UNIVERSIDADE ESTADUAL PAULISTA
INSTITUTO DE BIOCIÊNCIAS, LETRAS E CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIAS DE COMPUTAÇÃO E ESTATÍSTICA

Projeto Orientado a Objetos

Engenharia de Software

2o. Semestre de 2005

Projeto Orientado a Objeto

- Projetar sistemas usando objetos auto-contidos e classes de objetos.

Objetivos

- Mostrar como um projeto de software pode ser representado como um conjunto de objetos que interagem entre si, e gerenciam seus próprios estados e suas operações.
- Descrever as atividades em um processo geral de projeto orientado a objetos.
- Introduzir vários modelos que descreve um projeto orientado a objetos.
- Mostrar como a UML pode ser usada para representar esses modelos

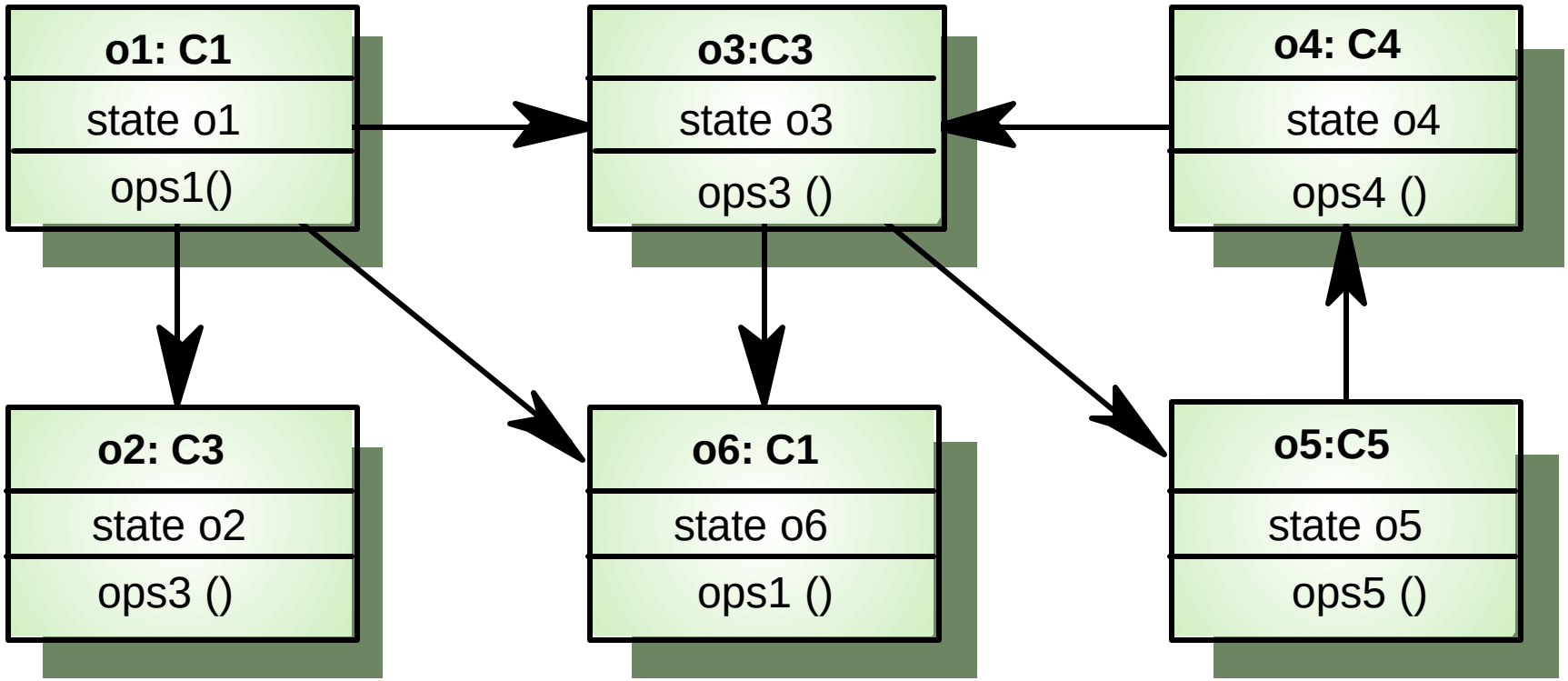
Tópicos

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

Características de Projeto Orientado a objetos

- Objetos são abstrações do mundo real ou entidades do sistema que se auto gerenciam.
- Objetos são independentes e encapsulam representações de informação e estado.
- A funcionalidade do sistema é expressa em termos de serviços dos objetos
- Áreas de dados compartilhado são eliminadas. Objetos se comunicam por passagem de mensagem.

Objetos que interagem entre si



Vantagens do Projeto OO

- Facilidade de manutenção. Objetos podem ser entendidos como entidades independentes.
- Os objetos são componentes potencialmente reutilizáveis.
- Para vários sistemas, existe um nítido mapeamento entre as entidades do mundo real para objetos no sistema.

Desenvolvimento Orientado a Objeto

- Análise, projeto e programação orientada a objetos são estratégias do processo de desenvolvimento OO.
- AOO se dedica a desenvolver um modelo orientado a objetos do domínio da aplicação.
- POO se dedica a desenvolver um modelo orientado a objetos de um sistema de software para identificar os requisitos identificados.
- POO se ocupa em realizar um projeto de software usando uma linguagem de programação tais como Java ou C++

Objetos e classes de de objetos

- Objetos são entidades no sistema de software que representam instâncias de entidades do mundo real e do sistema.
- Classes de objetos são *templates* utilizados para criar objetos.
- Classes de objetos podem herdar atributos e serviços de outras classes de objetos.

Objetos

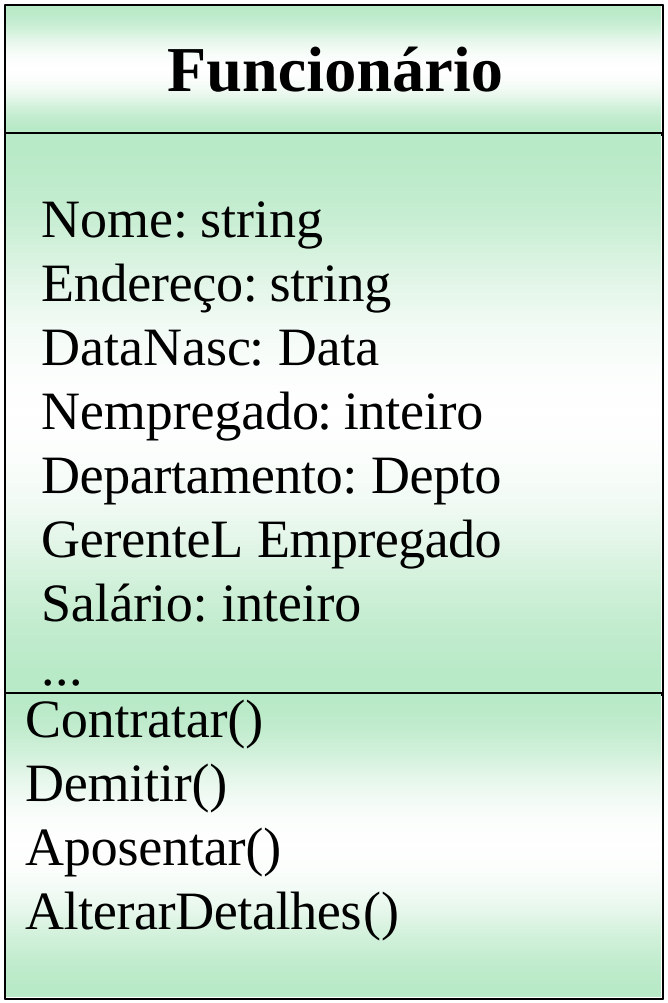
Um **objeto** é uma entidade que possui um estado e um conjunto de operações que operam nesse estado. O estado é representado por um conjunto de atributos. As operações associadas ao objeto fornecem serviços para outros objetos.

Objetos são criados de acordo com uma definição de **classe de objetos**. Uma classe inclui declarações de todos os atributos e serviços que devem ser associados a um objeto dessa classe.

A Linguagem de Modelagem Unificada (UML)

- Várias notações diferentes para descrever projetos orientado a objetos foram propostas nos anos 80 e 90.
- A UML é uma integração dessas notações.
- A UML descreve notações para vários modelos diferentes que podem ser produzidos durante a análise e projeto OO.
- A UML é atualmente um padrão para modelagem OO.

Classe de Objetos funcionário (UML)



Comunicação entre objetos

- Conceitualmente, objetos se comunicam por passagem de mensagem.
- Mensagens
  - O nome do serviço requerido pelo objeto chamador.
  - Cópias da informação necessária para executar o serviço e o nome do possuidor do resultado do serviço.
- Na prática, mensagens são implementadas como chamadas de procedimentos.
  - Nome = nome do procedimento
  - Informação = lista de parâmetros

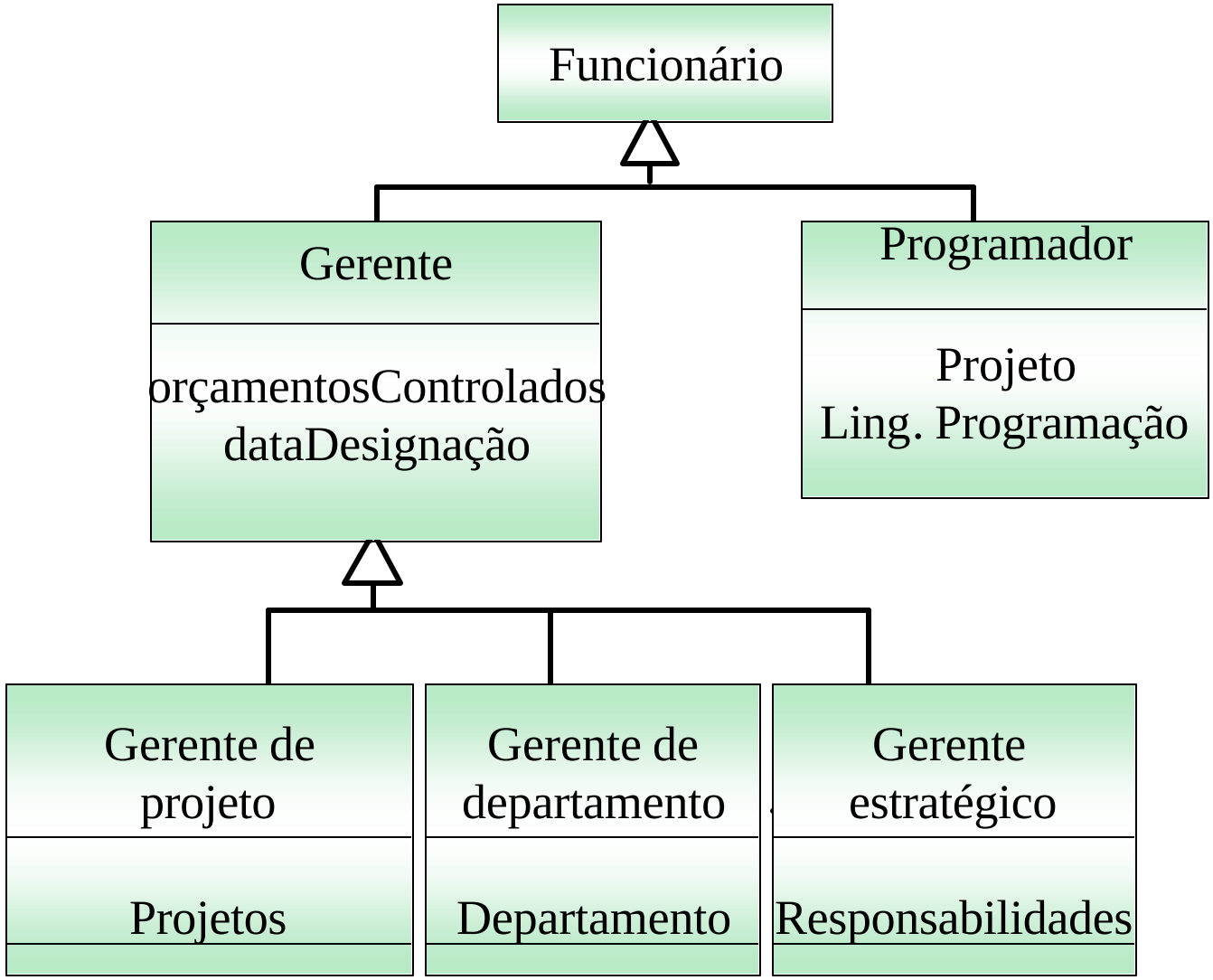
Exemplos de mensagens

```
// Chamar um  método associado com um
// objeto buffer que retorna o próximo valor no
// buffer
                    v = circularBuffer.Get () ;
// Chamar o método associado a um objeto
// termostato que ajuste a temperatura a ser
// mantida
                    termostado.setTemp (20) ;
```

Generalização e herança

- Objetos são membros de classes, as quais definem tipos de atributos e operações.
- Classes são organizadas em uma hierarquia de generalização ou herança, que mostra o relacionamento entre as classes gerais gerais ou mais específicas.
- Uma subclasse herda os atributos e operações de sua superclasse e pode adicionar novos métodos ou atributos a si.
- Generalização em UML é implementada como herança em linguagens de programação OO.

Uma hierarquia de generalização



Vantagens da herança

- É um mecanismo de abstração que pode ser usado para classificar entidades.
- É um mecanismo de reutilização tanto a nível de projeto quando de programação.
- O grafo de herança é uma forma de organizar o conhecimento sobre o domínio e os sistemas.

Problemas com herança

- Classes de objetos não são auto-contidas. Não podem ser entendidas sem fazer referência à suas superclasses.
- Projetistas podem querer reutilizar os grafos de herança criados durante a análise e isso pode levar a ineficiência.
- Os grafos de herança da análise, projeto e implementação tem funções diferentes e devem ser mantidos separadamente.

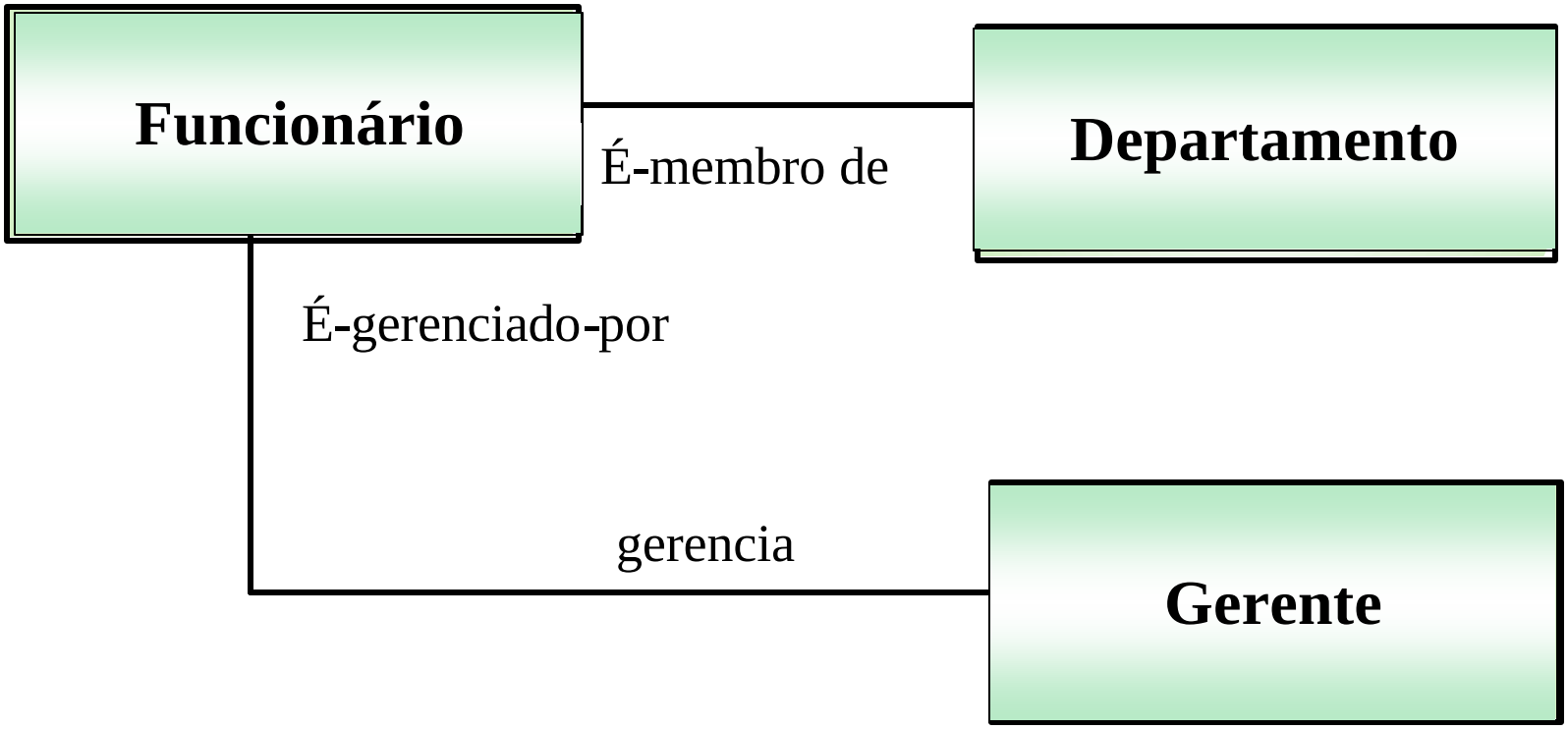
Herança e Projeto OO

- Existem pontos de vista diferentes sobre o uso de herança em projeto OO.
  - 1. Identificar hierarquia de herança é uma parte fundamental de projeto OO e isso obviamente só pode ser implementado usando uma LPOO.
  - 2. Herança é um conceito útil na fase de implementação, que permite o reuso de definições de atributos e operações. Identificar herança no estágio de projeto introduz restrições desnecessária na implementação.
- Herança introduz complexidade e isso não é desejável, principalmente em sistemas críticos.

Associações UML

- Objetos e classes de objetos participam em relacionamentos com outros objetos e classes de objetos.
- Em UML, as associações são denotadas por uma linha entre as classes de objetos.
- Associações podem ser anotadas com informações sobre a associação.
- Associações são relacionamentos gerais, mas podem indicar que o atributo de um objeto é um objeto associado ou que a implementação de um método de objeto conta com o objeto associado.

Um modelo de associação



Objetos concorrentes

- Objetos podem interagir entre si de forma que sejam executados simultaneamente como processos paralelos.
- Através de um mecanismo muito simples (*thread* em Java) é permitido a criação de objetos que podem ser executados simultaneamente.
- Existem dois tipos de implementação de objetos simultâneos: Servidores e Objetos Ativos.

Objetos Servidores e Objetos ativos

- Servidores.
  - O objeto é implementado como um processo paralelo (servidor), com métodos correspondentes às operações definidas pelo objeto. Quando completam sua operação, o objeto interrompe sua execução e aguarda por outras requisições de serviço.
- Objetos ativos.
  - O estado do objeto pode ser modificado por operações internas em execução dentro do próprio objeto. O processo que representa o objeto executa continuamente essas operações e, assim, nunca interrompe a si próprio.

Objetos servidores

- Usados quando o serviço requisitado leva algum tempo para ser concluído (tanto em ambientes distribuídos como também em uma única máquina).

Objetos ativos

- Usados quando precisam atualizar constantemente o seu próprio estado em intervalos específicos.
- Comum em STR, quando objetos estão associados a hardware que coletam informações sobre o ambiente do sistema.

Exemplo de Objeto ativo

- O objeto ativo ***transponder*** mantém o controle da posição de uma aeronave utilizando uma sistema de navegação por satélite. O método *run* (em java) traz o código para calcular a posição da aeronave, utilizando sinais de satélite.

Implementação do objeto transponder em Java

```
class Transponder extends Thread {

        Position currentPosition ;
        Coords c1, c2 ;
        Satellite sat1, sat2 ;
        Navigator theNavigator ;

        public Position givePosition ()
        {
             return currentPosition ;
        }

        public void run ()
        {
             while (true)
             {
                  c1 = s at1.position () ;
                  c2 = sat2.position () ;
                  currentPosition = theNavigator.compute (c1, c2) ;
             }

        }

} //Transponder
```

Threads java

- Threads em Java são um mecanismo muito simples, que permite criar objetos que são executados simultaneamente.
- As Threads devem incluir um método chamado run(), que é inicializado pelo sistema em tempo de execução Java.
- Objetos ativos, tipicamente, incluem um laço infinito de forma a estarem sempre em execução.

Processo de projeto OO

- 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 do Sistema 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.

Descrição da Estação Meteorológica

Uma estação meteorológica é um pacote de instrumentos (termômetros, barômetros, etc.) controlados por software que coleta dados, realiza alguns processamentos de dados e transmite esses dados para outros processamentos. Os dados são coletados a cada cinco minutos.

Quando um comando é dado para transmitir os dados meteorológicos, a estação meteorológica processa e resume os dados coletados. O dados resumidos são transmitidos para o computador, quando um pedido para tal é recebido.

Descrição da Estação Meteorológica (Principais subsistemas)

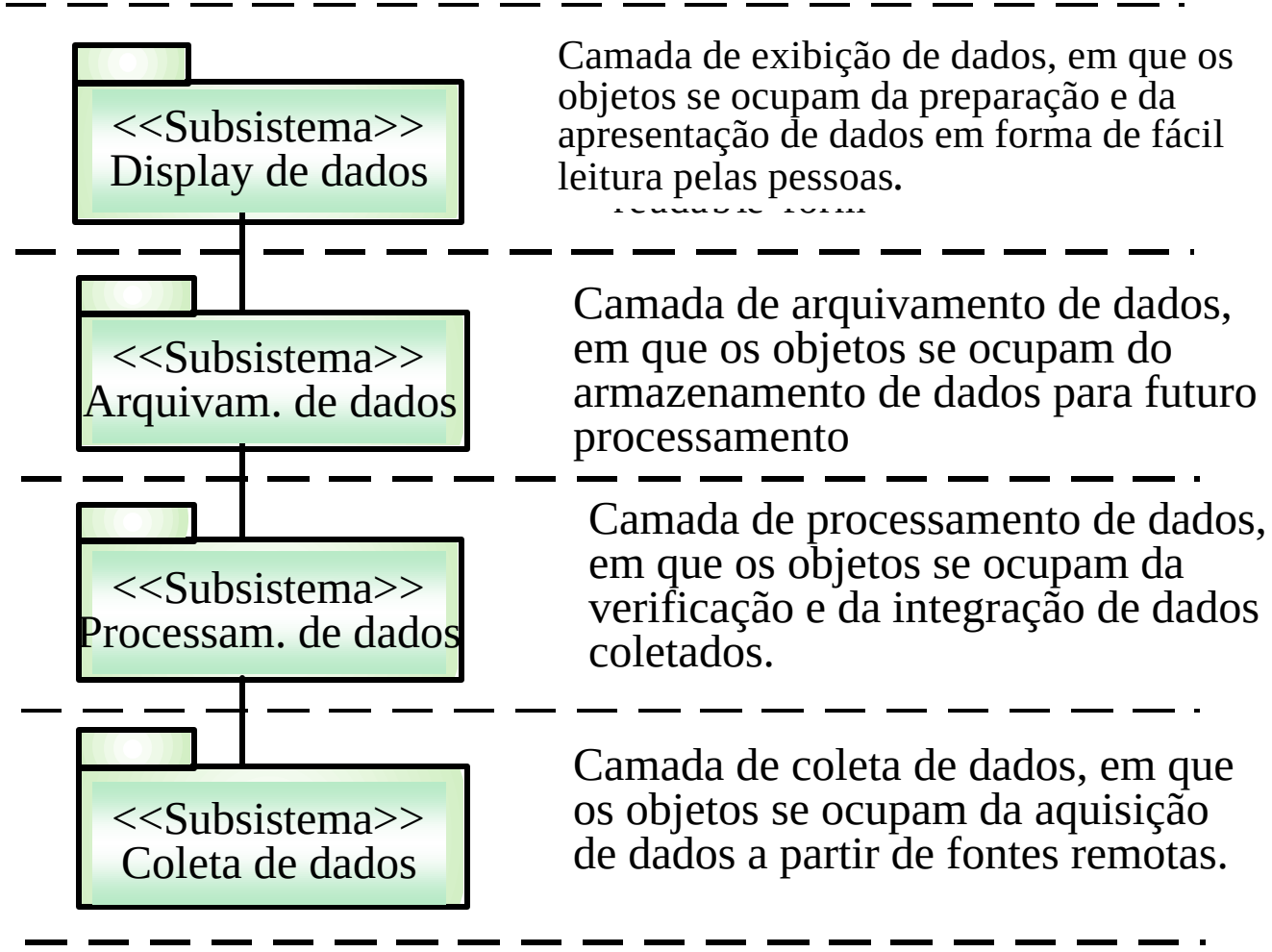
Coleta de dados

Integração de Dados (processamento)

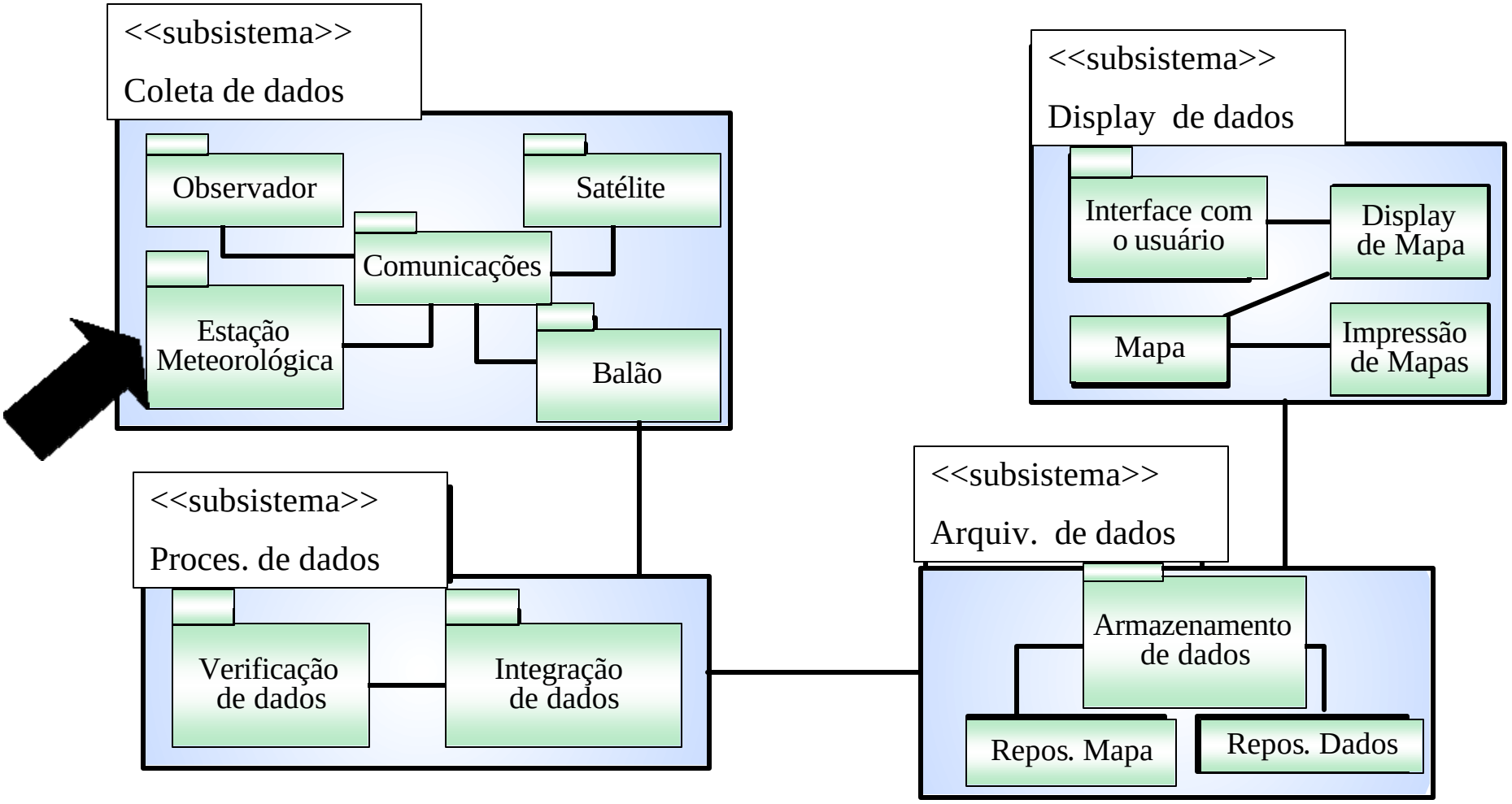
Arquivamento De dados

Criação de Mapas

Uma possível arquitetura – Arquitetura em Camada



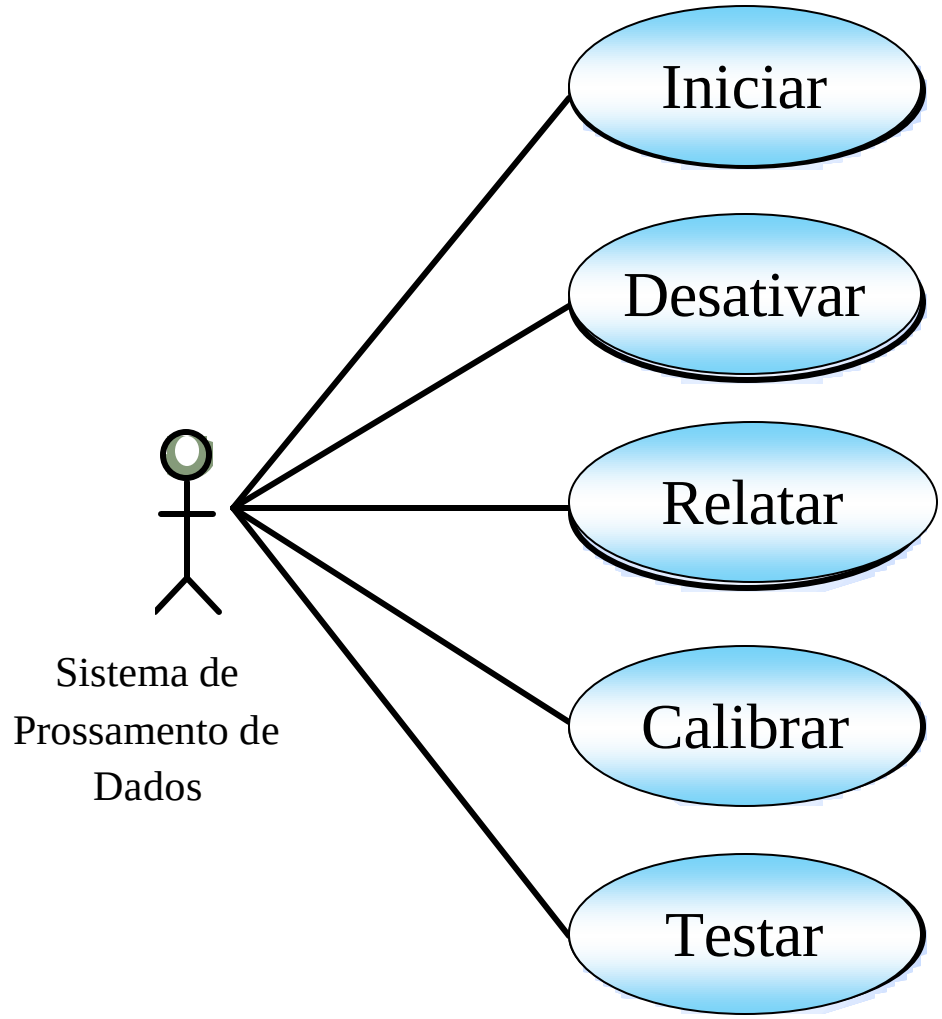
Subsistemas em um sistema de mapeamento meteorológico



Contexto do sistema e modelos de uso

- Desenvolver uma compreensão das relações entre o software que está sendo projetado e seu ambiente externo.
- Contexto do sistema
  - Um modelo estático que descreve os outros sistemas naquele ambiente. (ilustração anterior)
- Modelo de uso do sistema
  - Um modelo dinâmico, que descreve como o sistema realmente interage com seu ambiente. Pode-se usar casos de uso para mostrar essa interação.

Casos de uso para a estação meteorológica



Descrição do caso de uso Relatar dados climáticos

Sistema	Estação Meteorológica
Use-case	Relatar
Agentes	Sistema de processamento de dados sobre o clima, Estação meteorológica.
Dados	A estação meteorológica envia para o sistema de processamento de dados climáticos um resumo de 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 ar; às velocidades máxima, mínima e média do vento, conforme amostragem a cada intervalo de cinco minutos
Estímulo	O sistema de processamento de dados sobre o clima estabelece um link de modem com a estação meteorológica e requisita a transmissão dos dados
Resposta	Os dados são resumidos pelo sistema de coleta de dados sobre o clima e enviados ao sistema de processamento de dados.
Comentários	Em geral, as estações meteorológicas recebem um pedido de relatório por hora, mas essa freqüência pode diferir de uma estação para outra a ser modificada no futuro.

casos de uso

- É preciso desenvolver descrições para todos os casos de uso representados no modelo de caso de uso.
- Utilidade de casos de uso
  - Identificar objetos no sistema
  - Identificar operações no sistema

No exemplo em questão:

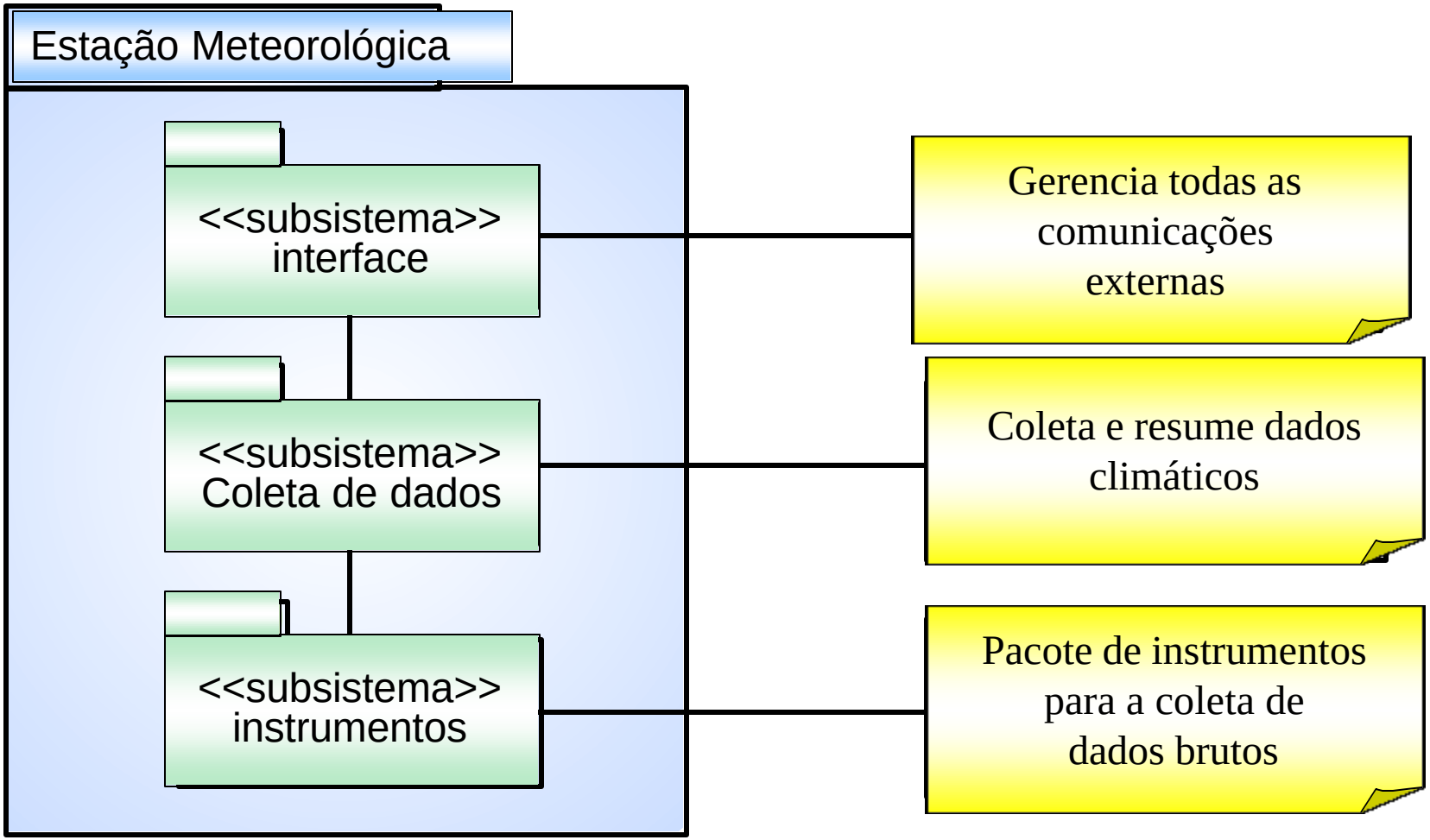
Objetos necessários: objetos que representem instrumentos que coletam dados e um objeto que faz o resumo dos dados

Operações necessárias: operações para requisitar e enviar dados sobre o clima

Projeto de Arquitetura

- Uma vez definidas as interações entre o sistema que está sendo projetado e o seu ambiente, pode-se utilizar essas informações para estabelecer a arquitetura do sistema.
- Uma arquitetura em camadas é apropriada para a estação meteorológica.
  - A camada de Interface para manipular comunicações.
  - Camada de coleta de dados para gerenciar a coleta de dados a partir dos instrumentos e resumir os dados antes da transmissão.
  - A camada de instrumentos que encapsula todos os instrumentos.

Arquitetura da estação metereológica



Identificação de objetos

- Identificar objetos (ou classes de objetos) é a parte mais difícil de projeto OO.
- Não existe uma “fórmula mágica” para a identificação de objeto. É preciso que o projetista tenha habilidade, experiência e conhecimento do domínio do sistema.
- A identificação de objeto é um processo iterativo. É improvável que se obtenha todos os objetos num primeiro esboço.

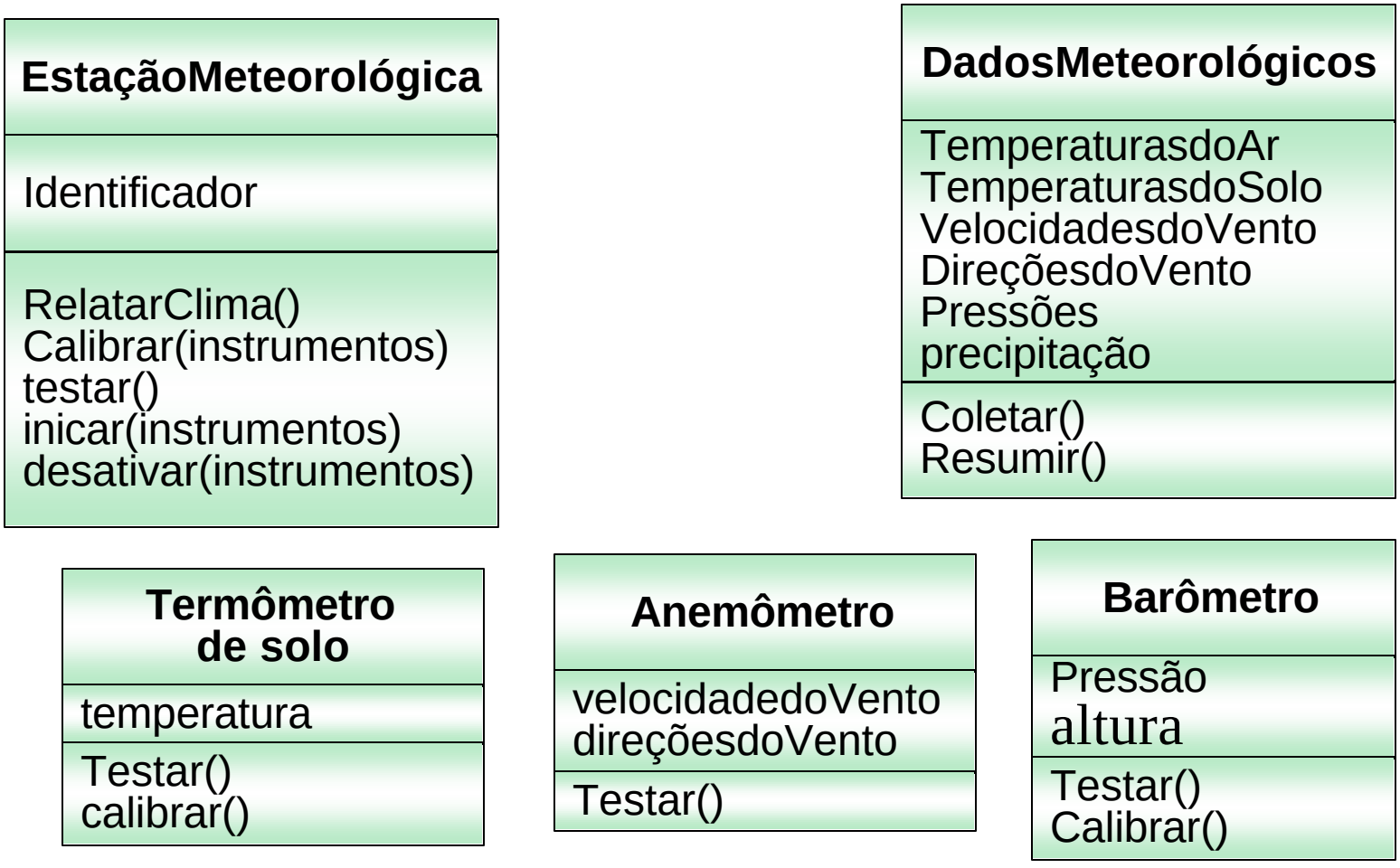
Abordagens para Identificar classes de objetos

- Utilize uma análise gramatical baseada em uma descrição em linguagem natural do sistema.
- Utilize entidades tangíveis (coisas); funções; eventos; locais; interações no domínio da aplicação.
- Utilize uma abordagem comportamental onde se analisa o comportamento do sistema Os participantes que desempenham papéis ativos são candidatos a objetos.
- Utilize uma análise baseada em cenários. Os objetos, atributos e métodos em cada cenário são identificados. (Cartões CRC).

Classes de objetos da estação meteorológica

- Termômetro de solo, Anemômetro, Barômetro
  - Objetos do domínio da aplicação que são entidades tangíveis de hardware relacionadas aos instrumentos no sistema. As operações se ocupam de controlar esse hardware.
- Estação meteorológica
  - É 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 dos diferentes instrumentos na estação meteorológica. Suas operações associadas se ocupam de coletar e resumir os dados que são requeridos.

Classes de objetos da estação meteorológica



Outros objetos e refinamentos de objetos

- Utilize o conhecimento do domínio do problema para identificar outros objetos e serviços.
  - Estações meteorológicas devem ter um identificador único.
  - Estações meteorológicas são localizadas em lugares remotos, assim falhas nos instrumentos devem ser registradas automaticamente. Portanto atributos e operações são necessários para verificar o funcionamento correto dos instrumentos.
- Objetos passivos ou ativos
  - Nesse caso, os objetos são passivos e a coleta de dados é feita quando necessário, e não de forma autônoma. Isso introduz flexibilidade ao realizar mudanças na estratégia de coleta, sem modificar os objetos associados aos instrumentos.

Modelos de projeto

- Modelos de projeto mostram as classes de objetos e os relacionamentos entre elas.
- Diferentes modelos com diferentes níveis de detalhes são desenvolvidos.
- Modelos estáticos descrevem a estrutura estática do sistema em termos de classes de objetos e relacionamentos.
- Modelos dinâmicos descrevem as interações dinâmicas entre os objetos do sistema.

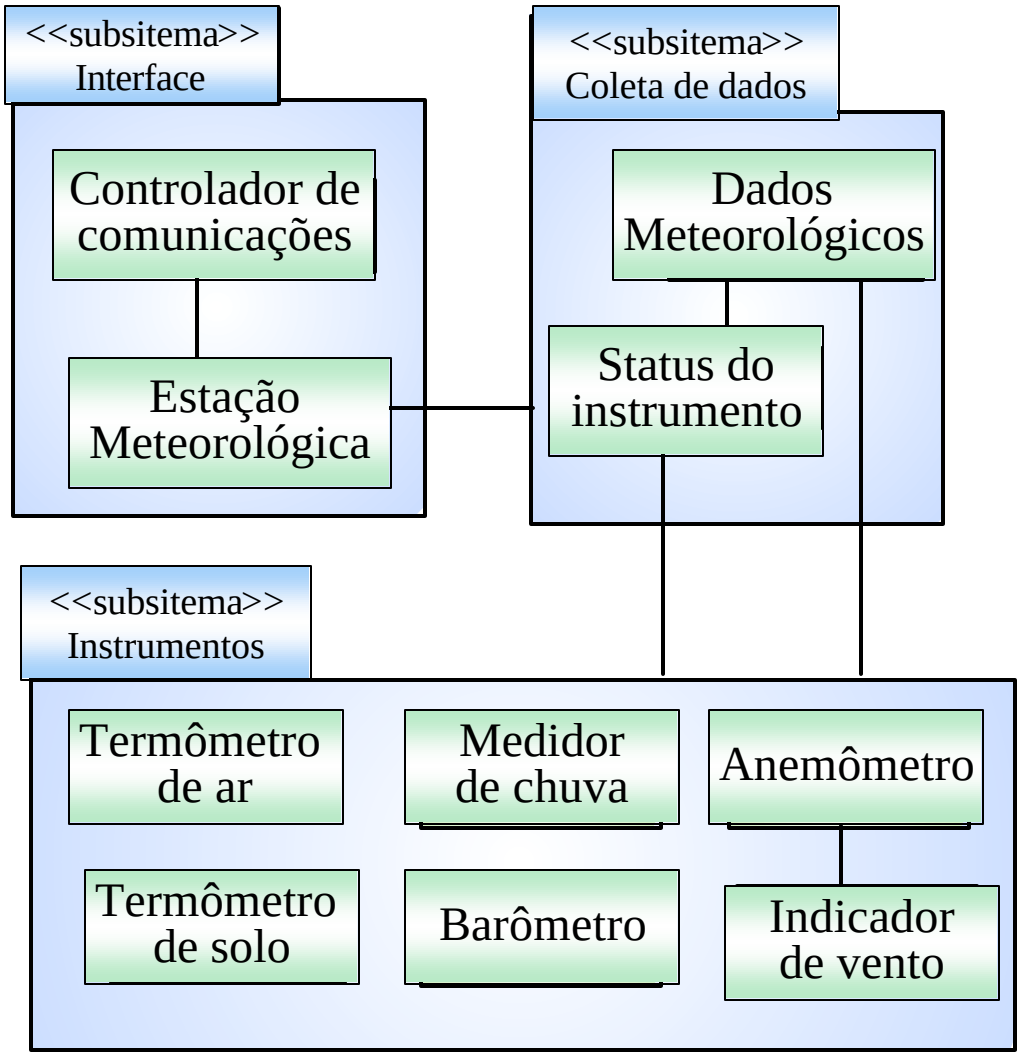
Exemplos de modelos de projeto

- Modelos de subsistema, que mostram agrupamentos lógicos de objetos em subsistemas coerentes.
- Modelos de Seqüência, que mostram a seqüência das interações entre objetos.
- Modelos de máquina de estados, que mostram as mudanças de estado de objetos individuais, em resposta a eventos.
- Outros modelos inclui modelos de caso de uso, modelos de objetos, modelos de agregação, modelos de generalização, etc.

Modelos de subsistemas

- Mostram como o projeto está organizado em termos de grupos de objetos logicamente relacionados.
- Na UML, são mostrados usando pacotes - uma construção encapsulada. É um modelo lógico.

Subsistemas da estação meteorológica



Modelo de seqüência

- Modelo de seqüência mostra a seqüência de interações de objetos que acontecem.
  - Objetos envolvidos são organizados horizontalmente, com uma linha vertical ligada a cada objeto.
  - O tempo é representado verticalmente, assim os modelos são lido de cima para baixo.
  - Interações entre objetos são representadas por setas rotuladas. As setas representam mensagens ou eventos, que são fundamentais para a interação.
  - Um retângulo estreito na linha de um objeto representa o tempo pelo qual o objeto é o objeto controlador no sistema.

Seqüência de operações - coleta de dados

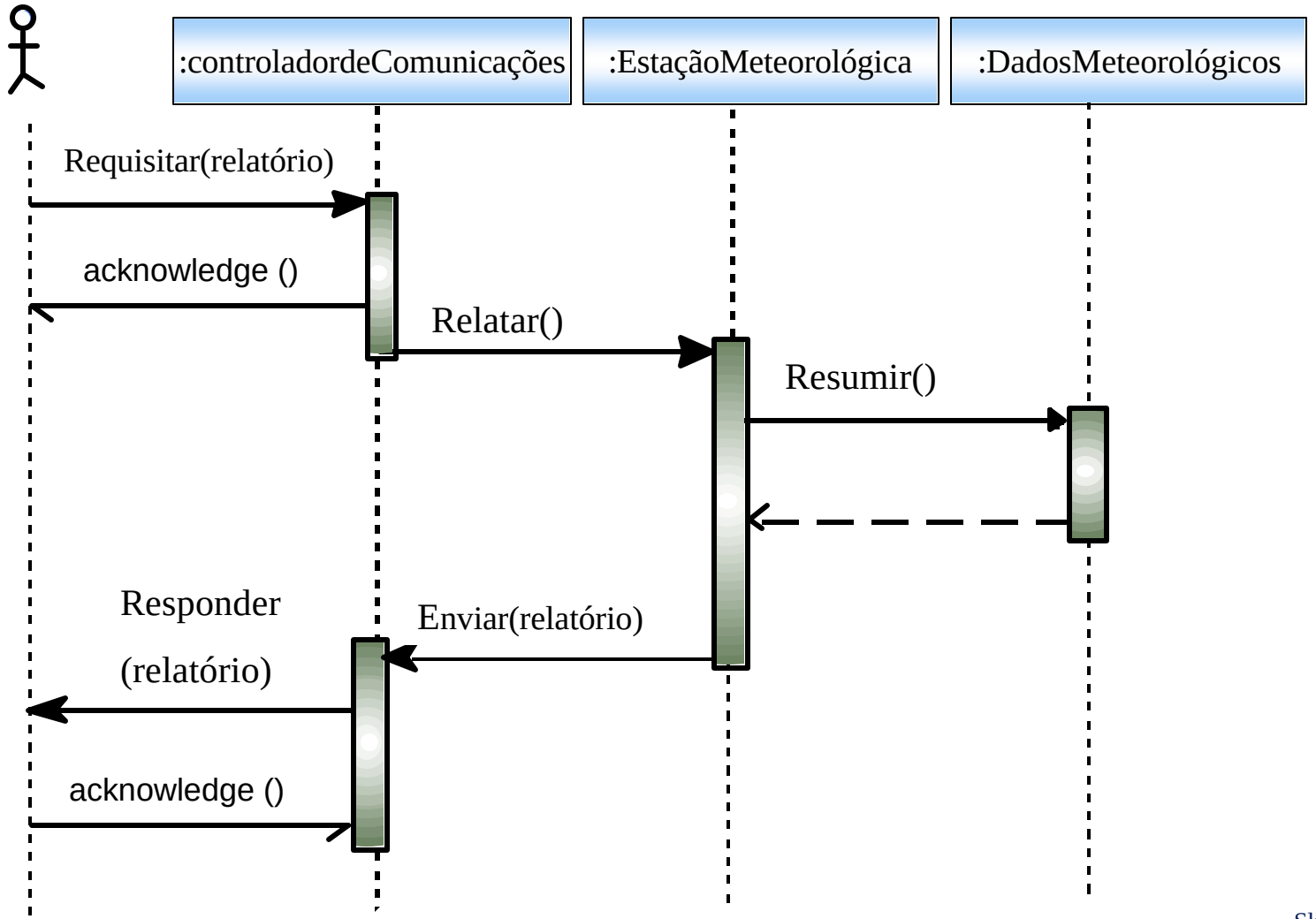


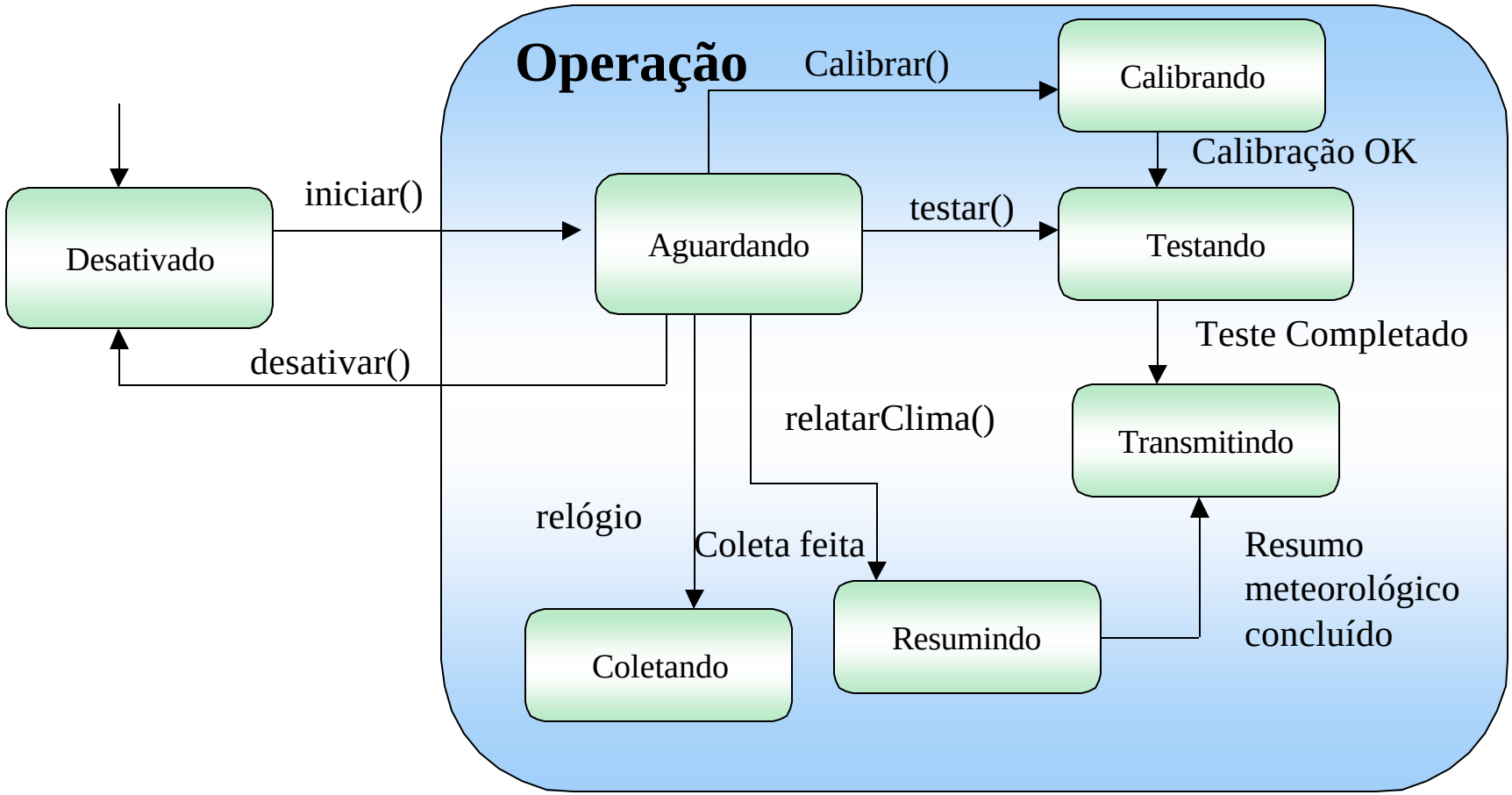
Diagrama de seqüência

- É preciso produzir um diagrama de seqüência para cada interação significativa.
- Deve haver um diagrama de seqüência para cada caso de uso identificado.
- DS é usado para modelar o comportamento combinado de em grupo de objetos.

Statecharts (Harel 87)

- Através de um *statechart* pode-se mostrar o comportamento de um único objeto em resposta a diferentes mensagens que ele pode processar.
- Basicamente, é um modelo de máquina de estado que mostra como a instância do objeto muda de estado, dependendo das mensagens que ele recebe.

Statechart para o objeto Estação Meteorológica



Especificação de interface entre objetos

- Interfaces devem ser especificadas para que os objetos e outros componentes possam ser projetados em paralelo.
- Projetistas devem evitar informações de representação de interface em seus projetos de interface. A representação deve ser oculta e as operações de objeto devem ser fornecidas.
- O mesmo objeto pode ter várias interfaces que são pontos de vista dos métodos que elas fornecem. (Compatível com java, onde as interfaces são declaradas separadamente dos objetos e os objetos “implementam interfaces”)

Projeto de interface entre objetos

- É a especificação dos detalhes da interface para um objeto ou um grupo de objetos.
- Significa definição das assinaturas e a semântica definida pelos serviços oferecidos pelos objetos.
- Em UML, define-se interfaces usando a mesma notação dos diagramas de classe, onde acrescenta-se o estereótipo <<interface>> na parte do nome da classe.
- Abordagem alternativa: usar uma LP para definir a interface.

Interface da estação meteorológica

```
interface Estação Meteorológica {

  public void EstaçãpMeteorológica () ;

  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 obtertID () ;

} //EstaçãoMeteorológica
```

Evolução de projeto

- Uma vantagem da abordagem OO é simplificar o problema de fazer mudanças no projeto
- O ocultamento de informação dentro dos objetos permite que alterações feitas em um objeto não afetem outros objetos de forma imprevisível.

Exemplo da robustez da abordagem OO

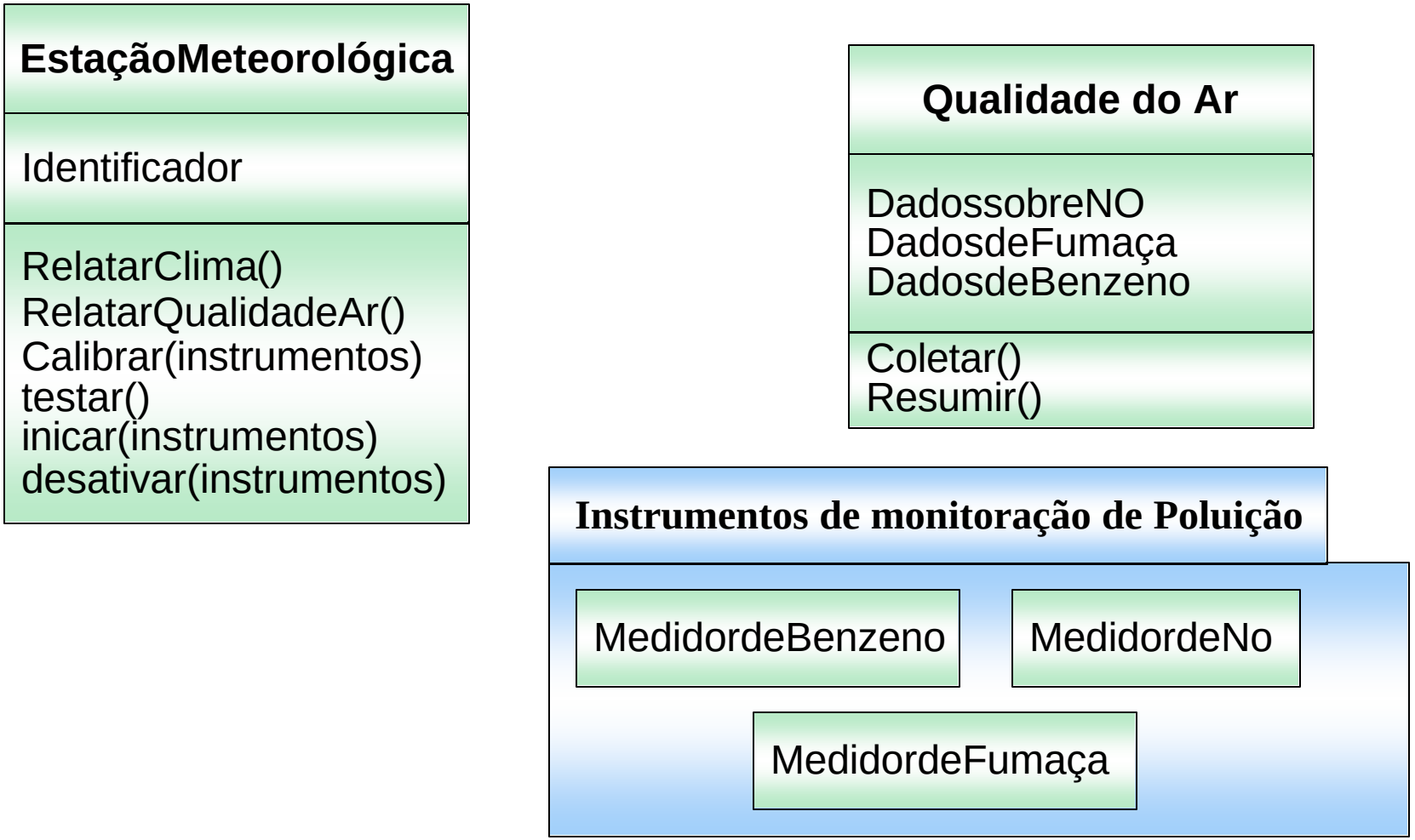
Suponha que a monitoração da poluição do ar será adicionada nas estações meteorológicas. Isso envolve adicionar um medidor de qualidade do ar para computar a concentração de vários poluentes na atmosfera.

Assim, leituras de poluição são transmitidas ao mesmo tempo que os dados meteorológicos.

Alterações necessárias

- Adição uma classe de objetos chamado “Qualidade do ar” como parte da Estação Meteorológica, no mesmo nível que DadosMeteorológicos.
- Adição de uma operação “RelatarQualAr” à Estação Meteorológica. Modificar o software de controle para coletar leituras de poluição.
- Adição de objetos representado instrumentos para monitorar a poluição.

Novos objetos para monitorar a poluição



Pontos Chave

- POO é um meio de projetar sofware de modo que os componentes possuem seus próprios estados e operações.
- Objetos devem ter operações de construção e inspeção. Eles fornecem serviços a outros objetos.
- Objetos podem ser implementados seqüencialmente ou concorrentemente
- A UML oferece diferentes notações para definir diferentes modelos de objetos.

Pontos Chave

- Uma série de diferentes modelos podem ser produzidos durante um processo de projeto OO, incluindo modelos estáticos e modelos dinâmicos do sistema.
- Interfaces com objetos precisam ser definidas precisamente, usando, por exemplo, uma linguagem de programação como Java.
- Uma das principais vantagens do projeto orientado a objeto é o fato de simplificar a evolução do sistema.