Métodos de Desenvolvimento de Software (MDS) 2011/2012

Miguel Goulão
mgoul@fct.unl.pt
http://ctp.di.fct.unl.pt/~mgoul/

Aula 12

Diagramas de Interacção

Motivação

Continuando o nosso processo de análise...

Recapitulando

- Estivemos a estudar:
  - Casos de uso
  - Diagramas de actividades
  - Diagramas de classes do domínio
  - OCL, para formalizar regras de negócio (incluindo restrições de integridade sobre os nossos diagramas)
- Agora, vamos revistitar e analizar em maior detalhe os casos de uso
  - **Realização de casos de uso**

Objectivos da realização de casos de uso

- Descobrir que classes de análise interagem para realizar o comportamento descrito nos casos de uso
- Descobrir que instâncias de mensagens dessas classes têm de ser trocadas entre as classes para produzir um determinado comportamento
  - Quais as operações principais das classes de análise?
  - Quais os atributos principais das classes de análise?
  - Quais as principais relações entre as classes de análise?
- Se necessário, **actualizar** o modelo de casos de uso, requisitos, e modelo de classes de análise
  - **É fundamental** **manter os vários modelos consistentes**

Realização de Casos de Uso

- Conjunto de classes que interagem para realizar o comportamento do caso de uso
  - Exº: Num caso de Uso Emprestar Livro, o actor Bibliotecário pode interagir com as classes de análise Livro, Registo de empréstimo e Utente para realizar o caso de uso

Especificação de requisitos funcionais	Especificação de alto nível do sistema
Caso de Uso	Diagrama de classes de análise
	Diagrama de interacção
	Requisitos especiais
	Caso de uso refinado

Realização do caso de uso

O método que estamos a seguir é **iterativo**

- Diagramas de classes de análise “contam uma história” sobre como as classes se relacionam para que as suas instâncias colaborem na construção do comportamento do caso de uso
- Diagramas de interacção demonstram como instânicas dessas classes colaboram para realizar o comportamento
- Novos requisitos podem ser descobertos
- Os próprios casos de uso podem ser refinados

7

Diagramas de interacção

FCT FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Interacção

- Unidade de comportamento de um classificador
  - O classificador fornece o contexto da interacção
  - Na realização de casos de uso, o classificador que dá contexto é o caso de uso
  - À medida que detalhamos uma interacção, é natural que encontremos novas operações e atributos das classes de análise
    - Portanto, temos de actualizar os diagramas de classes!

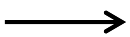
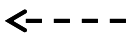
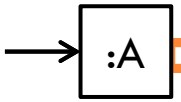
Elementos principais numa interacção

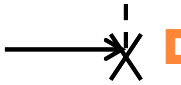
- Lifeline – participante na interacção
  - Nome – usado para referir a lifeline
  - Tipo – nome do classificador que a lifeline representa
  - Selector – expressão booleana que especifica um participante em particular (se omitido, o participante pode ser qualquer instância arbitrária do tipo)
- Representa **como** uma instância do classificador participa na interacção

Elementos principais numa interacção

- Mensagens – representam a comunicação entre duas lifelines de uma interacção
  - Envio de mensagens
  - Criação e destruição de instâncias
  - Envio de sinal

Mensagens

- Mensagens podem ser
  -  Síncronas (quem envia espera resposta)
  -  Assíncronas (quem envia não espera resposta)
  -  De retorno (devolução de controlo a anterior emissor)
  - 

De criação (de uma nova lifeline)
  -  De destrução (de uma lifeline)

-  Encontradas (origem desconhecida)
-  Perdidas (destino desconhecido)

Diagramas de interacção

- Mostram comunicação entre objectos
- Objectivo:
  - **especificar a realização de um use case**
  - especificar a realização de uma operação
- Dois tipos:
  - Diagramas de sequência
  - Diagramas de colaboração

Diagramas de sequência e de colaboração

- Ambos especificam a mesma informação
- Cada um foca aspectos diferentes
  - **Diagrama de sequência é "orientado" no tempo**
    - mostra graficamente a ordem das mensagens
    - não mostra como se obtém o objecto receptor
  - **Diagrama de colaboração é "orientado" no espaço**
    - mostra relações estáticas e dinâmicas entre objectos
    - a ordem das mensagens é dada explicitamente
    - tempo não é uma dimensão

Diagramas de sequência

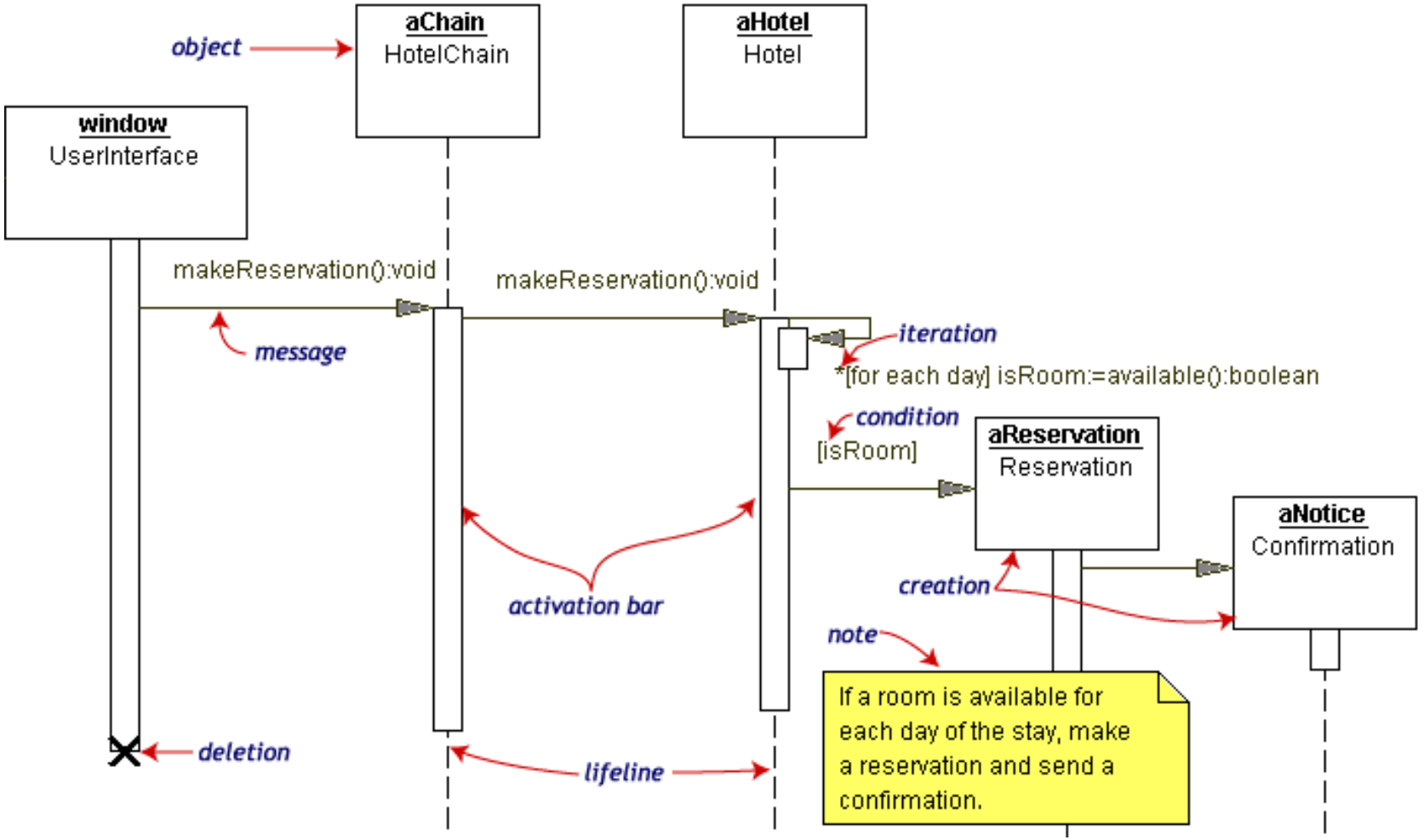
Diagramas de sequência

- Mostram a comunicação (troca de mensagens) entre objectos necessária para a execução de um caso de uso
- Fonte: informação disponível nos use cases e noutros modelos disponíveis (por exemplo, diagramas de actividade).
- Como um use case descreve todas as vertentes de uma dada funcionalidade (incluindo casos de erro e excepção), podemos optar por construir um diagrama de sequência por cenário.

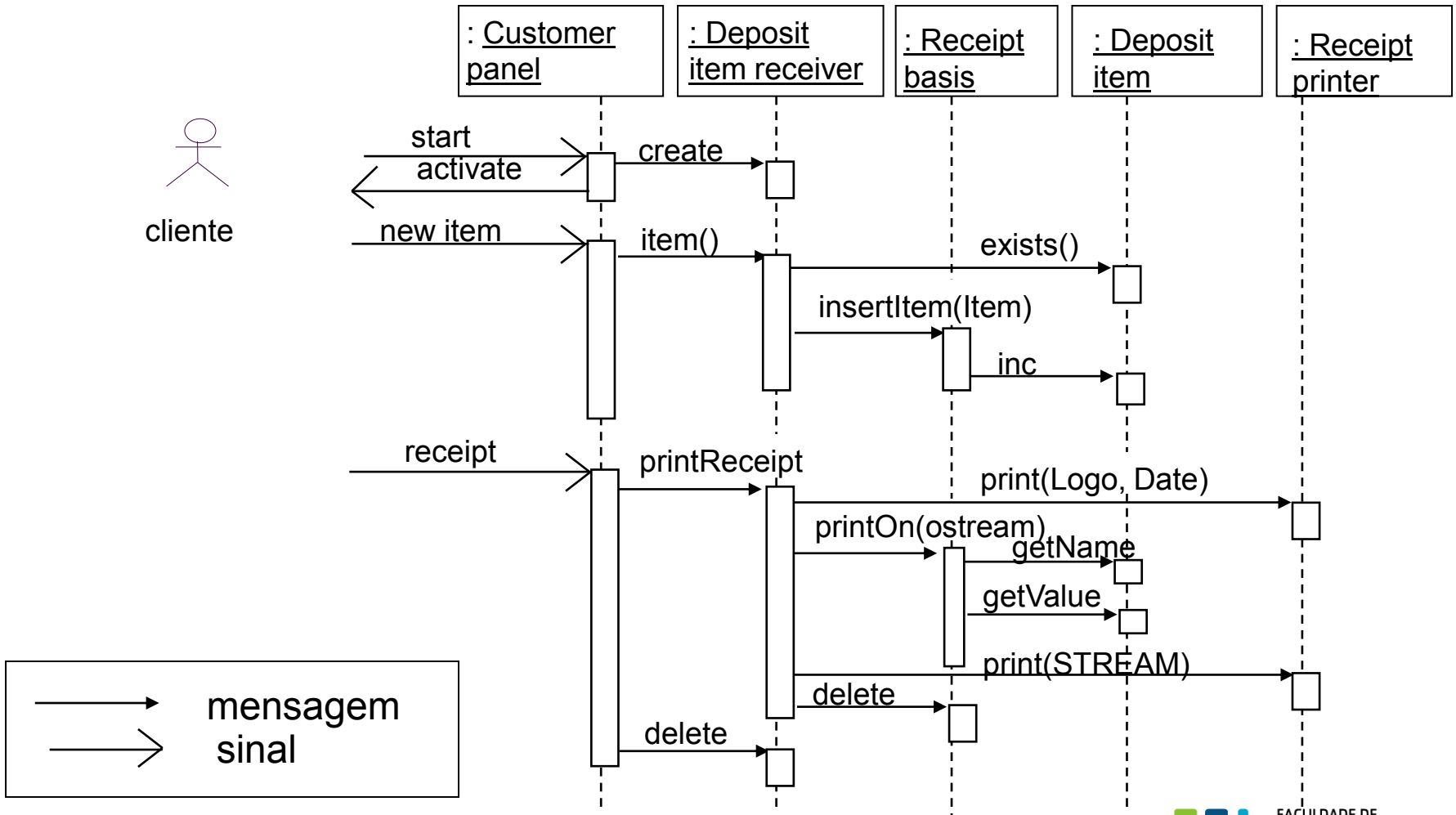
Diagramas de sequência: notação

- Objectos (rectângulos) organizados ao longo do eixo X
- Linha de vida de um objecto: linha tracejada que representa a existência de um objecto num período de tempo
- Mensagens (setas), ordenadas temporalmente, no eixo Y
- Fluxo de controlo execução: rectângulo estreito que mostra o período de tempo desde que o objecto recebe a mensagem até que fornece uma resposta (tempo de execução de uma operação);
  - este tempo pode incluir tempos de execução de operações subordinadas.

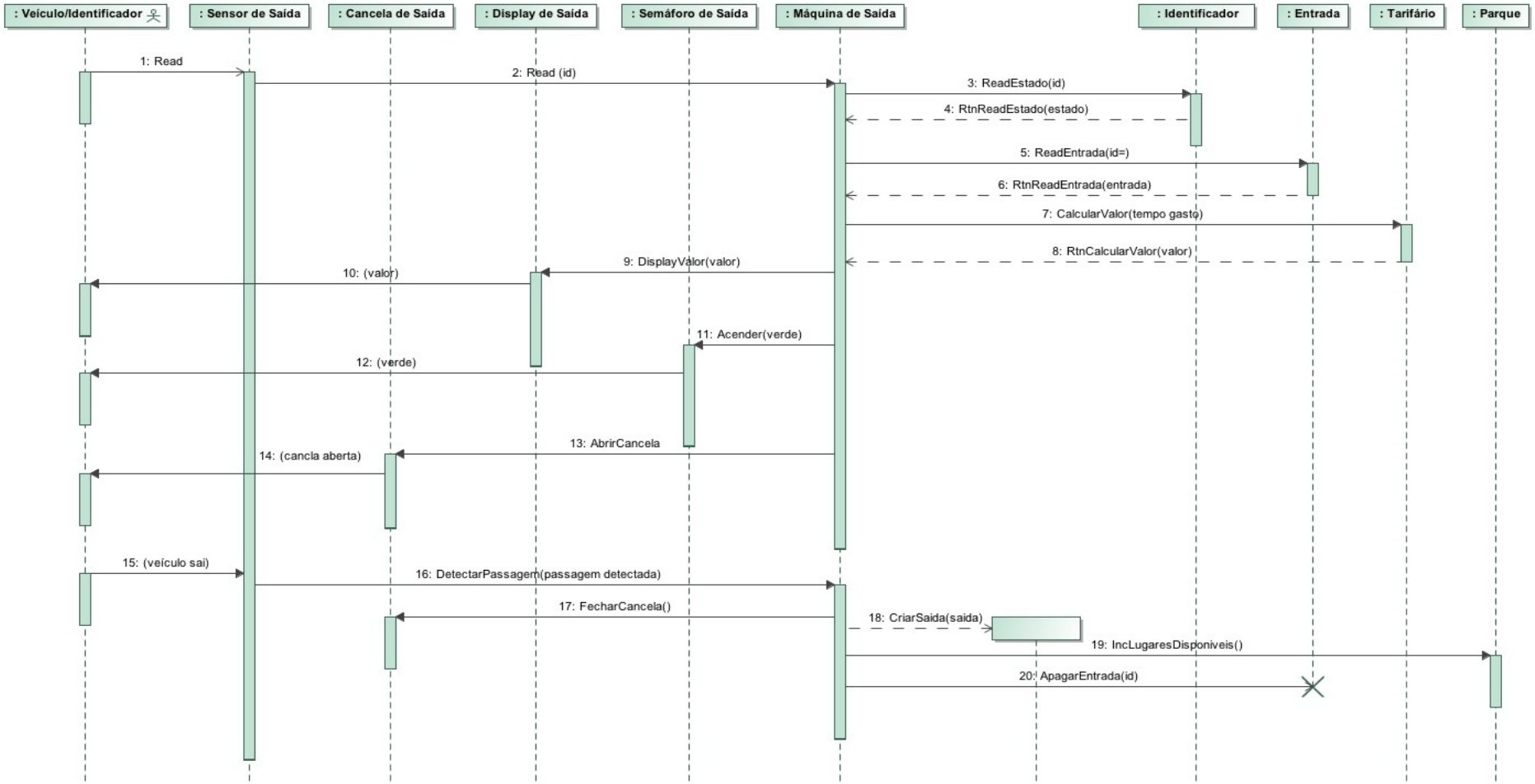
Diagrama sequência: reserva quarto de hotel



Exemplo: máquina de reciclagem



DS para o cenário “veículo autorizado sai do parque”

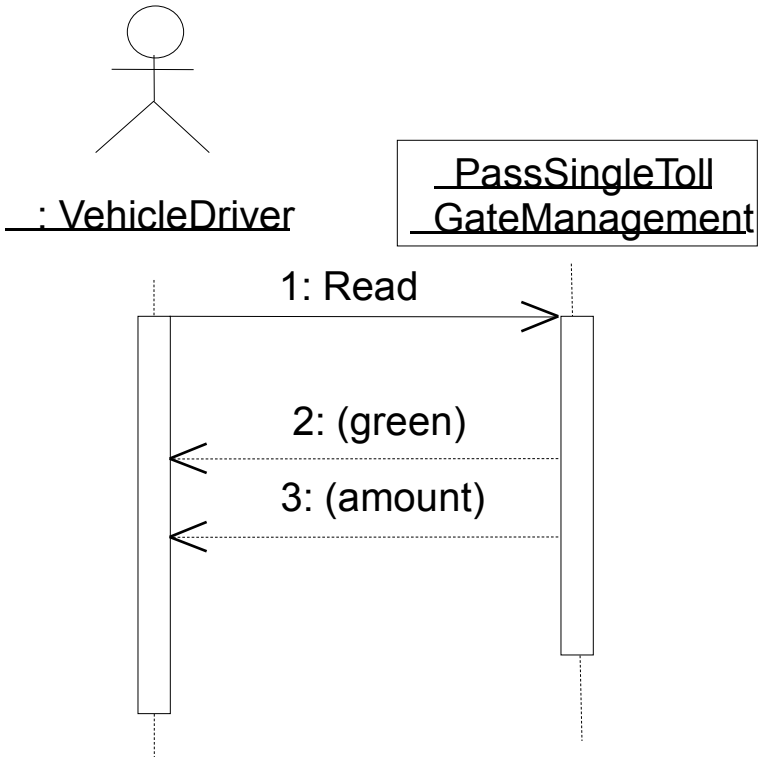


Problemas na construção do DS

- Nem sempre é fácil construir um DS
  - Objectos de controlo estão envolvidos sempre que detectamos que um objecto de interface lidera/controla a execução;
  - O DS pode construir-se por partes, usando os três tipos de objectos como "guia" de construção.
- Algumas regras de construção:
  - A primeira mensagem é sempre enviada por um actor
  - A primeira mensagem é sempre recebida por um objecto de interface
  - Adicionar um objecto de controlo sempre que o objecto de interface se torne o "decisor"

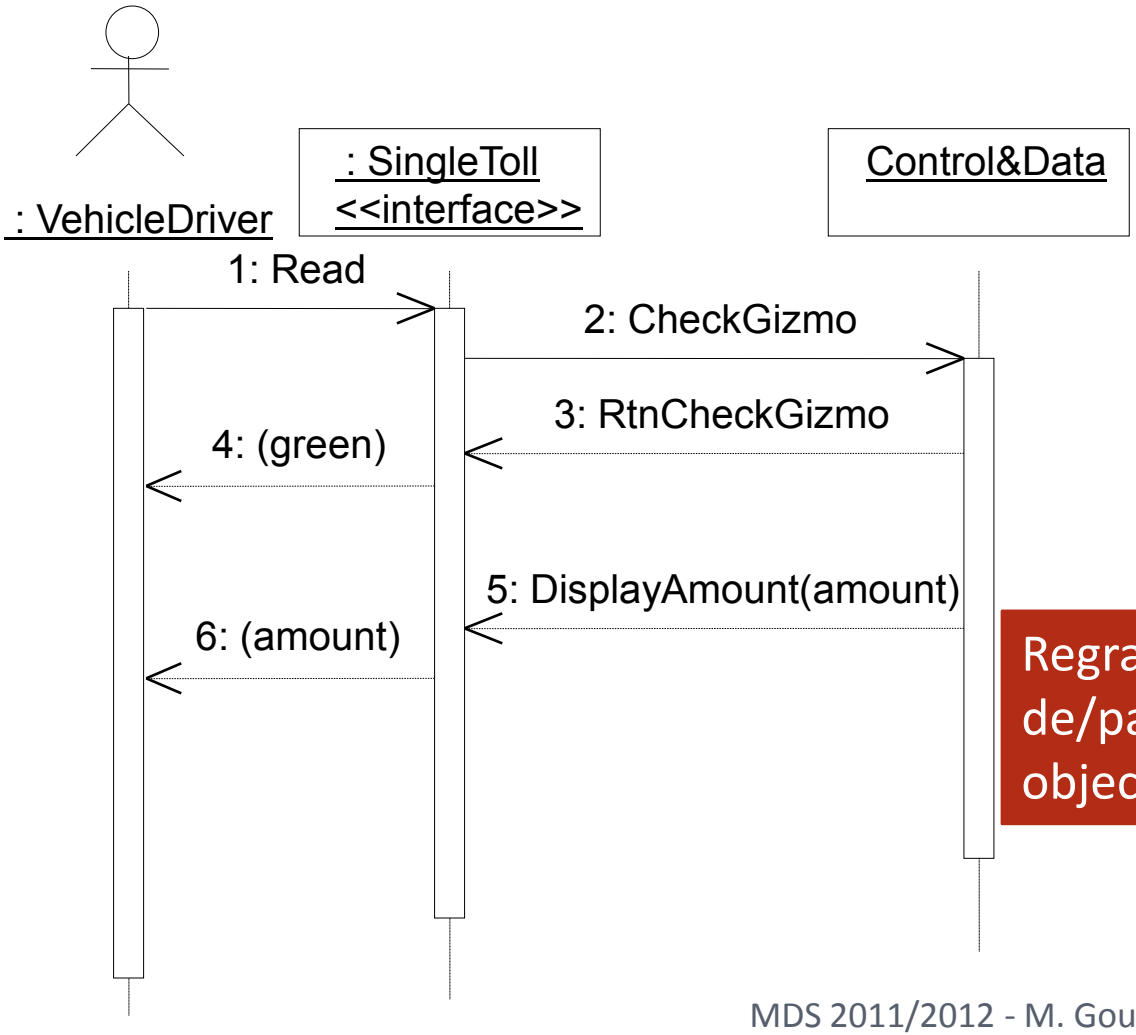
DS, passo a passo

Exemplo: veículo passa numa portagem de ponto único (e.g., ponte 25 Abril).



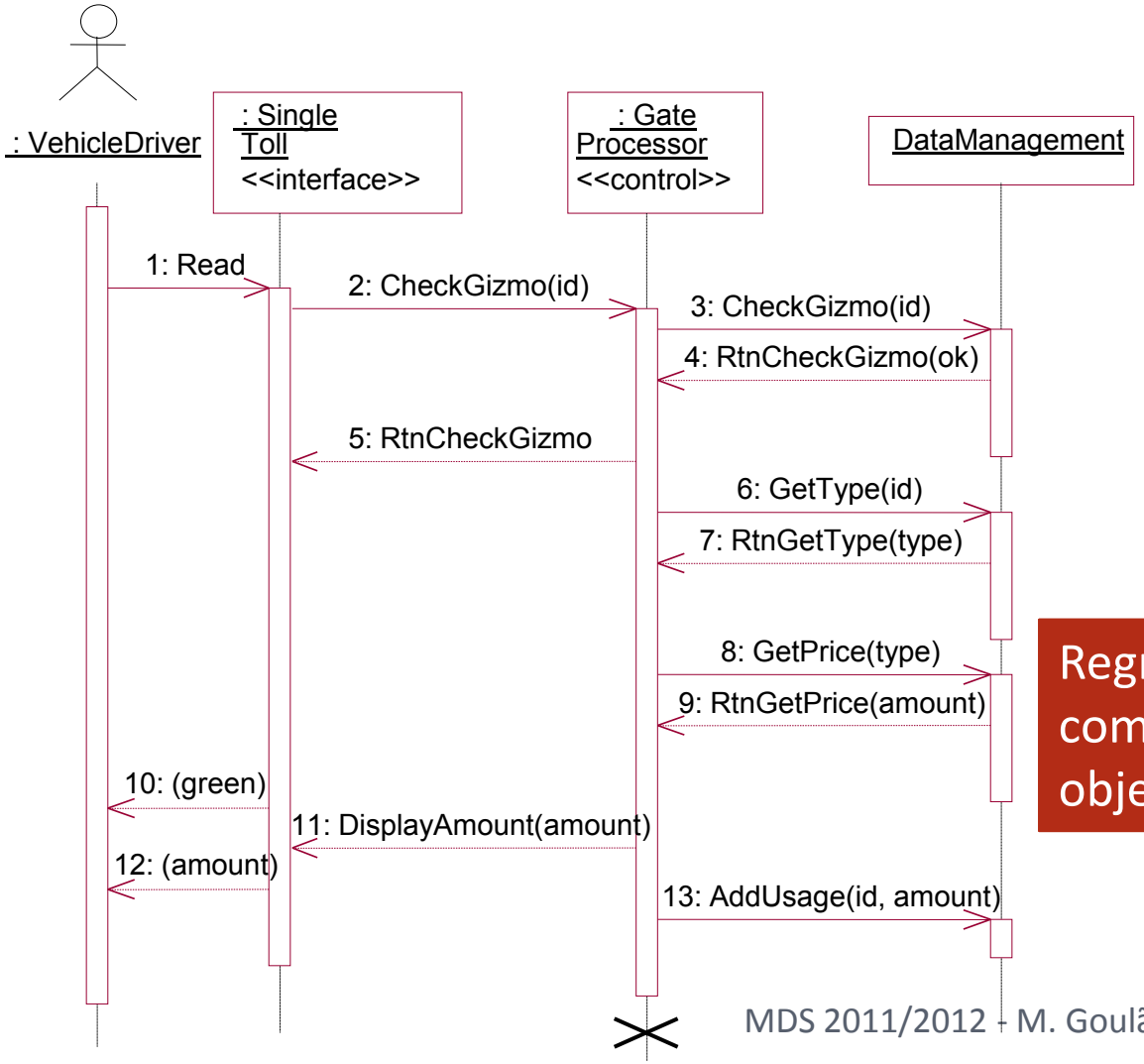
Regra 1: ao iniciar, o sistema é visto como uma caixa preta.

DS com um objecto de interface



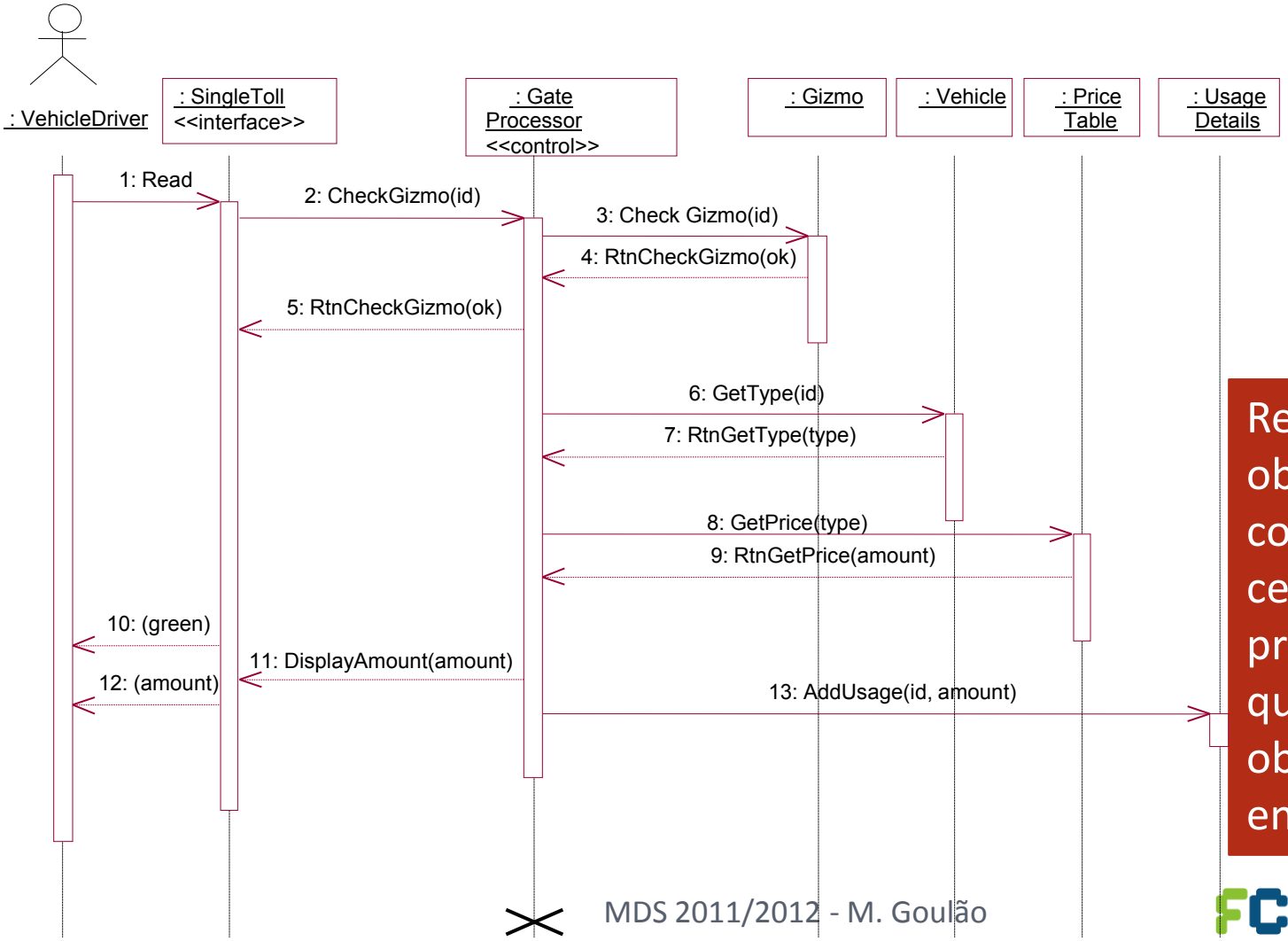
Regra 2: Todas as mensagens de/para actores passam por objectos de interface.

DS com objectos de interface e controlo



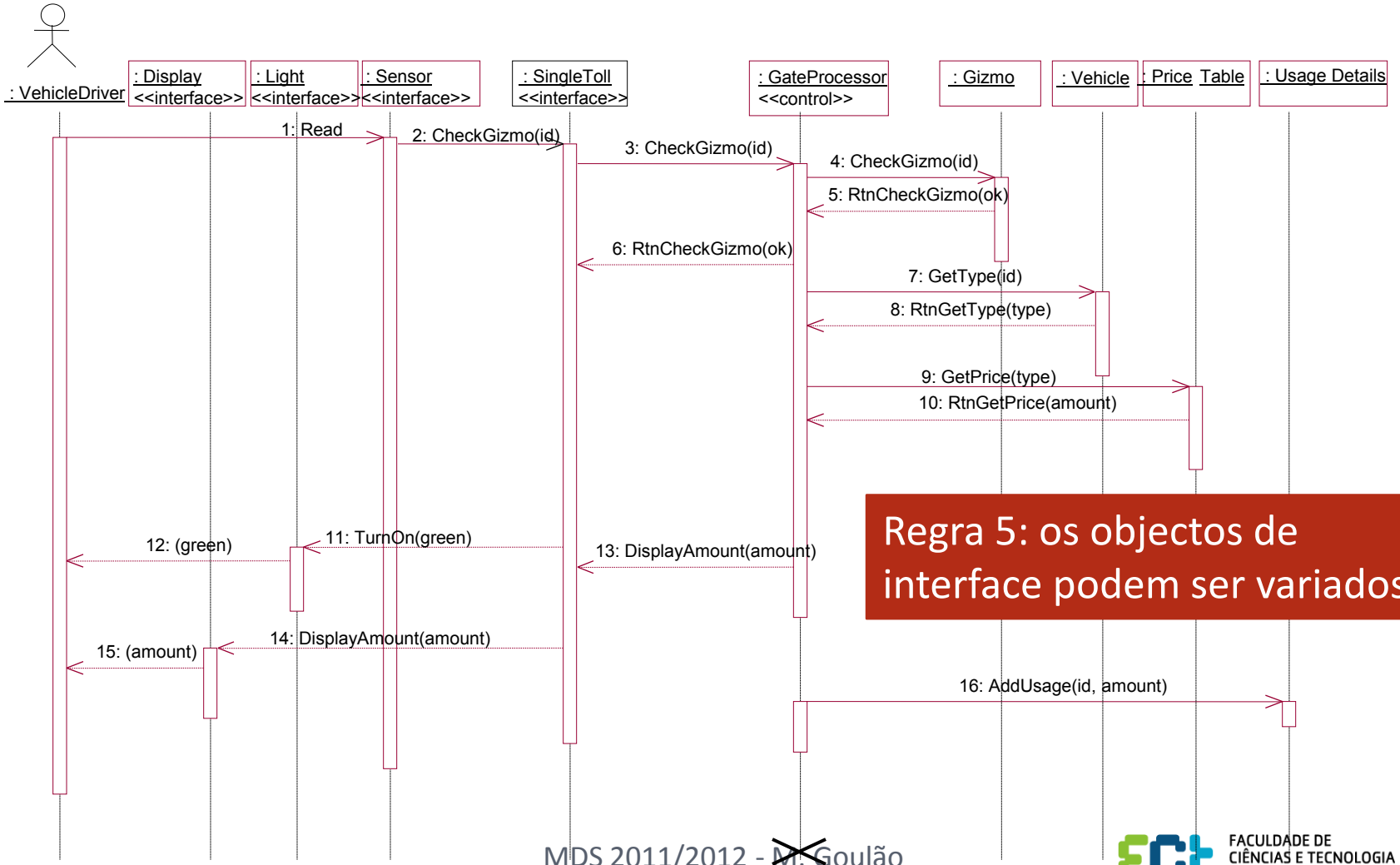
Regra 3: para funcionalidades complexas precisamos de objectos de controlo.

DS com objectos de interface, controlo e entidade



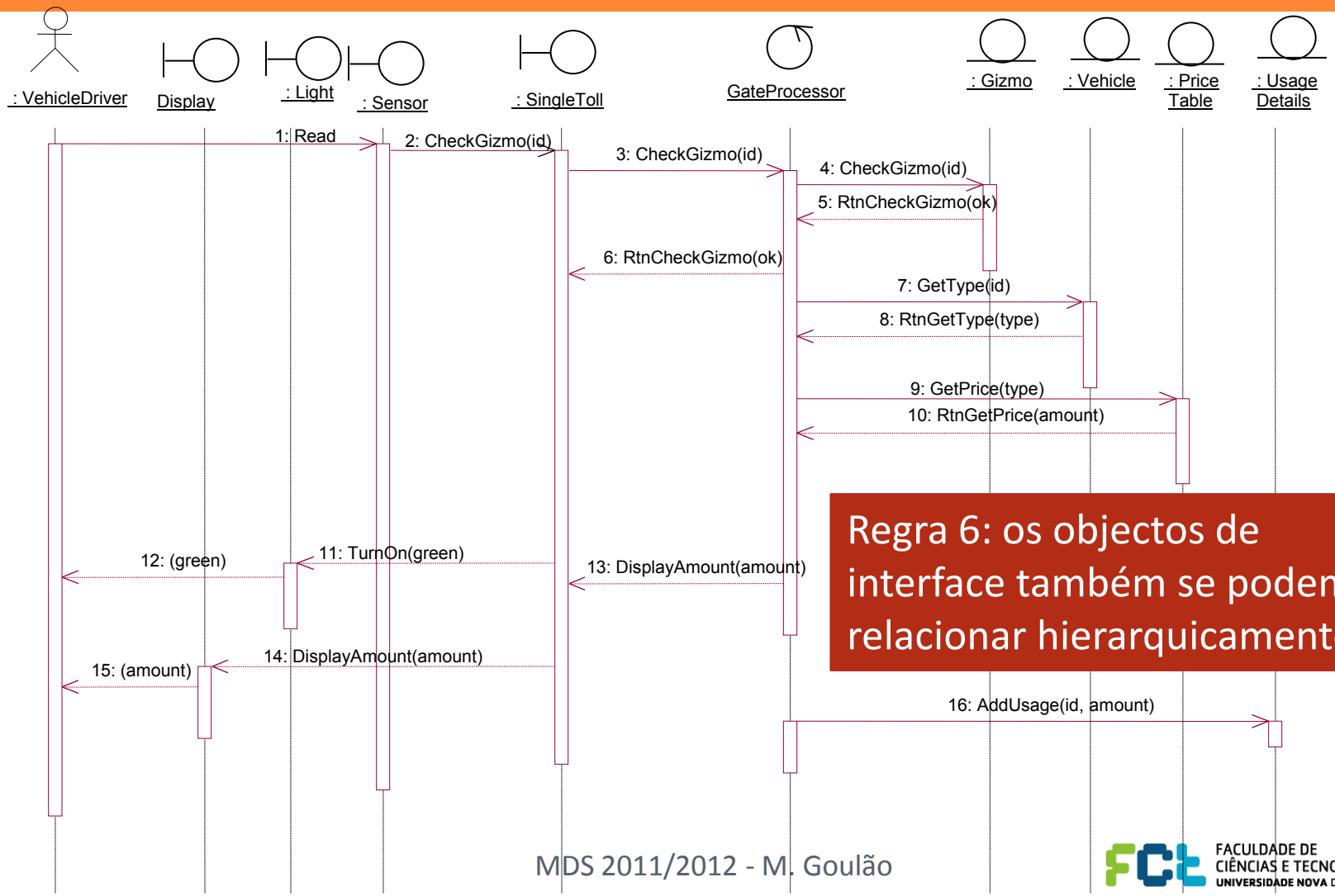
Regra 4: objectos de controlo centralizam o processamento que envolve objectos de entidade.

DS com os componentes da portagem



Regra 5: os objectos de interface podem ser variados.

DS: notação alternativa



Regra 6: os objectos de interface também se podem relacionar hierarquicamente.

Métodos de Desenvolvimento de Software (MDS) 2011/2012

Miguel Goulão
mgoul@fct.unl.pt
http://ctp.di.fct.unl.pt/~mgoul/

Aula 13

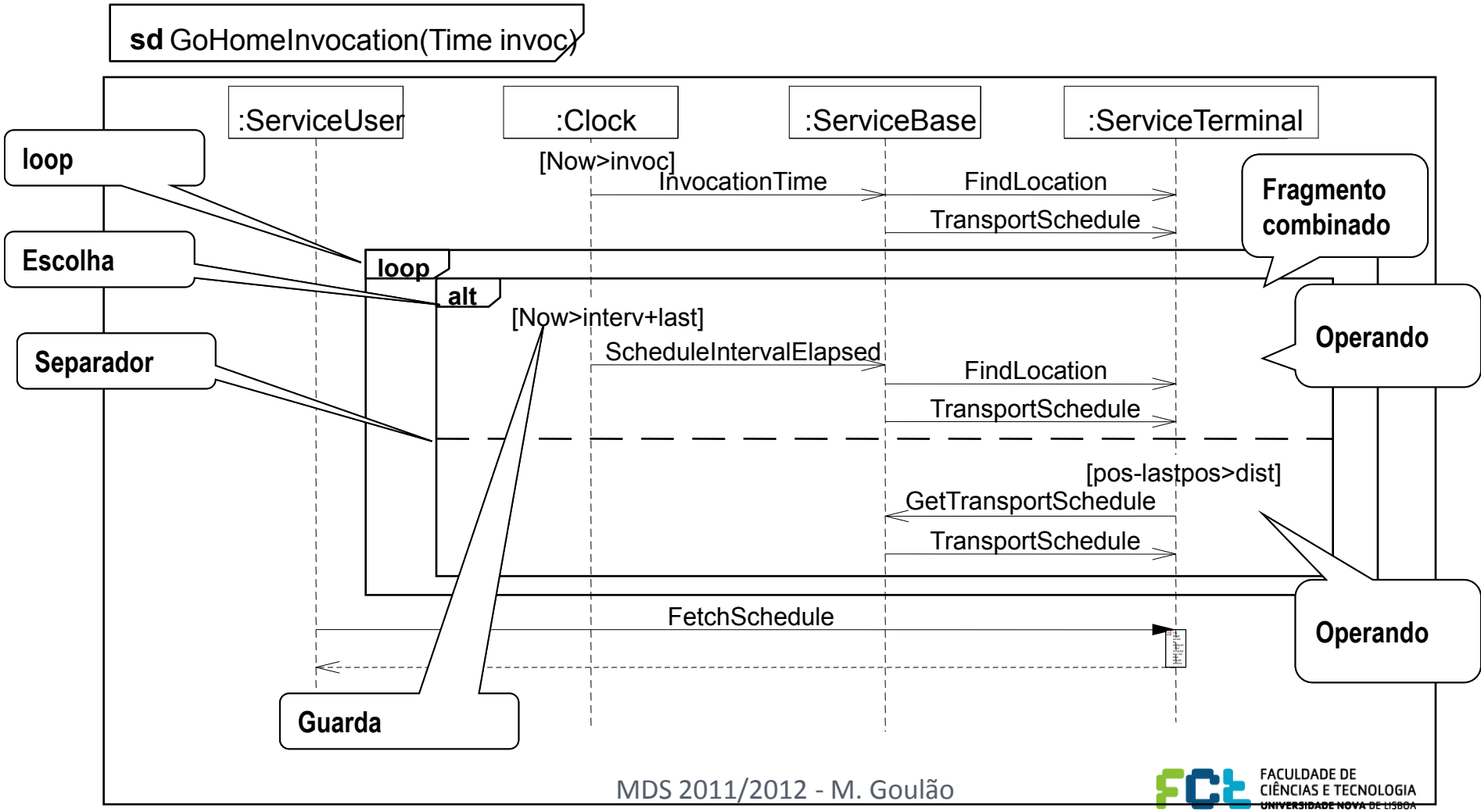
Diagramas de Interacção

Fragmentos combinados e operadores

Fragmentos combinados e operadores

- Fragmentos combinados dividem um diagrama de sequência em áreas diferentes com comportamentos diferentes
- Compostos por
  - Um operador
    - Determina **como** os operandos executam
  - Um ou mais operandos
  - Zero ou mais condições de guarda
    - Determinam **se** os operandos executam ou não

Fragmentos combinados e operadores



Tipos de fragmentos combinados

- Alternativas (**alt**)
  - Escolha alternativa de comportamentos – no máximo, um vai executar
  - Depende do valor da guarda (a guarda "else" é suportada)
    - (semelhante a um switch)
- Opção (**opt**)
  - Caso especial de alternativa em que apenas um operando executa
    - (semelhante a um if… then)
- Interrupção (**break**)
  - Representa uma alternativa que é executada em vez do resto do fragmento
    - (semelhante a um break num ciclo)
    - Quando a guarda é verdadeira o operando executa, mas não o resto da interacção em que esta se insere

Tipos de fragmentos combinados

- Ciclo (**loop**)
  - Guarda opcional: [<min>, <max>, <Boolean-expression>]
    - loop min, max [condition]
      - loop min vezes, depois, enquanto condition for verdade, executa (max – min) vezes
  - A ausência de guarda significa que não existe limite especificado
- Referência (**ref**)
  - Referência para outra interacção
- Paralelo (**par**)
  - Todos os operandos executam concorrentemente (intercalados)

Tipos de fragmentos combinados

- Região crítica(**region**)
  - Os traços não podem ser intercalados com eventos de outras lifelines, ou seja, executam sem interrupção
- Sequência fraca (**seq**)
  - Os operandos executam em paralelo nas diferentes linhas de vida, com uma restrição: eventos recebidos na mesma linha de vida, originados por diferentes operandos ocorrem na mesma sequência que os operandos ocorrem
- Sequência forte (**strict**)
  - Os operandos executam em sequência estrita

Tipos de fragmentos combinados

- Negativo (**neg**)
  - Identifica sequências que não podem ocorrer
- Ignorar (**ignore**)
  - Listar mensagens propositadamente omitidas
    - Exemplo: {m1, m2, m3}
- Considerar (consider)
  - Listar mensagens intencionalmente incluídas
    - Exemplo:
- Asserção (**assert**)
  - Representa o único comportamento válido num dado ponto da interacção

Dos diagramas de actividades aos DS

- A estrutura básica do DS pode extrair-se do diagrama de actividades.
- Em particular:
  - Actividade -> Operação em classe
  - Transição -> Mensagem
  - Branching-Merge -> Alt
  - Fork-Join -> Par
  - (Ciclo: transição para trás) -> Loop

Complementos ao Diagrama de Classes

Dos diagramas de Sequência aos Diagramas de Classes

- O diagrama de classes de domínio deve completar-se com a informação dos diagramas de sequência.
- Em particular:
  - Linha de Vida -> Classe
    - Interface, controlo, (entidade)
  - Mensagem -> Operação
  - Mensagem e argumentos -> Associação
  - Mensagem -> Dependência

Diagrama de sequência

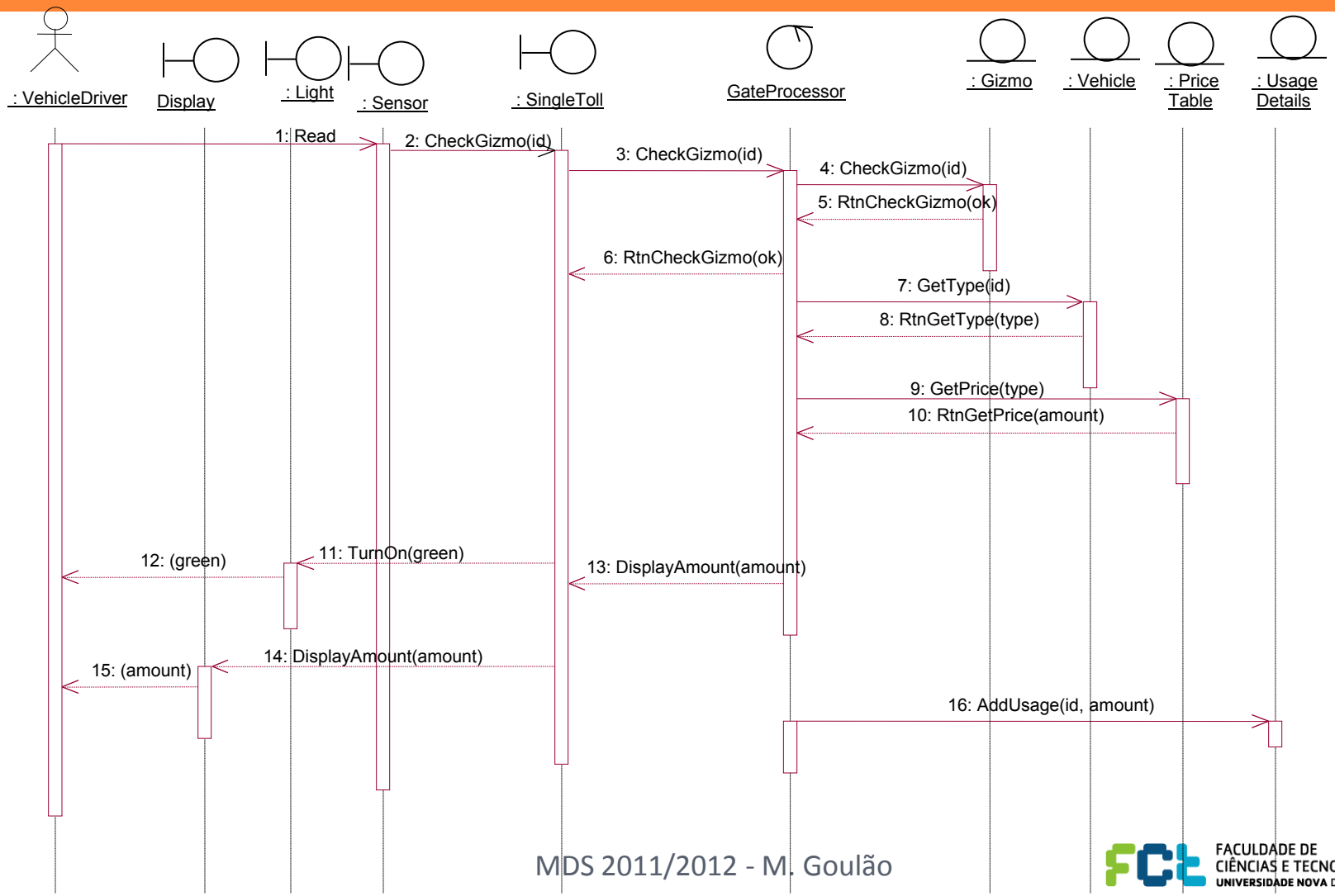


Diagrama de classes (parcial) resultante

Exemplo: veículo passa numa portagem de ponto único (e.g., ponte 25 Abril).

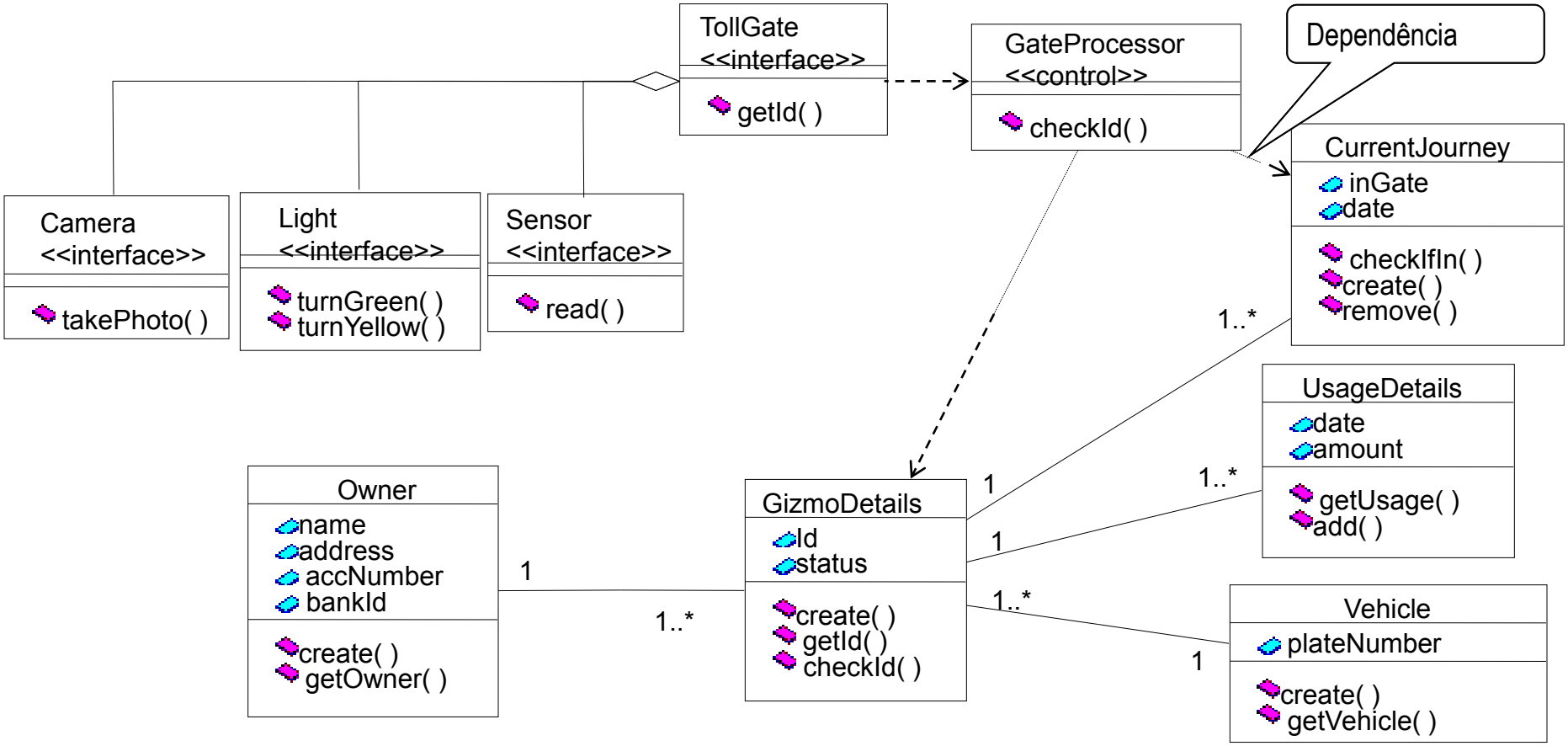


Diagrama de Colaboração

Diagrama de Colaboração

- Um diagrama de colaboração é um diagrama de interacção que enfatiza a organização estrutural dos objectos que enviam e recebem mensagens.
- Mostra o conjunto de objectos, ligações (links) entre estes objectos, e mensagens enviadas e recebidas por esses objectos.
- É útil para ilustrar a visão dinâmica do sistema.

Diagrama de Colaboração

- Oferece uma segunda forma de mostrar a sequência na qual os eventos ocorrem
  - Os objectos são mostrados em rectângulos ligados por linhas que indicam links entre eles
  - Números indicam a ordem na qual as operações são executadas
  - Os números são escritos junto dos nomes das mensagens e uma seta indica o sentido do fluxo

Diagrama de Colaboração: exemplo simples

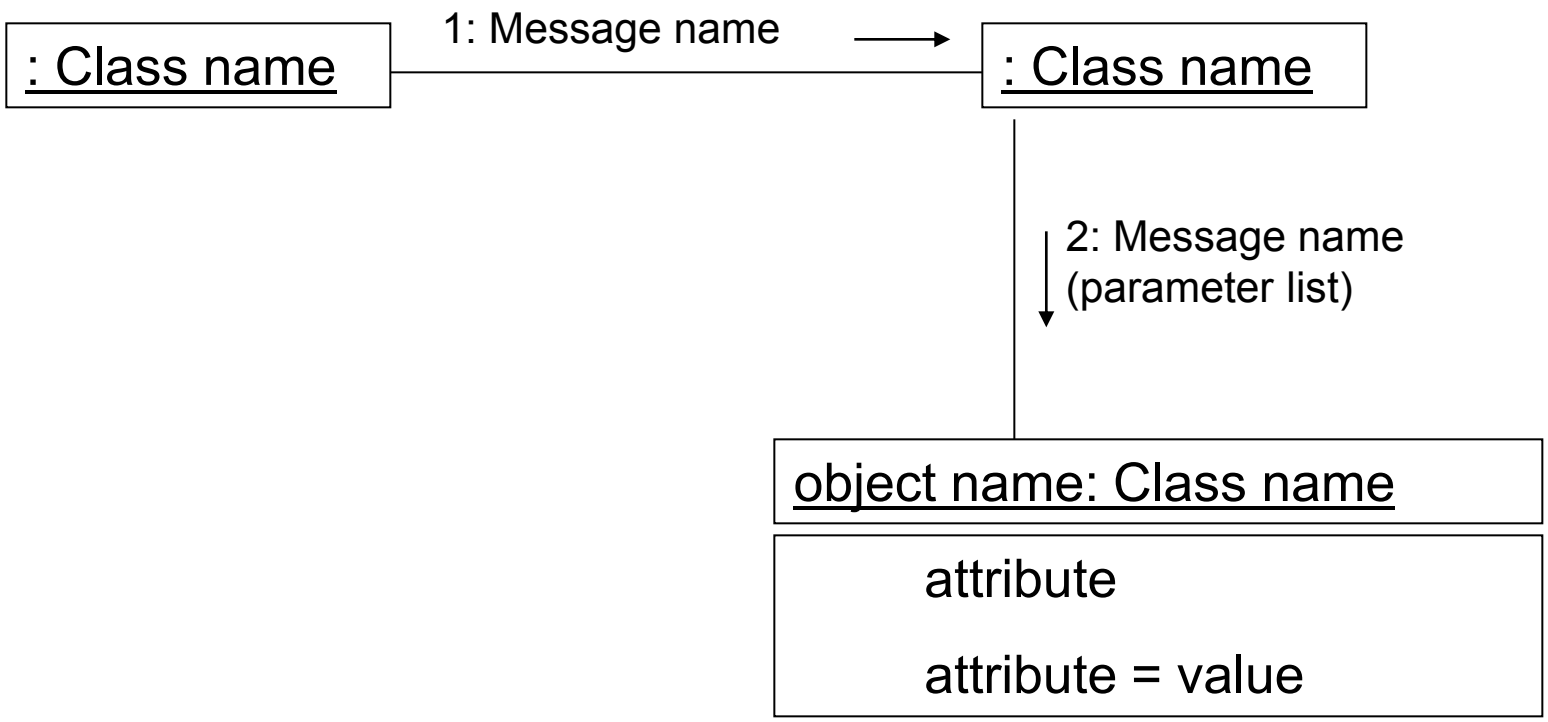


Diagrama de colaboração (cont)

- A numeração associada às mensagens pode representar aninhamento.
  - Por exemplo mensagem 1.1 é a 1ª mensagem aninhada na mensagem 1, 1.2 é a segunda e assim sucessivamente.
- Equivalência semântica com os DS: pode converter-se automaticamente um Diagrama de Sequência num Diagrama de Colaboração

Diagrama de colaboração reserva de quarto de hotel (com aninhamento de mensagens)

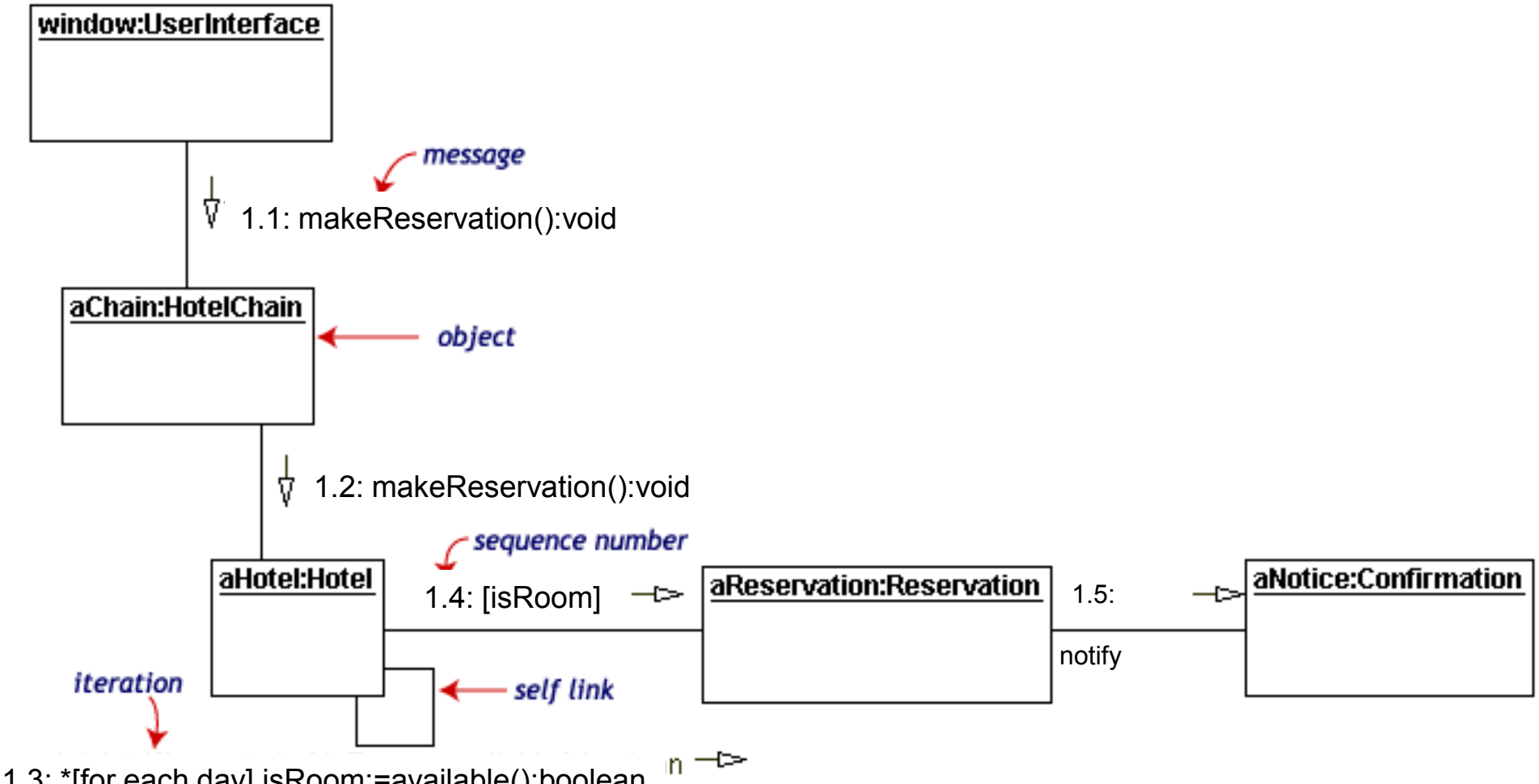
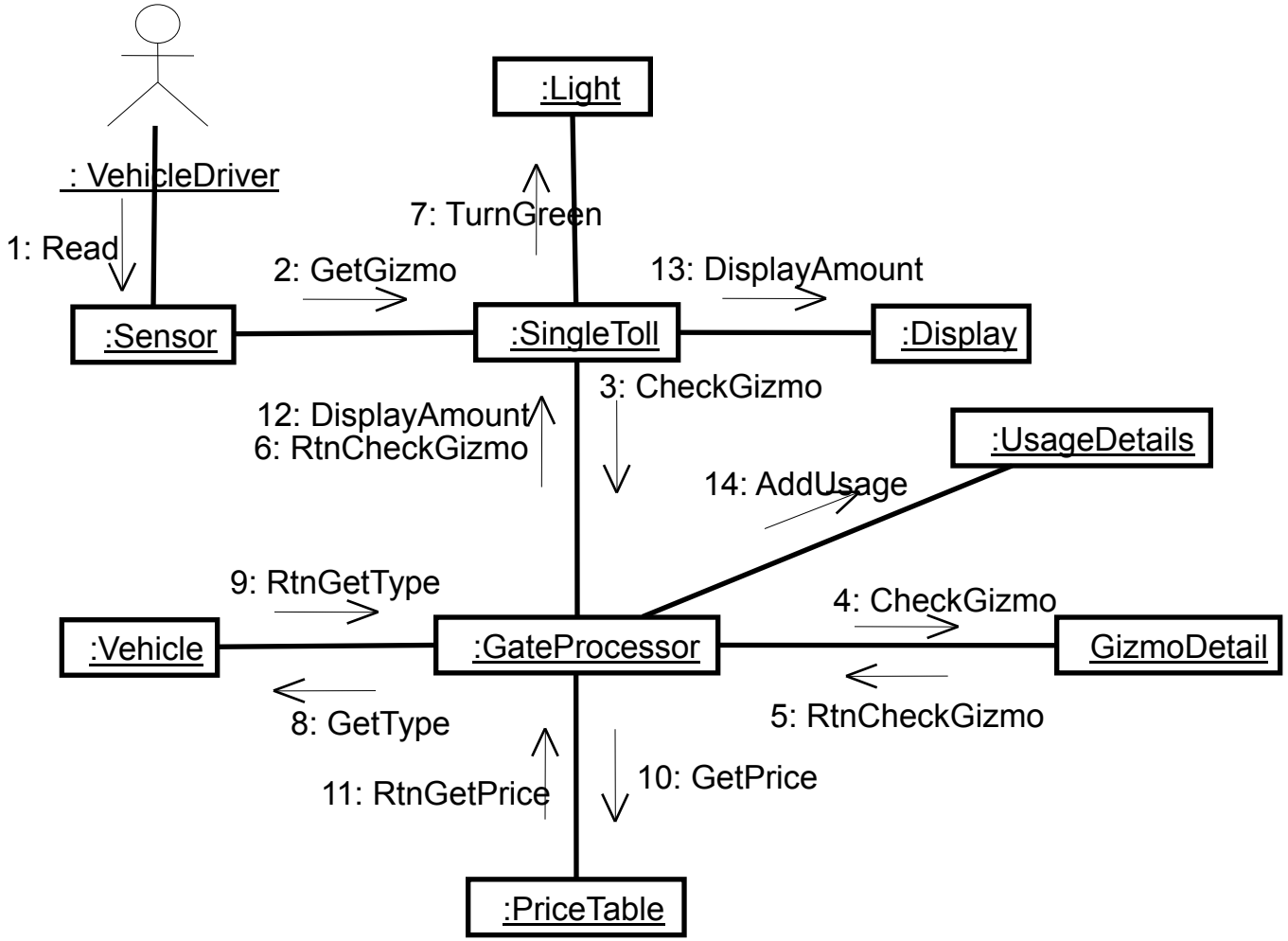


Diagrama de colaboração (portagem de ponto único)



Colaboração vs Sequência

- Os diagramas de sequência
  - lêem-se e compreendem-se rapidamente
  - a ordem das mensagens é clara e muito intuitiva
  - oferecem estruturas próprias para representar ciclos, concorrência, alternativas, etc
  - mas requerem mais disciplina na sua construção
- Os diagramas de colaboração
  - exigem menos disciplina na sua construção (não precisamos de conhecer o local certo onde o objecto deve ficar)
  - a ordem das mensagens pode ser adiconada à posteriori
  - mas são mais pobres nas estruturas que oferecem

Complementos de Diagramas de Classes

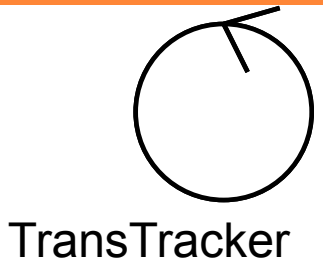
Tipos de (classes de) objectos de análise

- **Objecto interface** (boundary): interage directamente com o sistema.
  - Separa a funcionalidade do sistema da apresentação de resultados.
- **Objecto entidade**: guarda o estado do sistema.
  - Sobrevive à execução de um use case.
- **Objecto controlo**: contém funcionalidade que não "cabe" nos outros objectos.
  - Não sobrevive para além da realização do caso de uso que o originou. (Um por caso de uso, mas apenas se necessário.)

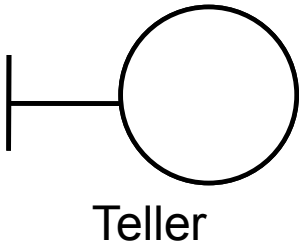
Tipos de classes de objectos

- **Entidade**: objectos desta classe são **passivos**, i.e., não iniciam interações por si próprios. Um objecto entidade **pode participar na realização de muitos casos de uso** diferentes e normalmente **vive para além da realização destes**.
- **Controlo**: controla interações entre vários objectos. Uma classe de controlo tem o **comportamento específico para um caso de uso**. Um objecto de controlo normalmente não vive para além da realização do caso de uso em que participa.
- **Interface** (Boundary): fica na periferia do sistema, mas dentro dele. **Interage com actores fora do sistema** e com qualquer dos 3 tipos de classes de análise.

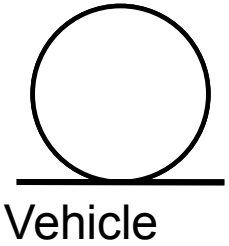
Notação



<<control>>
TransTracker

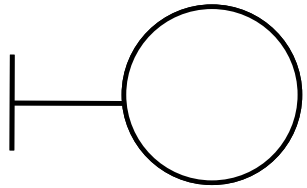


<<boundary>>
Teller



<<entity>>
Vehicle

Identificar objectos interface



- Objectos interface traduzem as acções dos actors para eventos que o sistema entende e, no outro sentido, traduzem os eventos do sistema em algo que pode ser apresentado ao actor.
- Encontram-se na descrição de interfaces ou então nos casos de uso, começando pelos actors e vendo como eles comunicam com o sistema.
- Devemos tomar em consideração:
  - apresentação da informação ao utilizador;
  - pedidos de informação do utilizador;
  - comportamento que muda se o comportamento do actor for modificado;
  - sequências de acções que dependem de um tipo de interface.

Identificar objectos entidade

- O objectivo dos objectos entidade é guardar informação.
- A informação guarda-se em atributos.
- Cada atributo tem um tipo (simples ou composto) e uma cardinalidade (0..n, onde n>0).
- Objectos entidade existem nos casos de utilização.
  - É fácil identificar objectos em excesso.
  - É difícil identificar apenas os necessários.

Identificar objectos entidade

- Pode ser difícil distinguir um atributo de um objecto.
  - Atributo: a informação nunca é manipulada separadamente.
  - Objecto: a informação pode ser tratada separadamente.
- Objectos entidade têm comportamento.
- Operações típicas são:
  - guardar e procurar informação
  - actualizar informação
  - criar e remover (objectos)

Identificar objectos controlo

- Objectos controlo são responsáveis por funcionalidade que não pertence a nenhum dos outros objectos.
  - Ligam os objectos interface com os objectos entidade.
  - Se esta funcionalidade fosse atribuída a outros tipos de objectos (que é o que alguns métodos de análise orientados pelos objectos fazem), reduzia-se a modificabilidade.
  - Seria preciso espalhar a funcionalidade por mais que um objecto, o que reduz a localização.
- Primeiro devem identificar-se objectos interface e entidade. Funcionalidade que reste, vai para os objectos controlo (cuidado, apenas um por caso de uso)

Vantagens em usar três tipos de objectos

- Um sistema está em mudança constante.
- Usando vários tipos de objectos, as modificações são mais localizáveis, afectando menos objectos.
  - Modificações na representação dos dados, por exemplo, afectam apenas objectos interface.
  - Modificações nos dados, afectam apenas objectos entidade.
  - Modificações na funcionalidade afectam objectos de entidade ou de controlo.

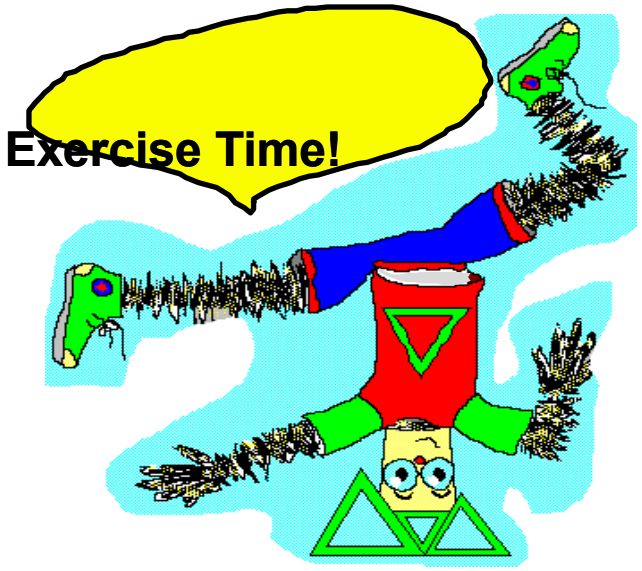
Análise: em resumo

- Análise detalhada do domínio do problema
  - complementar o diagrama de classes de domínio (anterior) adicionando mais classes;
  - definir o comportamento das classes (operações);
  - definir as interacções entre objectos (dependências);
  - refinar objectos (interface) e identificar associações em falta.
- Preservar abstracção para garantir reutilização
- Classificação dos objectos aumenta localização
  - objectos entidade
  - objectos fronteira ou interface
  - objectos controlo

Diagrama de classes: resumindo

- Estratégia:
  - Identificar classes de objectos
  - Desenhá-las no diagrama de classes
  - Algumas das classes são mais facilmente identificáveis durante a construção do modelo dinâmico, em particular, durante a construção dos diagramas de sequência
- O melhor mesmo é construir o diagrama de classes em paralelo com os diagramas de sequência.

Experimente, faça você mesmo!
Sistema bancário automático



Os clientes do banco podem levantar dinheiro das suas contas, fazer depósitos ou pedir o saldo corrente.

Todas estas operações são completadas usando as caixas automáticas (Multibanco) ou ao balcão.

As transacções numa conta fazem-se por cheque, ordens de pagamento, ou então usando as caixas automáticas com um cartão.

Há dois tipos de contas: à ordem e a prazo. Uma conta a prazo dá juros e não pode ser acedida através das caixas automáticas

Agradecimentos

Versão revista e aumentada dos slides de MDS 2010/2011, de Ana Moreira e João Araújo