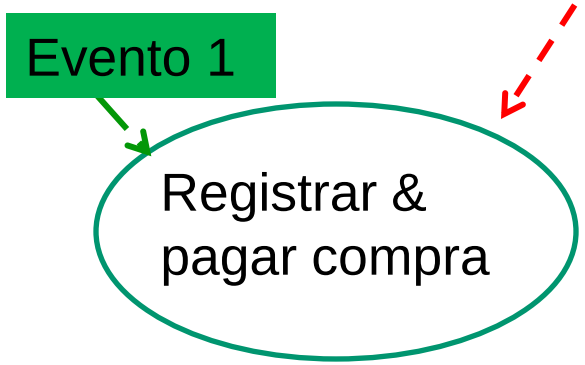
Sistema de Ponto-de-Venda

Funções: 1) Registro de uma compra;
2) Pagamento da compra.

Quantos UCs devemos ter?

5º

Como o **Evento 2** não é autônomo, podemos juntar os dois UCs em um único (**Registrar & pagar refeição**) sem infringir o Critério 3 (um único evento autônomo envolvido em cada UC).



6º

Este novo UC é válido, pois:
Tem um único evento autônomo envolvido nele (**Critério 3**);
Deixa o sistema em um estado estável (**Critério 2**);
Representa o alcance de um objetivo do ator (**Critério 1**).

Portanto...
no contexto do sistema PDV:

Apenas 1 UC !!

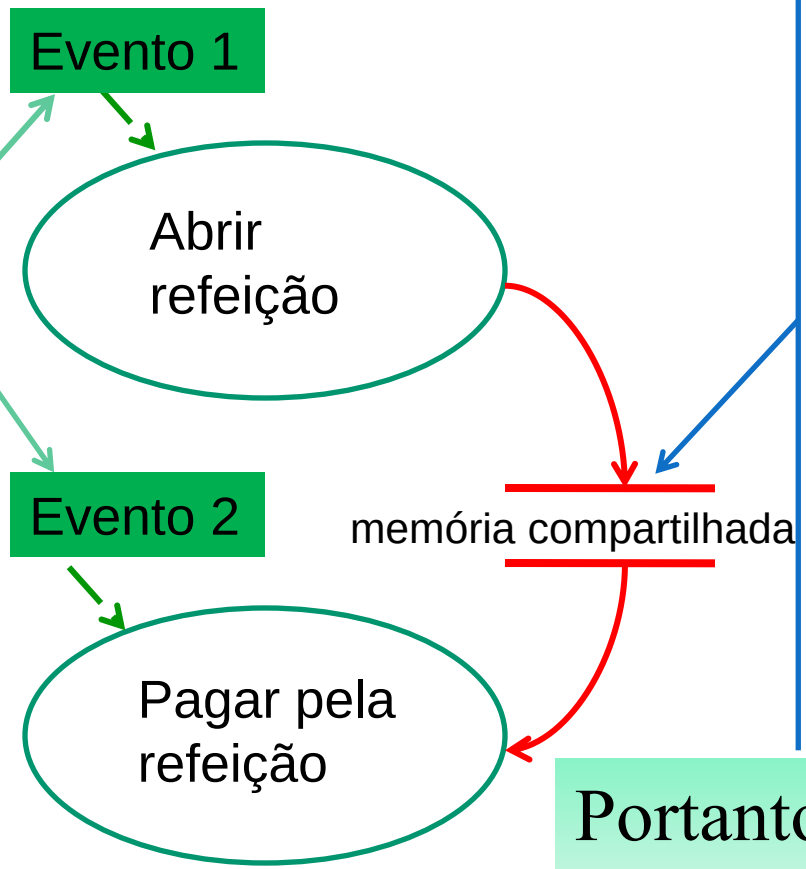
Efeitos da Utilização dos Critérios (2)

Sistema Restaurante

Funções: 1) Abertura de refeição;
2) Pagamento de refeição.

Quantos UCs devemos ter? **Dois?**

Eventos autônomos, pois podem ocorrer em qualquer momento e em qualquer ordem (p. ex., após abrir uma refeição, o ator pode abrir outra antes de pagar a primeira)



Estado estável, pois o registro da refeição deve ser mantido mesmo se o pagamento não ocorrer (p. ex., para que sistema tenha o registro do consumo). Ou seja, nenhuma outra ação no sistema é necessária para que o estado do sistema, resultante da abertura da refeição, possa persistir (ser mantido) indefinidadmente.

Como os dois eventos são autônomos, a junção dos UCs infrigiria o Critério 3 (um único evento autônomo envolvido em cada UC).

Portanto…
2 UCs!

Exercício

Para os sistemas Restaurante e PDV, apresente os UCs escolhidos segundo os 3 critérios vistos