OBJECT-ORIENTED DESIGN

Ian Sommerville, 8° edição – Capítulo 14

Aula de Luiz Eduardo Guarino de Vasconcelos

Objetivos

- Explicar como um projeto de software pode ser representado como um conjunto de objetos que interagem entre si, que gerenciam seus próprios estados e operações
- Descrever as atividades no processo de projeto orientado a objetos
- Introduzir vários modelos que descrevem um projeto orientado a objetos
- Mostrar como a UML pode ser utilizada para representar esses modelos

Tópicos abordados

- Objetos e classes de objetos
- Processo de projeto orientado a objetos
- Evolução de projeto

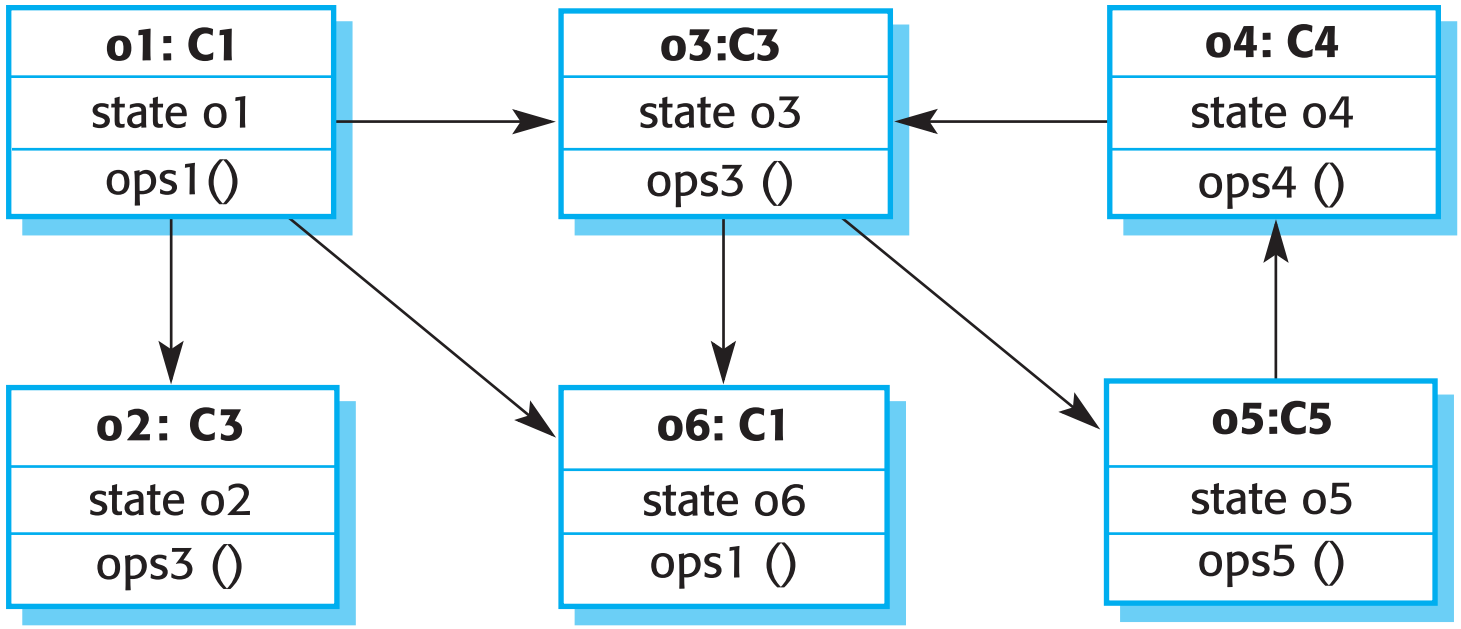
Object-oriented development (POO)

- Análise Orientada Objeto, projeto e programação são relacionadas mas distintas.
- OOA (Analysis) concentra-se no desenvolvimento do modelo de objetos do domínio da aplicação. Objetos refletem as entidades.
- OOD (Development/Projeto) concentra-se no desenvolvimento do modelo do sistema orientado a objetos, seus relacionamentos e na implementação dos requisitos.
- OOP (Programming) concentra-se em resolver o OOD usando uma linguagem de programação OO

Características de OOD/POO

- Objetos são abstrações do mundo real ou entidades de sistema e gerenciam a si mesmos
- Objetos são independentes e encapsulam estado e representação de informação
- Funcionalidade do sistema é expressa em termos de serviços de objetos
- Áreas de dados compartilhados são eliminadas. Objetos se comunicam por troca de mensagens
- Objetos podem estar distribuídos e podem executar sequencialmente ou em paralelo

Interação de objetos



Vantagens do OOD

- Manutenção mais fácil. Objetos podem ser entendidos como entidades em *stand-alone*
- Objetos são componentes de software reutilizáveis
- Para alguns sistemas, pode haver um mapeamento óbvio de entidades do mundo real em objetos de sistema

Objetos e classes

- Objetos são entidades em um sistema de software que representam instâncias de entidades de sistema e do mundo real
- Classes são gabaritos (*templates*) para objetos. Elas podem ser utilizadas para criar objetos
- Classes de objetos podem herdar atributos e serviços de outras classes de objetos

Objetos

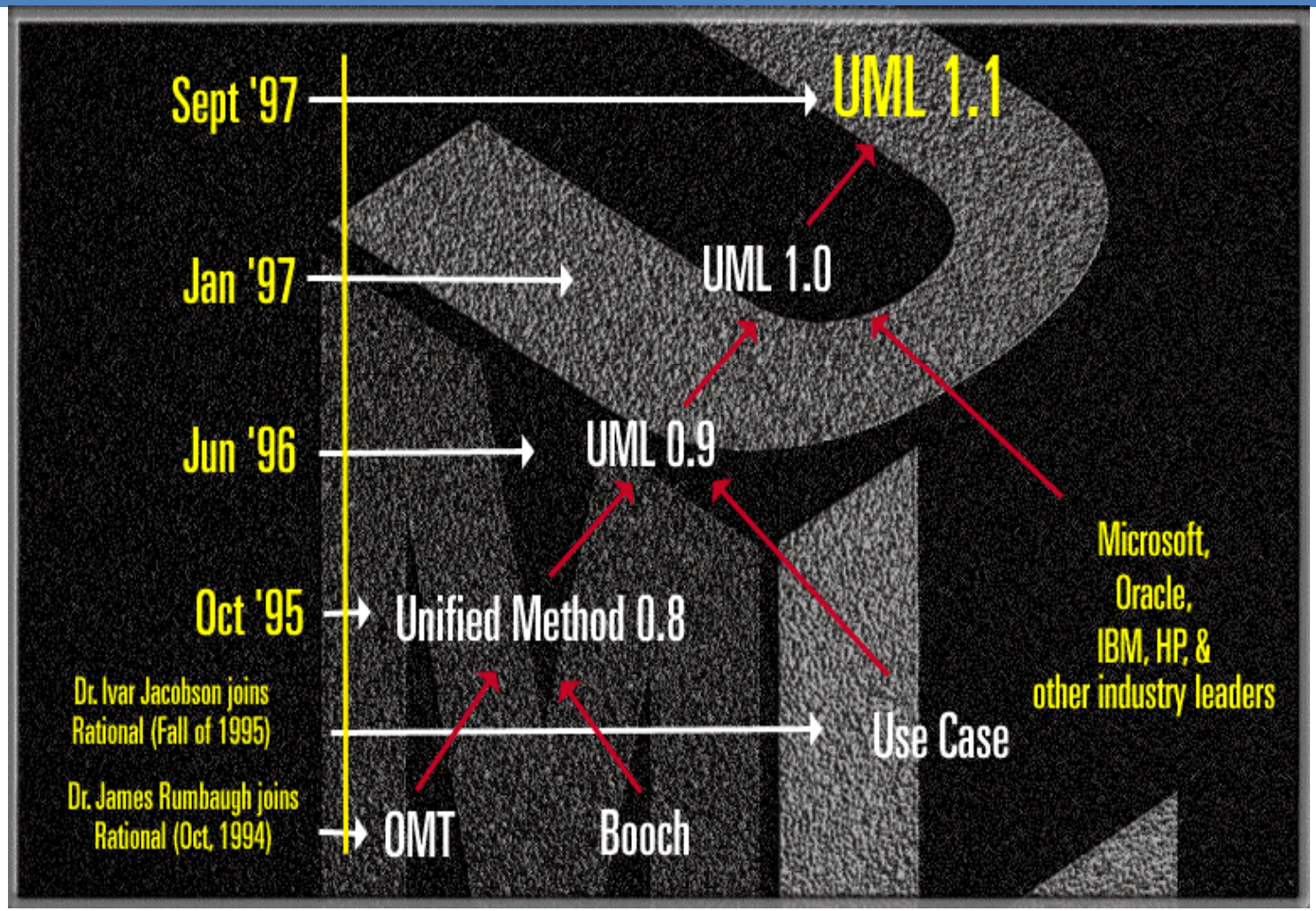
Um **objeto** é uma entidade que possui um estado e um conjunto definido de operações que operam nesse estado. O estado é representado por um conjunto de atributos de objeto. As operações associadas com o objeto fornecem serviços para outros objetos (clientes), que requisitam esses serviços quando alguma computação é necessária.

Objetos são criados de acordo com uma definição de **classe de objetos**, que serve como um *template* para criar objetos. Essa classe inclui declarações de todos os atributos e operações que devem ser associados com um objeto dessa classe.

A Unified Modeling Language (UML)

- Muitas notações diferentes para descrever projetos orientados a objetos foram propostas nas décadas de 1980 e 1990
- A Linguagem de Modelagem Unificada (UML) é uma integração dessas notações
- Ela descreve notações para um número de diferentes modelos que podem ser produzidos durante a análise e projeto orientado a objetos
- Tornou-se um padrão para modelagem orientada a objetos

Histórico da UML



Nov 97, aprovada pela OMG

UML dá suporte ao desenvolvimento de aplicações

Objects

Business Objects

Relationships

Large scale system

ORDBMS
Oracle

Classes

Application partitioning

Components
Microsoft

Scenarios

CORBA
OMG

Use Cases

ActiveX/COM
Microsoft

Business Process

Classe de objeto Funcionário (Employee) em UML

Comunicação de objetos (métodos)

- Conceitualmente, objetos se comunicam por troca de mensagens
- Mensagens
  - O nome do serviço requisitado pelo objeto chamador
  - Cópias das informações necessárias para executar o serviço e o nome de um proprietário para o resultado do serviço
- Na prática, mensagens são frequentemente implementadas como chamadas de procedimento
  - Nome = nome do procedimento
  - Informação = lista de parâmetros

Exemplos de mensagens/métodos

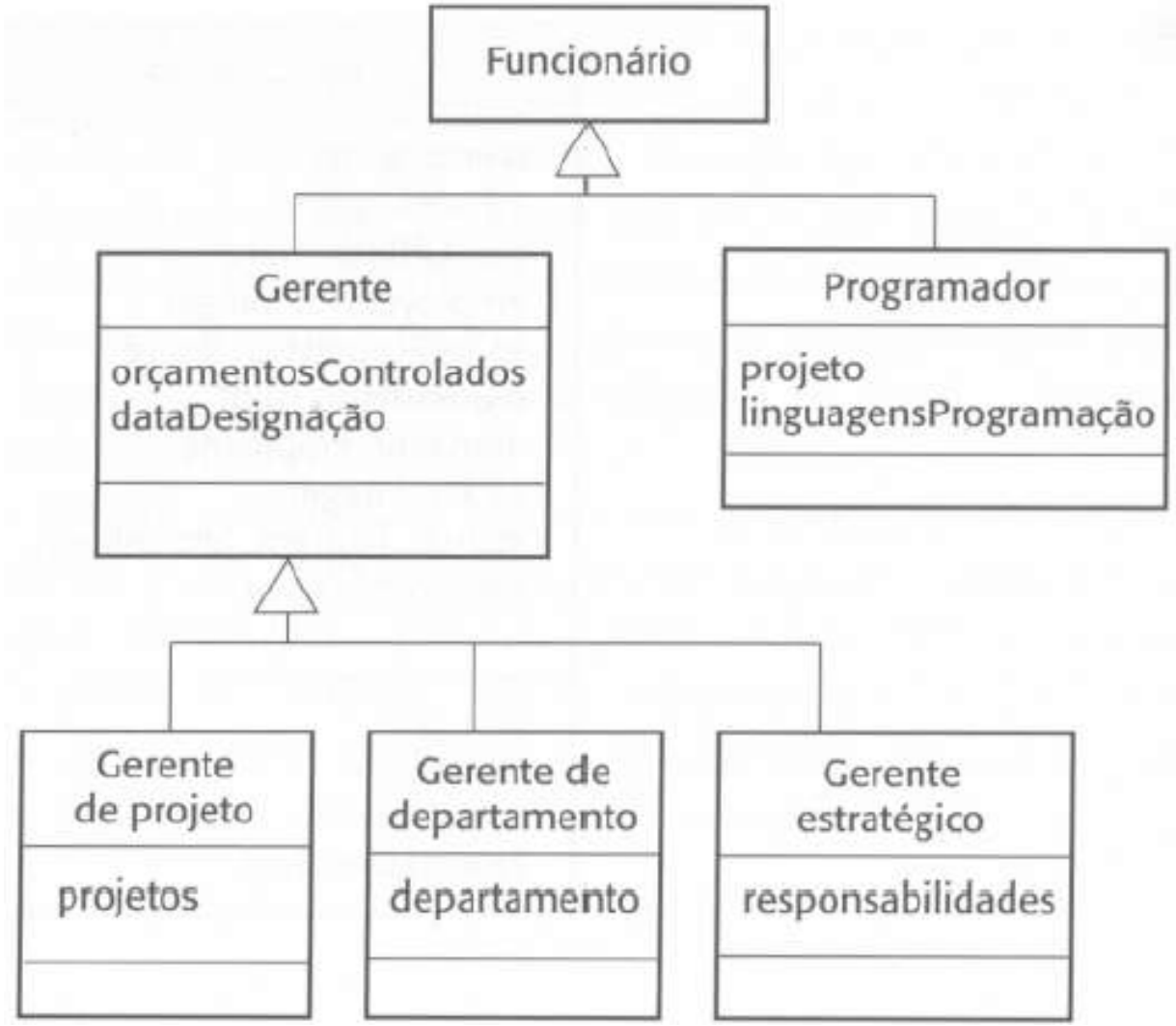
```
// Chamar um método associado com um objeto buffer
// que retorna o próximo valor no buffer
   v = circularBuffer.Get () ;

// Chamar o método associado com um termostato que
// ajuste a temperatura a ser mantida
   thermostat.setTemp (20) ;
```

Generalização e Herança

- Objetos são membros de classes que definem os tipos dos atributos e as operações
- Classes podem ser organizadas em uma hierarquia onde uma classe (uma super-classe) é uma generalização de uma ou mais classes (subclasses)
- Uma classe herda os atributos e operações da sua super-classe e pode adicionar novos atributos e operações
- Generalização em UML é implementada como herança nas linguagens de programação orientadas a objetos

Uma hierarquia de generalização



Vantagens da Herança

- É um mecanismo de abstração que pode ser utilizado para classificar entidades
- É um mecanismo de reuso tanto a nível de projeto, quanto a nível de programação
- O grafo de herança é uma fonte de conhecimento da organização sobre domínios e sistemas

Problemas com Herança

- Classes de objetos não são auto contidos, eles não podem ser compreendidos sem fazer referência à suas super-classes
- Projetistas têm a tendência de reutilizar o grafo de herança obtido na fase de análise. Isso pode levar à ineficiência significativa
- Os grafos de herança da análise, projeto e implementação possuem diferentes funções e devem ser mantidos separadamente

Associações em UML

- Objetos e classes de objetos participam no relacionamento com outros objetos e classes de objetos
- Na UML, as associações são denotadas por uma linha entre as classes de objetos
- Associações podem ser anotadas com informações que descrevem a associação
- Associações podem indicar que um atributo de um objeto é um objeto associado ou que um método envolve um objeto associado

Um modelo de associação

Um processo de projeto orientado a objetos

- Definir o contexto e os modos de utilização do sistema
- Projetar a arquitetura do sistema
- Identificar os principais objetos do sistema
- Desenvolver os modelos de projeto
- Especificar as interfaces dos objetos

Descrição de um sistema de mapeamento meteorológico

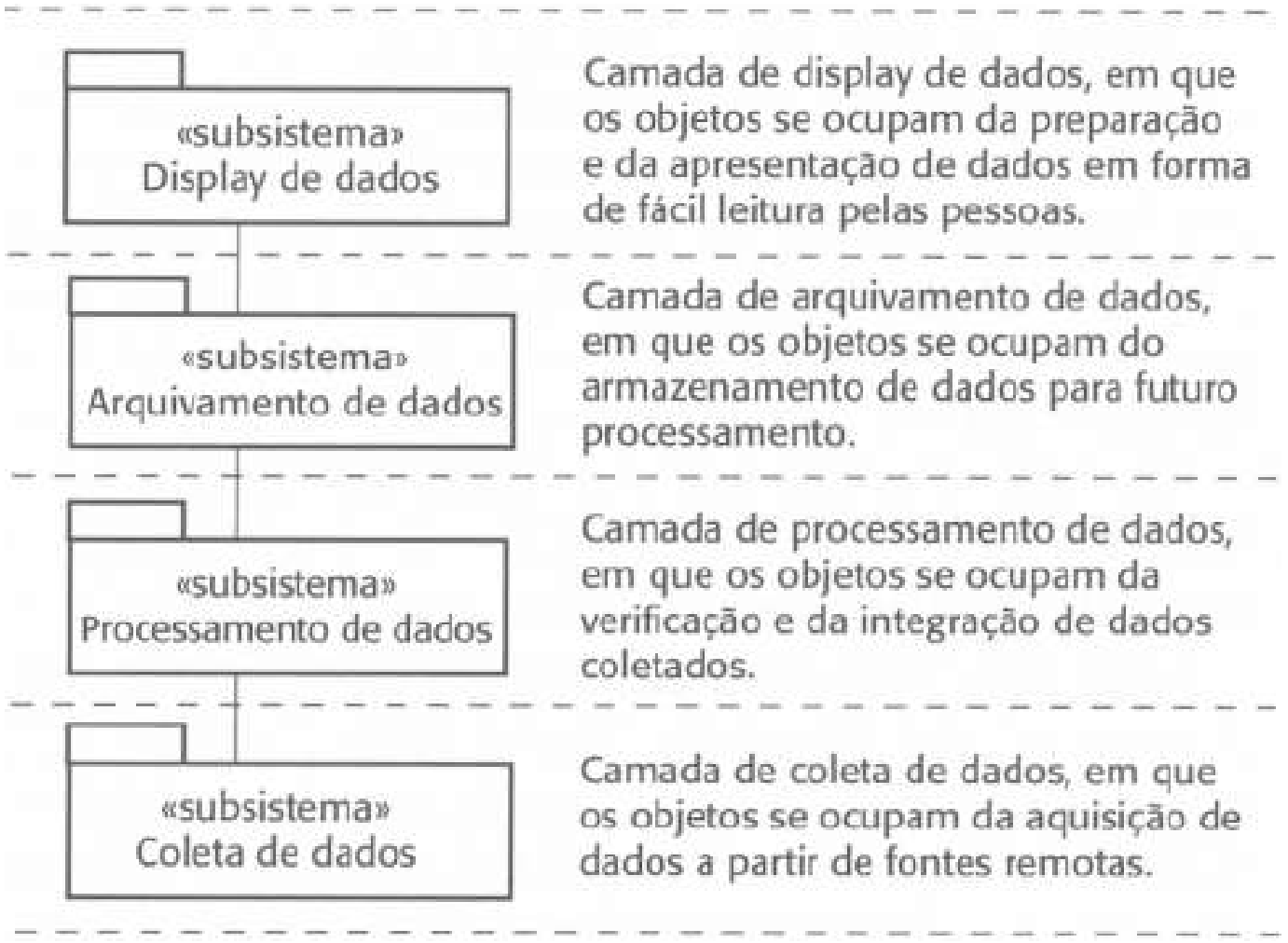
Um sistema de mapeamento meteorológico é necessário para gerar mapas meteorológicos regularmente, utilizando dados coletados a partir de estações meteorológicas remotas sem que seus funcionários estejam presentes, e de outras fontes de dados, como observadores de tempo, balões e satélites meteorológicos. As estações meteorológicas transmitem seus dados ao computador da área em resposta a uma requisição dessa máquina.

O sistema de computador da área valida os dados coletados e faz a integração dos dados a partir de diferentes fontes. Os dados integrados são arquivados e, com os dados desse arquivo e um banco de dados de mapas digitalizados, é criado um conjunto de mapas meteorológicos locais. Os mapas podem ser impressos para distribuição em uma impressora especial ou ser exibidos em diversos formatos.

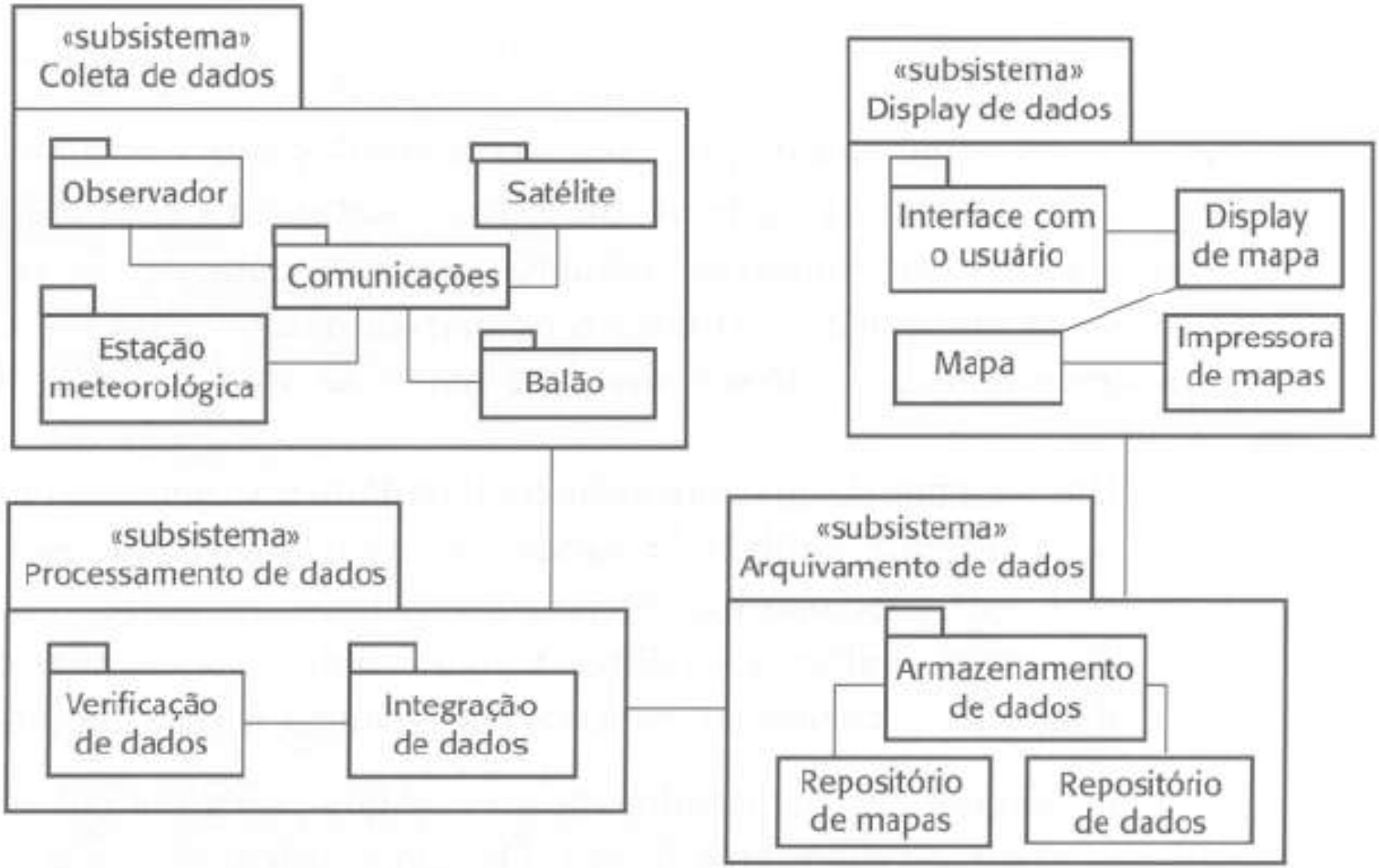
Contexto do sistema e modelos de uso

- Desenvolver uma compreensão das relações entre o software que está sendo desenvolvido e seu ambiente externo
- Contexto do sistema
  - Um modelo estático que descreve os outros sistemas no ambiente. Utiliza um modelo de subsistema para mostrar outros sistemas
- Modelo de uso do sistema
  - Um modelo dinâmico que descreve como o sistema interage com seu ambiente. Utiliza casos de uso para mostrar interações

Arquitetura em camadas



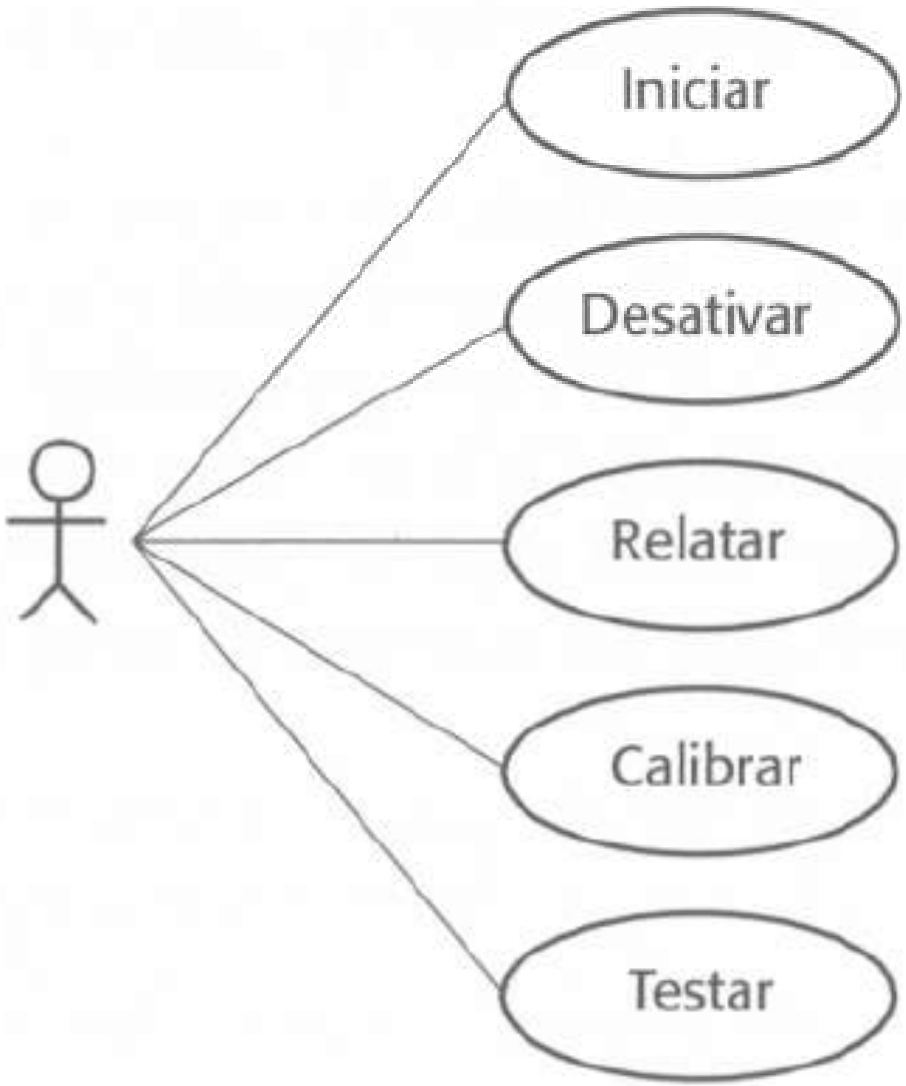
Sub-sistemas em um sistema de mapeamento meteorológico



Use-case models

- Modelos de Use-case são usados para representar cada interação com o sistema.
- O modelo de use-case mostra as características do sistema como elipses e interações de entrada como uma figura (ator)

Casos de uso para estação meteorológica



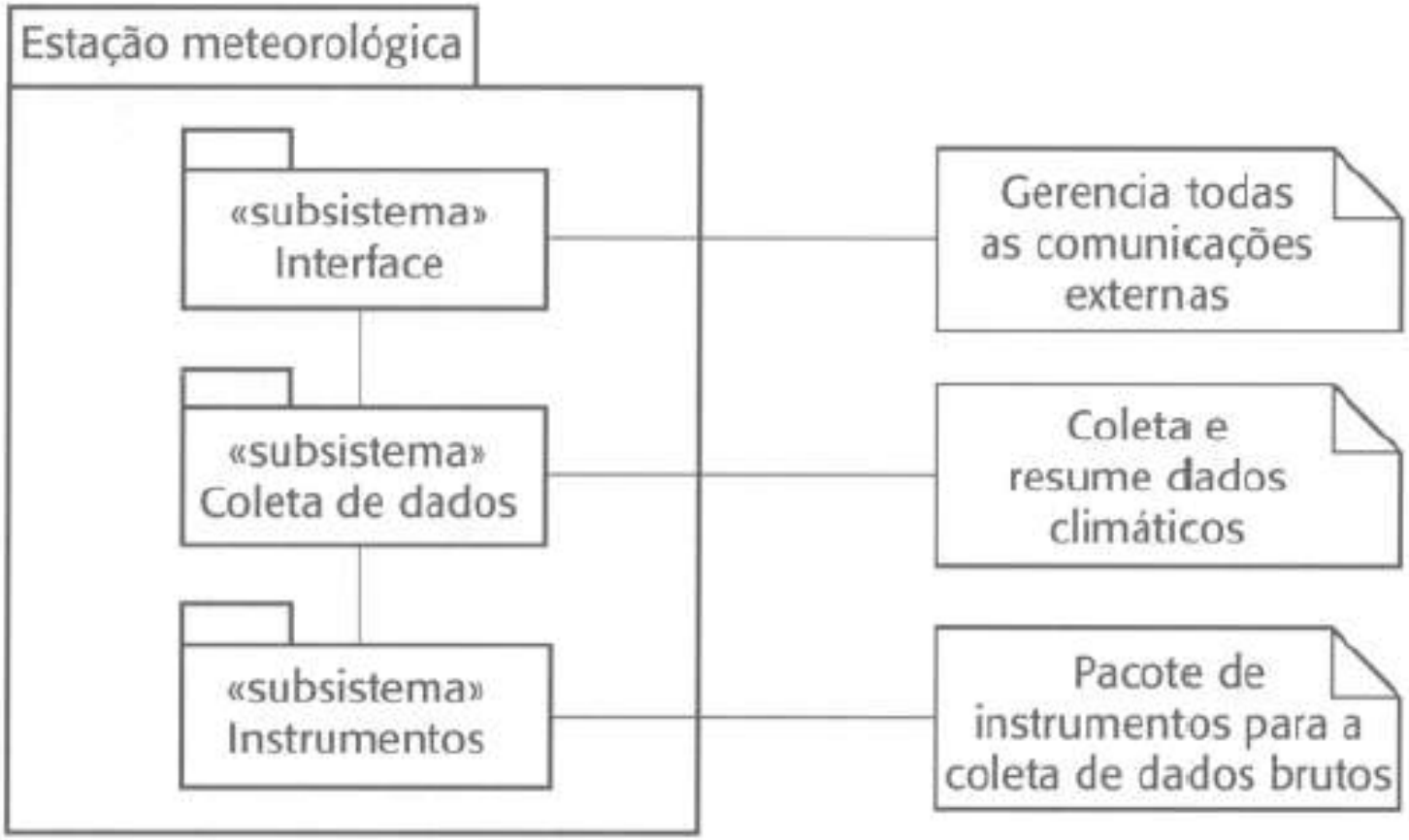
Descrição do caso de uso

Sistema	Estação meteorológica.
Caso de uso	Relatar.
Agentes	Sistema de coleta de dados sobre o clima, Estação meteorológica.
Dados	A estação meteorológica envia para o sistema de coleta de dados climáticos um resumo dos dados sobre o clima, que foram coletados a partir de instrumentos, no período de coleta. Os dados enviados referem-se às temperaturas máximas, mínimas e médias do solo e do ar; à pressão máxima, mínima e média do vento; à precipitação total das chuvas, e à direção do vento, conforme a amostragem a cada intervalo de cinco minutos.
Estímulo	O sistema de coleta de dados sobre o clima estabelece um link de modem com a estação meteorológica e requisita a transmissão dos dados.
Resposta	Os dados resumidos são enviados para o sistema de coleta de dados sobre o clima.
Comentários	Em geral, as estações meteorológicas recebem um pedido de relatório por hora, mas essa freqüência pode diferir de uma estação para outra e ser modificada no futuro.

Projeto de arquitetura

- Uma vez que as iterações entre o sistema e seu ambiente foram compreendidas, é possível utilizar essas informações como base para projetar a arquitetura do sistema
- Arquitetura em camadas é apropriada para a estação meteorológica
  - Camada de interface para gerenciar as comunicações
  - Camada de coleta de dados para gerenciar instrumentos
  - Camada de instrumentos para coletar dados
- Não deve haver mais que 7 entidades em um modelo de arquitetura

Arquitetura da estação meteorológica



Identificação de Objetos

- Identificar objetos (ou classes de objetos) é a parte mais difícil do projeto orientado a objetos
- Não há uma 'fórmula mágica' para a identificação de objetos. Essa tarefa depende da habilidade, experiência e conhecimento de domínio do projetista do sistema
- Identificação de objetos é um processo iterativo

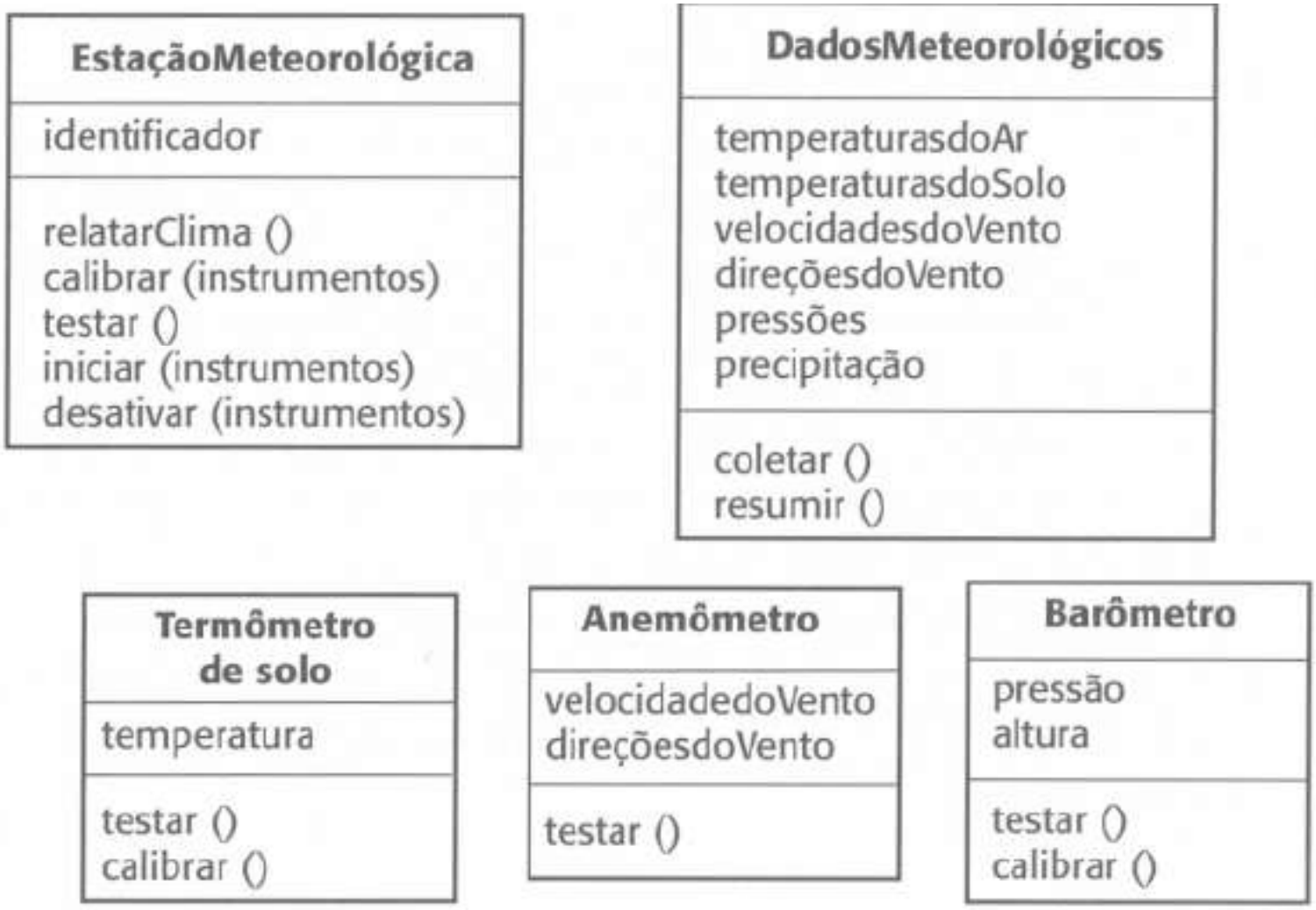
Abordagens para identificação

- Utilizar uma abordagem gramatical baseada em uma descrição em linguagem natural de um sistema
- Basear a identificação em entidades tangíveis no domínio da aplicação
- Utilizar uma abordagem comportamental identificando objetos baseados nos participantes num determinado comportamento
- Utilizar uma análise baseada em cenários. São identificados os objetos, atributos e métodos em cada cenário

Classes de Objetos da Estação Meteorológica

- Termômetro de solo, Anemômetro, Barômetro
  - Objetos do domínio da aplicação relacionados a instrumentos (hardware) no sistema
- Estação meteorológica
  - A interface básica da estação meteorológica com seu ambiente. Reflete as interações identificadas no modelo de caso de uso
- Dados meteorológicos
  - Encapsula os dados resumidos provenientes dos instrumentos

Classes de Objetos da Estação Meteorológica



Refinamento e identificação de outros objetos

- Utilizar conhecimento do domínio para identificar mais objetos e operações
  - Estações meteorológicas devem ter um identificador único
  - Estações meteorológicas são localizadas em lugares remotos de forma que as falhas dos instrumentos devem ser relatadas automaticamente. Portanto deve haver atributos e operações para verificar o funcionamento dos instrumentos
- Objetos ativos ou passivos
  - Neste caso, objetos são passivos e coletam dados conforme solicitados em vez de fazê-lo de maneira autônoma. Isso introduz flexibilidade às custas do aumento do tempo de processamento do controlador

Modelos de Projeto

- Modelos de projeto mostram os objetos ou as classes de objetos e os tipos de relações entre essas entidades
- Modelos estáticos descrevem a estrutura estática do sistema em termos de classes de objetos e relacionamentos
- Modelos dinâmicos descrevem as interações dinâmicas entre objetos

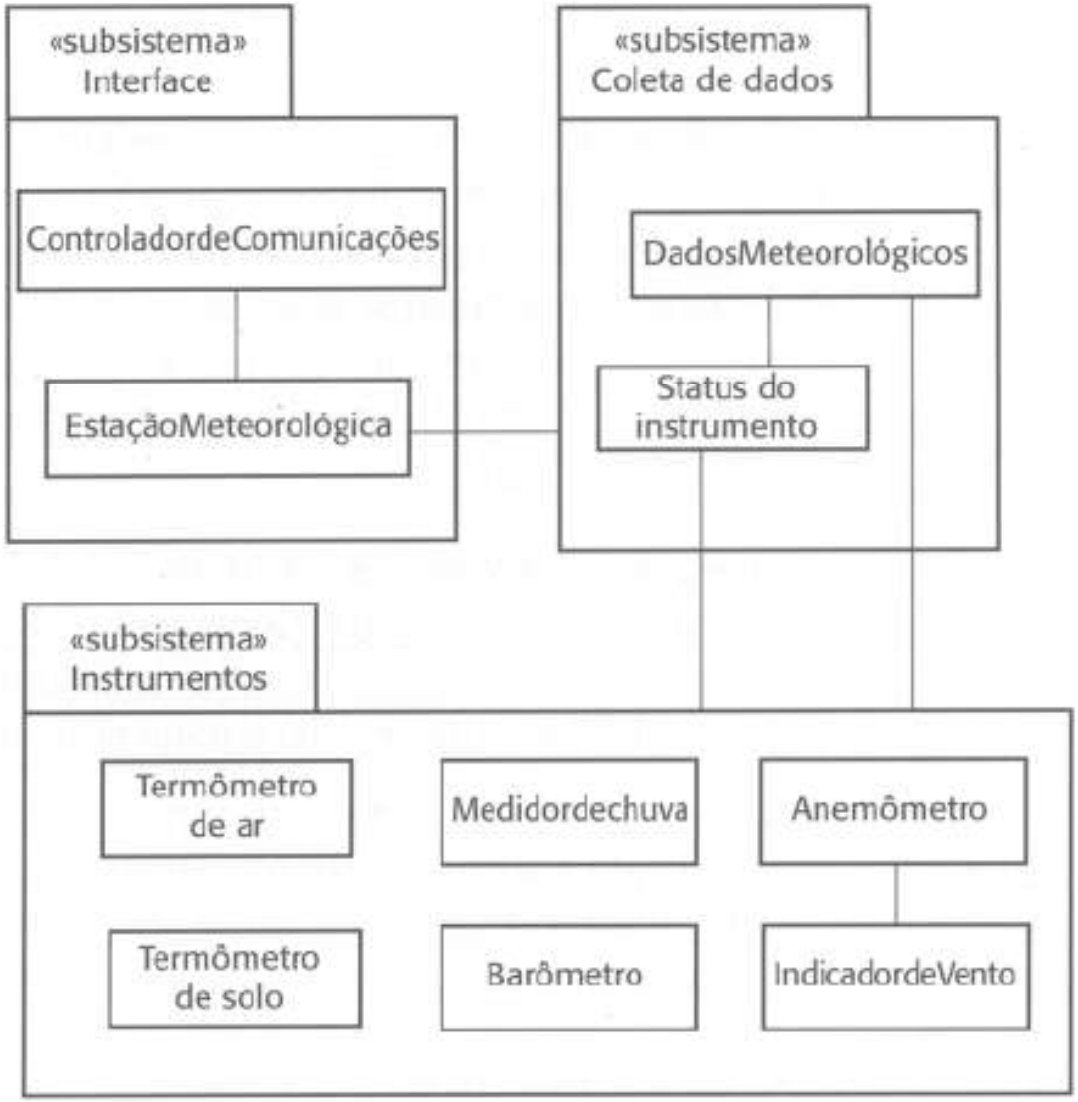
Exemplos de Modelos de Projeto

- Modelos de subsistemas mostram os agrupamentos lógicos de objetos em subsistemas coerentes
- Modelos de seqüência que mostram a seqüência de interações entre objetos
- Modelos de máquina de estado que mostram como objetos individuais mudam seu estado em resposta aos eventos
- Outros modelos incluem modelos de caso de uso, modelos de agregação, modelos de generalização etc

Modelos de subsistemas

- Mostram como o projeto é organizado em grupos de objetos relacionados logicamente
- Na UML, esses grupos são mostrados utilizando pacotes – um construtor de encapsulamento. Este é um modelo lógico

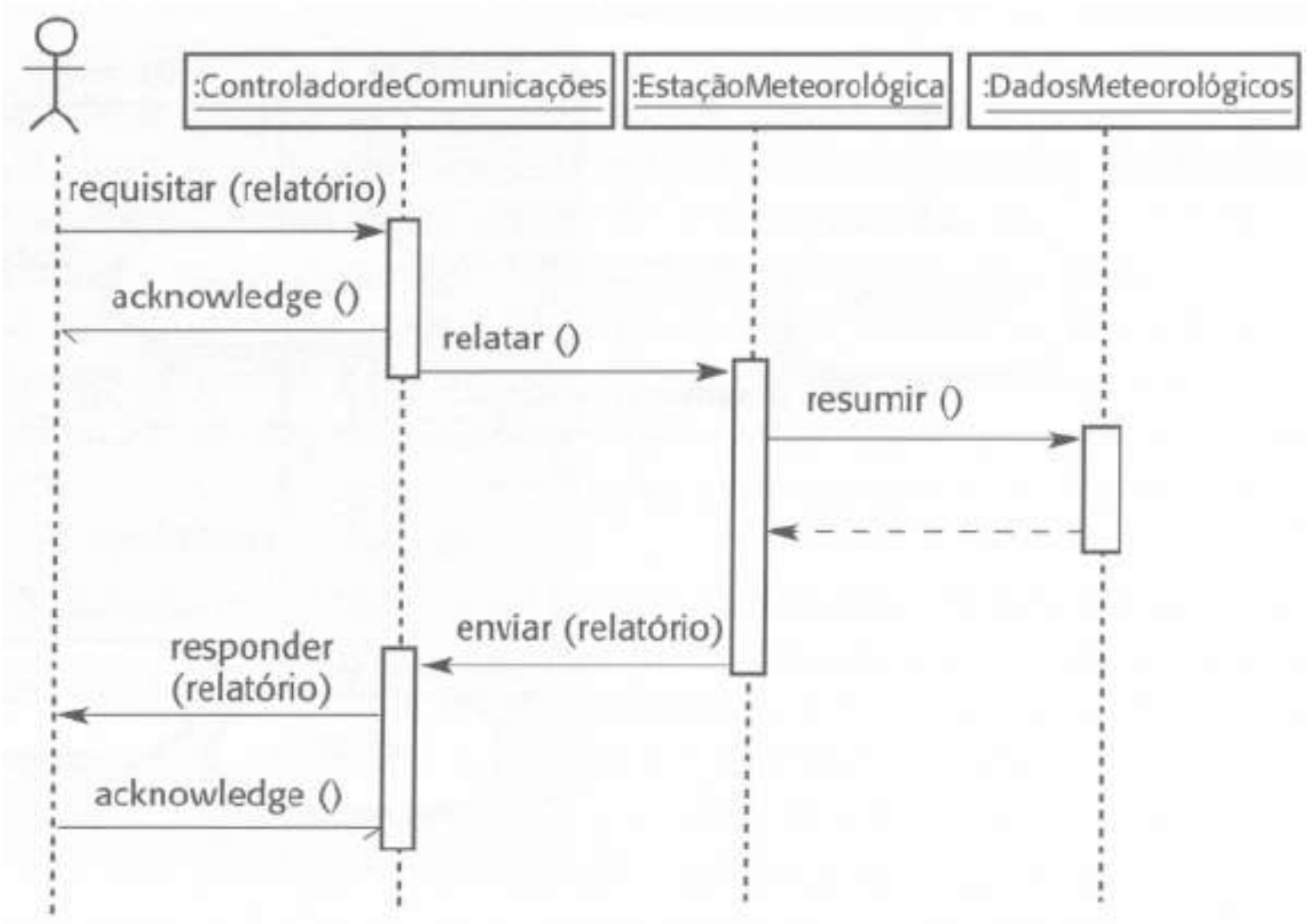
Subsistemas da Estação Meteorológica



Modelos de sequência

- Mostram a seqüência de interações entre objetos
  - Objetos são organizados horizontalmente no topo do diagrama
  - Tempo é representado verticalmente de forma que os modelos são lidos de cima para baixo
  - Interações são representadas por setas rotuladas. Diferentes estilos de setas representam diferentes tipos de interação
  - Um retângulo estreito na linha de vida do objeto representa o tempo pelo qual o objeto é o objeto controlador no sistema

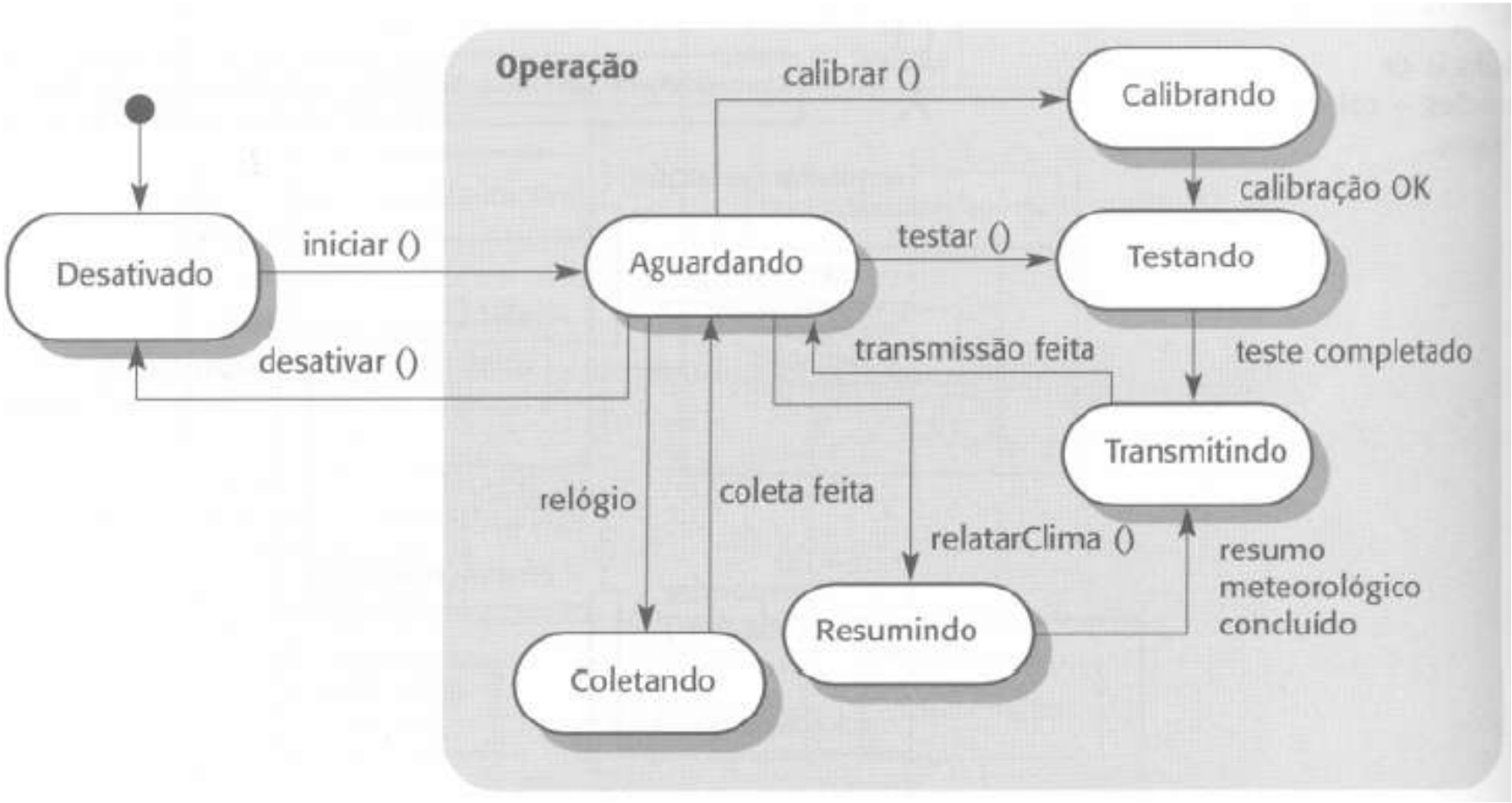
Sequência de operações – Coleta de dados



Diagramas de estado

- Mostra como a instância de um objeto responde a diferentes requisições de serviços e as transições de estados disparados por essas requisições
  - Se o estado do objeto é 'Desativado' então ele responde a uma mensagem iniciar()
  - No estado 'Aguardando' o objeto espera outras mensagens
  - Se o objeto receber uma mensagem relatarClima() então ele passa para o estado Resumindo
  - Se o objeto receber uma mensagem calibrar() então ele passa para o estado calibrando
  - Se um sinal de relógio é recebido o sistema passa para um estado de coleta

Diagrama de estado da estação meteorológica



Especificação de interface entre objetos

- Interfaces entre objetos devem ser especificadas de forma que os objetos e outros componentes possam ser projetados em paralelo
- Projetistas devem evitar informações de representação de interface mas devem ocultar isto no próprio objeto
- Objetos podem ter muitas interfaces que são pontos de vista dos métodos elas fornecem
- A UML usa diagramas de classe para a especificação de interface mas Java também pode ser utilizada

Interface da Estação Meteorológica

```
interface EstaçãoMeteorológica {

 public void EstaçãoMeteorológica () ;

 public void iniciar () ;
 public void iniciar (Instrumento i) ;

 public void desativar () ;
 public void desativar (Instrumento i) ;

 public void relatarClima () ;

 public void testar () ;
 public void testar (Instrumento i) ;

 public void calibrar (Instrumento i) ;

 public int obterID () ;

} //EstaçãoMeteorológica
```

Evolução de Projeto

- Ocultar informações dentro de objetos significa que mudanças feitas em um objeto não afetam outros objetos de uma maneira imprevisível
- Considere que algumas facilidades de monitoração da poluição devam ser adicionadas a cada estação meteorológica. Isso envolve adicionar um medidor de qualidade do ar para computar a concentração de vários poluentes na atmosfera
- As leituras de poluição são transmitidas ao mesmo tempo que os dados meteorológicos

Mudanças requeridas

- Adicionar uma classe de objeto chamada 'Qualidade do Ar' como parte da EstaçãoMeteorológica
- Adicionar uma operação relatarQualidadedoAr na EstaçãoMeteorológica. Modificar o software de controle para que as leituras de poluição sejam automaticamente coletadas
- Adicionar objetos que representam os instrumentos de monitoração de poluição

Monitoração da poluição

Pontos-chave

- OOD é uma abordagem para componentes do projeto que tem estados e operações privadas
- Objetos devem tem construtor e operações de inspeção. Devem prover serviços para outros objetos.
- Objetos podem ser implementados sequencialmente ou concorrentemente.
- A Unified Modeling Language provê diferentes notações para diferentes modelos de objetos

Pontos-chave

- Uma faixa de diferentes modelos pode ser produzida durante o processo de projeto orientado. Estes incluem modelos de sistemas estáticos e dinâmicos.
- Interfaces de Objetos devem ser definidas precisamente usando uma linguagem de programação
- Uma potencial vantagem de projeto OO é que ele simplifica a evolução do sistema

Exercício para casa

- Defina as classes e os objetos de um dos seguintes sistemas:
  - uma agência de correio
  - um posto de gasolina
  - um PDV
  - um computador.
- Especifique o diagrama de classes onde houver hierarquia.