Progettazione orientata ad oggetti

Alberto Gianoli

Univ. Ferrara - Corso di Laurea in Informatica

Dove?

- Pressman, qualcosa dal cap. 6
- Sommerville, cap. 14

Progettazione orientata ad oggetti

- La progettazione software può essere rappresentata come un insieme di oggetti che interagiscono
- OOD (object oriented design) traduce OOA(object oriented analysis) in un modello specifico di implementazione, che possa essere codificato in un linguaggio OOP (object oriented programming)
- In particolare vedremo due metodologie per OOD:
  - Coad & Yourdon
  - OMT (Rambaugh)

Parole chiave dell'approccio OO

- **Classe**
  - denota le caratteristiche comuni di entità che hanno uno stato e un insieme di operazioni che operano su quello stato
  - un oggetto è una istanza di una classe
- **Attributi**
  - lo stato di un oggetto è rappresentato da un insieme di attributi
- **Metodi**
  - le operazioni definite su un oggetto, che offrono servizi ad altri oggetti
- **Incapsulamento**
- **Ereditarietà** (relazione tra classi, non tra oggetti!)
- **Specializzazione** (overriding)
- **Polimorfismo** (overloading)

Sviluppo orientato ad oggetti

- L'**analisi**, la **progettazione** e la **programmazione** object oriented sono legate ma distinte
- **OOA** riguarda lo sviluppo di un modello ad oggetti del dominio di applicazione
- **OOD** riguarda la progettazione di un modello orientato ad oggetti del sistema che soddisfi i requisiti
- **OOP** riguarda la realizzazione del progetto usando uno specifico linguaggio orientato ad oggetti (C++, Java, ...)

OOA & OOD

- **OOA** è essenzialmente una attività di **classificazione**: il problema viene analizzato e vengono determinate le **classi** e le relazioni tra classi
- **OOD** indica gli **oggetti** che sono derivati da ogni classe, come **interagiscono** tra di loro, come si realizzano le relazioni tra oggetti, come se ne implementa il **comportamento**, come viene implementata la **comunicazione** tra oggetti

OOA & OOD

ANALISI	DESIGN
classi	oggetti
attributi	strutture dati
metodi	algoritmi
relazioni	passaggio di messaggi
comportamento	controllo

Esempio

- Nel documento dei requisiti di un sistema di allarme potremmo avere:
  - "ogni sensore ha assegnato un numero e un tipo"
  - "si può programmare il sistema, utilizzando una password, per attivare o disattivare il sistema"
- Nella OOA si può derivare:
  - l'operazione di assegnamento (di numero e di tipo) è rilevante per l'oggetto sensore
  - l'operazione di programmazione è rilevante per l'oggetto sistema
  - le operazioni di attivazione e disattivazione si applicano all'oggetto sistema, il cui stato può essere definito come attivato/disattivato

Esempio

- Nella fase di OOD
  - ogni operazione viene definita in termini di operazioni più specifiche, che permettano di configurare il sistema
  - p.e.: l'operazione di "programmazione" del sistema di allarme può essere definita così
    - ➡ il programma consente all'utente di configurare il sistema una volta installato
    - ➡ l'utente può: (1) installare numeri telefonici d'emergenza; (2) definire un tempo di attesa prima di attivare l'allarme; (3) costruire una tabella dei sensori, che contenda la ID di ogni sensore, il suo tipo e la sua collocazione; (4) caricare una password
  - In questo modo la singola operazione "programmare" è definita in termini di altre operazioni (installare, definire delay, costruire tabella, caricare password)

Metodo OOD

- Identificare oggetti, attributi e servizi (metodi)
- Organizzare gli oggetti in una gerarchia di ereditarietà (relazione “is-a”)
- Organizzare gli oggetti in una gerarchia di aggregazioni (relazione “part-of”)
- La costruzione di diagrammi che mostrino come i metodi di un oggetto sono usati da altri oggetti
- La specifica delle interfacce degli oggetti

Caratteristiche del OOD

- Gli oggetti sono astrazioni di entità del mondo reale o di entità del sistema e si auto-gestiscono
- Gli oggetti sono indipendenti e contengono informazione sul proprio stato e sulla rappresentazione dei propri dati
- La funzionalità del sistema si esprime in termini di servizi offerti dagli oggetti
- Non esistono aree di memoria condivise. Gli oggetti comunicano chiamando funzioni e passando parametri
- Gli oggetti possono essere distribuiti e possono essere eseguiti sequenzialmente o in parallelo

Livelli di progettazione

- Progettare i sottosistemi
  - rappresentare ognuno dei sottosistemi che permettono al sistema di rispondere ai requisiti
- Progettare le classi e gli oggetti
  - definire le gerarchie di classi, usando astrazione e specializzazione, e la rappresentazione degli oggetti
- Progettare la comunicazione tra oggetti
  - definire i dettagli su come avviene la comunicazione tra oggetti e con le interfacce esterne del sistema
- Progettare le strutture dati e l'implementazione per gli attributi e i metodi di ciascun oggetto

Identificazione degli oggetti

- E' la parte difficile della progettazione a oggetti, ed è sostanzialmente un processo iterativo
- Alcuni metodi:
  - approccio grammaticale basato su una descrizione del sistema in linguaggio naturale
  - basare l'identificazione su entità tangibili del dominio di applicazione
  - usare un approccio comportamentale: identificare gli oggetti in quanto si comportano in un certo modo
  - usare una analisi basata sullo scenario (metodo ObjectOry)

Esempio: un sistema informativo da ufficio

- Il Sistema di Reperimento di informazioni d'ufficio è un archiviatore automatico che può
  - archiviare documenti aventi un certo nome in uno o più indici
  - recuperare documenti
  - mostrare e mantenere indici di documenti
  - archiviare e distruggere documenti
- Il sistema è attivato su richiesta dell'utente e restituisce sempre un messaggio all'utente indicando il successo o il fallimento della richiesta

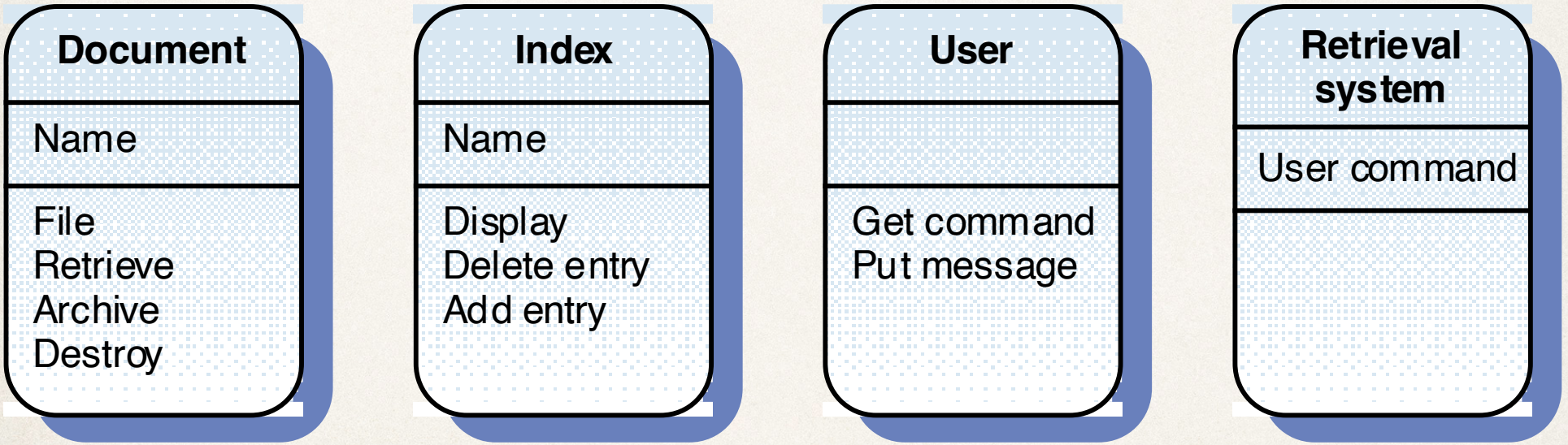
Approccio Grammaticale

- I **nomi** corrispondo ad oggetti del sistema (individuare i sinonimi)
- I **verbi** corrispondono ad operazioni associate ad oggetti
- I **predicati** corrispondono ad operazioni che restituiscono valori di tipo booleano
- L'approccio presuppone che dal documento dei requisiti il progettista abbia una buona conoscenza del dominio di applicazione, in quanto non tutti gli oggetti e i servizi possono essere presenti nella descrizione

Un sistema informativo d'ufficio

- Il **Sistema di Reperimento** di informazioni d'ufficio è un archiviatore automatico che può
  - **archiviare documenti** aventi un certo nome in uno o più **indici**
  - **recuperare documenti**
  - **mostrare** e **mantenere indici** di **documenti**
  - **archiviare** e **distruggere documenti**
- Il **sistema** è attivato su **richiesta** dell'**utente** e **restituisce** sempre un **messaggio** all'**utente** indicando il successo o il fallimento della richiesta

Identificazione preliminare degli oggetti



Tipi di oggetti

- **Entità esterne** (p.e. altri sistemi, persone, device,..) che producono o consumano informazione usata dal sistema
- **Entità interne** (p.e. segnali, displays, dati,..) che sono parte del dominio informativo del sistema
- **Eventi** che occorrono nelle operazioni del sistema
- **Ruoli** delle persone che interagiscono col sistema
- **Luoghi** che stabiliscono il contesto del sistema
- **Strutture** (p.e. sensori, computers, veicoli, ..) costituite da insiemi di entità correlate

Non sono oggetti

- Un oggetto non deve mai identificarsi con un nome di procedura imperativa
- Esempio: “inversione di immagine” non è un oggetto: inversione sarà un metodo dell’oggetto immagine

OOD secondo Coad & Yourdon

1. Componenti del **dominio**
   - ✦ identifica i sottosistemi che sono diretti responsabili della soddisfazioni dei requisiti
   - ✦ elenca tutte le classi specifiche del dominio d'applicazione
   - ✦ **progetta** una gerarchia per tali classi
   - ✦ semplifica l'ereditarietà per quanto possibile
   - ✦ sviluppa una interfaccia con la componente di gestione dei dati

OOD secondo Coad & Yourdon

2. Componenti di **interazione con l'utente**
   - ✦ identifica i sottosistemi che implementano l'interfaccia utente
   - ✦ definisci gli attori umani
   - ✦ sviluppa uno scenario dell'utilizzo del sistema
   - ✦ **progetta** una gerarchia di comandi per l'utente
   - ✦ **progetta** le classi rilevanti e mettile in gerarchia
   - ✦ utilizza classi GUI se necessario

OOD secondo Coad & Yourdon

3. Componenti di **gestione delle mansioni**
   - ✦ identifica i sottosistemi che sono responsabili del controllo
   - ✦ classifica le mansioni (p.e. event-driven, clock-driven, ..)
   - ✦ stabilisci le priorità
   - ✦ identifica le mansioni che fanno da coordinamento per altre mansioni
   - ✦ **progetta** oggetti opportuni per ogni mansione

OOD secondo Coad & Yourdon

4. Componenti di **gestione dei dati**

- identifica i sottosistemi che sono responsabili della memorizzazione e del recupero di oggetti
- **progetta** le strutture dati e i layout
- **progetta** i servizi richiesti per gestire le strutture dati
- identifica gli strumenti utili per implementare la gestione dei dati
- **progetta** classi appropriate e stabilisci una gerarchia tra esse

OOD secondo Rambaugh
OMT (object modeling technique)

1. Realizza la progettazione globale del sistema

- dividi il sistema in sottosistemi
- identifica i problemi di concorrenza
- alloca ogni sottosistema a un corrispondente processo
- scegli una strategia per implementare la gestione dei dati
- identifica le risorse globali e i meccanismi di controllo necessari per accedervi
- progetta un meccanismo di controllo per il sistema
- considera la gestione delle condizioni al contorno

OOD secondo Rambaugh
OMT (object modeling technique)

2. Progetta i singoli oggetti
   - definisci gli algoritmi che realizzano ciascuna operazione
   - seleziona le strutture dati che sono appropriate per gli algoritmi
   - definisci internamente le classi ottimizzando l'accesso ai dati e garantendo l'efficienza di calcolo
   - progetta gli attributi delle classi
3. Progetta i meccanismi di controllo
4. Rivedi la struttura gerarchica delle classi per rendere più stretti i meccanismi di ereditarietà
5. Progetta il passaggio di messaggi per implementare le relazioni tra oggetti
6. "Impacchetta" le classi in moduli

Progettare un oggetto (punto 2)

1. Descrivere il protocollo
   - "cosa" offre l'oggetto
   - stabilire l'interfaccia dell'oggetto, definendo ogni messaggio che l'oggetto può ricevere e le operazioni che l'oggetto può realizzare quando riceve un messaggio
2. Descrivere la realizzazione
   - "come" lo realizza
   - specificare il nome dell'oggetto e la classe di appartenenza
   - specificare gli attributi privati con i corrispondenti tipi
   - descrivere in modo procedurale le operazioni/metodi
     - ➡ operazioni che manipolano lo stato dell'oggetto (aggiungono, cancellano, ..)
     - ➡ operazioni che calcolano qualcosa
     - ➡ operazioni che controllano l'occorrenza di particolari eventi

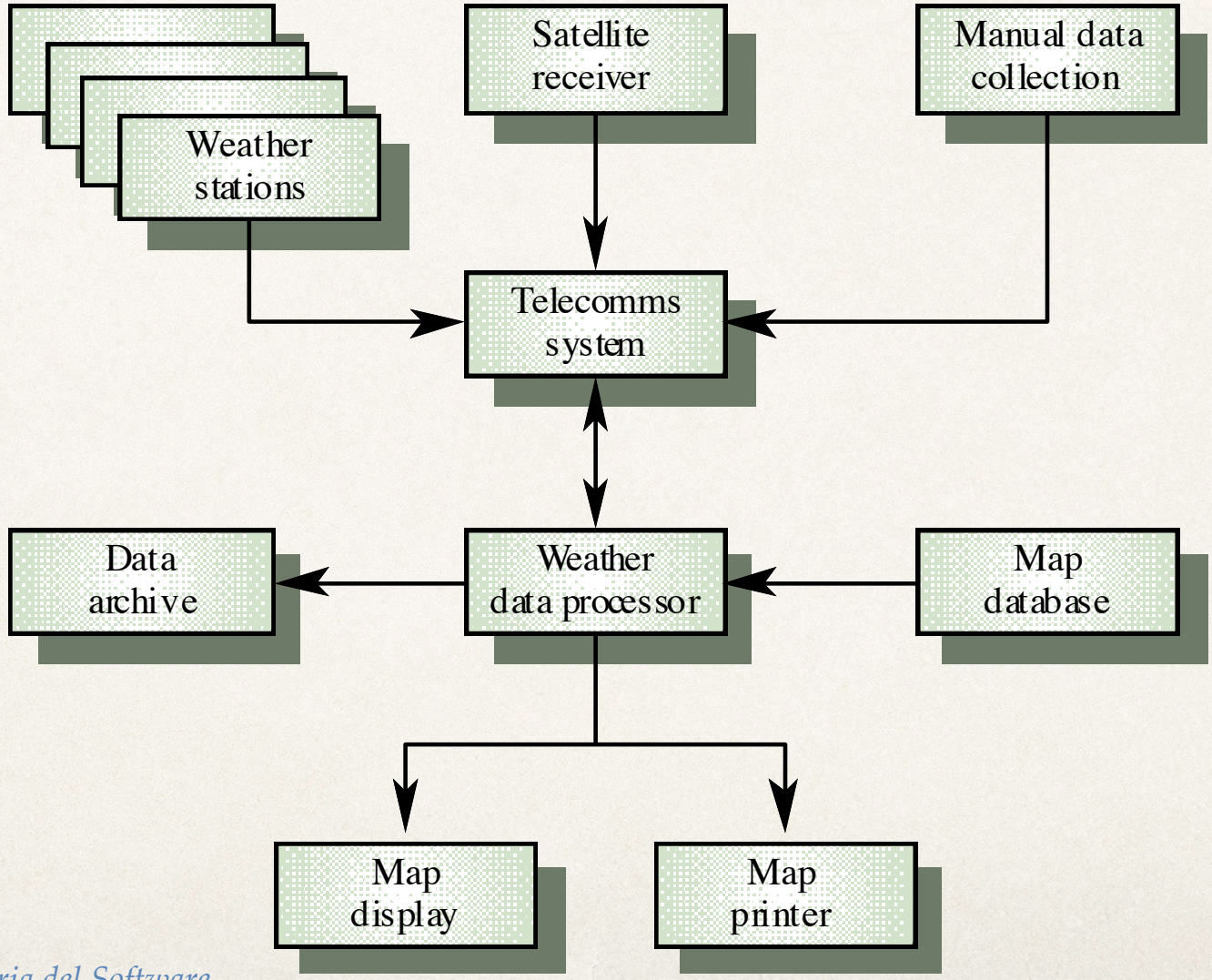
Esempio: sistema di previsioni meteo

Ad un sistema di previsioni meteorologiche viene richiesto di generare periodicamente delle cartine del tempo basate su dati raccolti da stazioni di rilevamento remote.
Ogni stazione di rilevamento raccoglie dati meteorologici relativi ad un certo periodo di tempo e produce un riassunto di tali dati
A richiesta, manda l'informazione raccolta ad un computer d'area perché vengano ulteriormente elaborata.
Ogni stazione di rilevamento raccoglie dati su: temperatura dell'aria, temperatura a terra, velocità e direzione del vento, pressione barometrica, precipitazioni.
Le stazioni trasmettono i loro dati al computer d'area in risposta ad una richiesta da tale macchina.
Il computer d'area mette insieme di dati raccolti e li integra con dati provenienti da altre sorgenti di informazioni (p.e. satellite).
Usando una mappa digitale il computer d'area genera un insieme di previsioni del tempo locali.

Esempio: sistema di previsioni meteo

Ad un **sistema di previsioni meteorologiche** viene richiesto di **generare** periodicamente delle **cartine del tempo** basate su dati **raccolti** da **stazioni di rilevamento** remote.
Ogni stazione di rilevamento raccoglie dati meteorologici relativi ad un certo periodo di tempo e **produce** un **riassunto** di tali dati.
A richiesta, **manda** l'informazione raccolta ad un **computer d'area** perché vengano ulteriormente **elaborata**.
Ogni stazione di rilevamento **raccoglie** dati su: **temperatura dell'aria, temperatura a terra, velocità e direzione del vento, pressione barometrica, precipitazioni.**
Le stazioni **trasmettono** i loro dati al computer d'area in **risposta** ad una **richiesta** da tale macchina.
Il computer d'area **mette insieme** di dati raccolti e li **integra** con dati provenienti da **altre sorgenti di informazioni** (p.e. satellite).
Usando una **mappa digitale** il computer d'area **genera** un insieme di **previsioni** del tempo locali.

Architettura del sistema



Principali classi

- Stazioni di rilevamento
  - insieme di strumenti che collezionano dati, eseguono dei calcoli e trasmettono i dati per ulteriori processi
- Database di carte geografiche
  - permette di generare carte geografiche a scale diverse
- Carta del tempo
  - rappresentazione di un'area con informazioni meteorologiche sovrapposte
- Dati meteorologici
  - usati per produrre le carte e archiviati per processi futuri

Identificazione degli oggetti

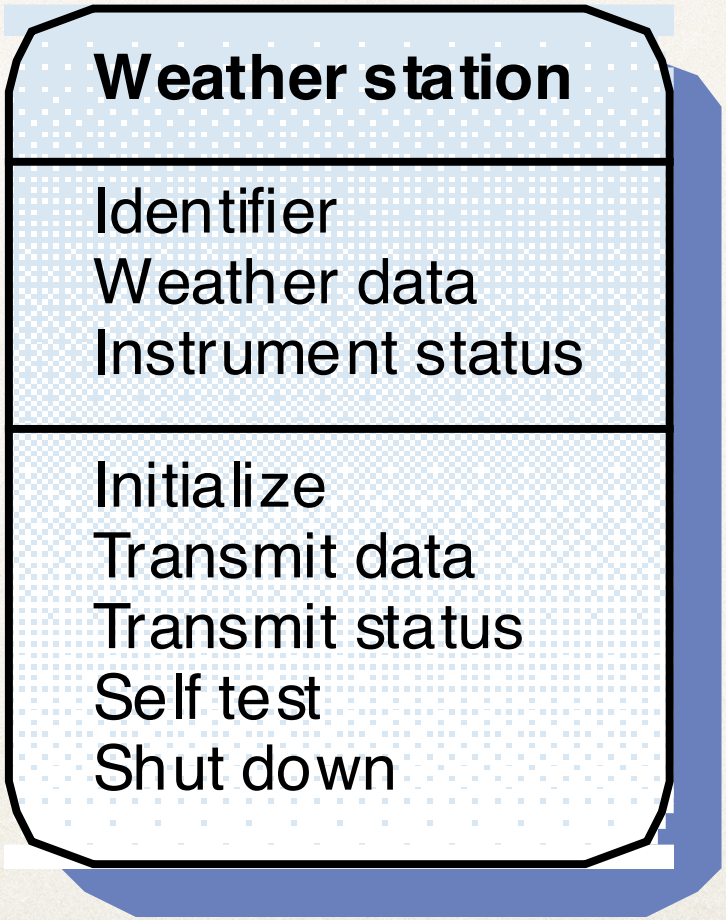
- Una stazione meteorologica è un package di strumenti controllati da software che raccoglie dati, realizza alcune elaborazioni di questi dati e li trasmette per ulteriori elaborazioni.
- Gli strumenti includono termometri (aria e terra), un anemometro, un barometro e un pluviometro. I dati sono raccolti ogni 5 minuti.
- Quando viene dato il comando di elaborazione dei dati la stazione di raccolta elabora i dati e ne crea un sommario. Quest'ultimo viene trasmetto al computer d'area appena ne riceve la richiesta.

Oggetti della stazione meteo

- Oggetti
  - termometro a terra e in aria
  - anemometro
  - barometro
  - pluviometro
- Operazioni
  - colleziona i dati
  - calcola delle operazioni sui dati
  - trasmette i dati
- Attributi
  - dati riassuntivi

Questa descrizione è “ripulita” usando conoscenze sul dominio, p.e. il fatto che una stazione meteo è identificata in modo univoco

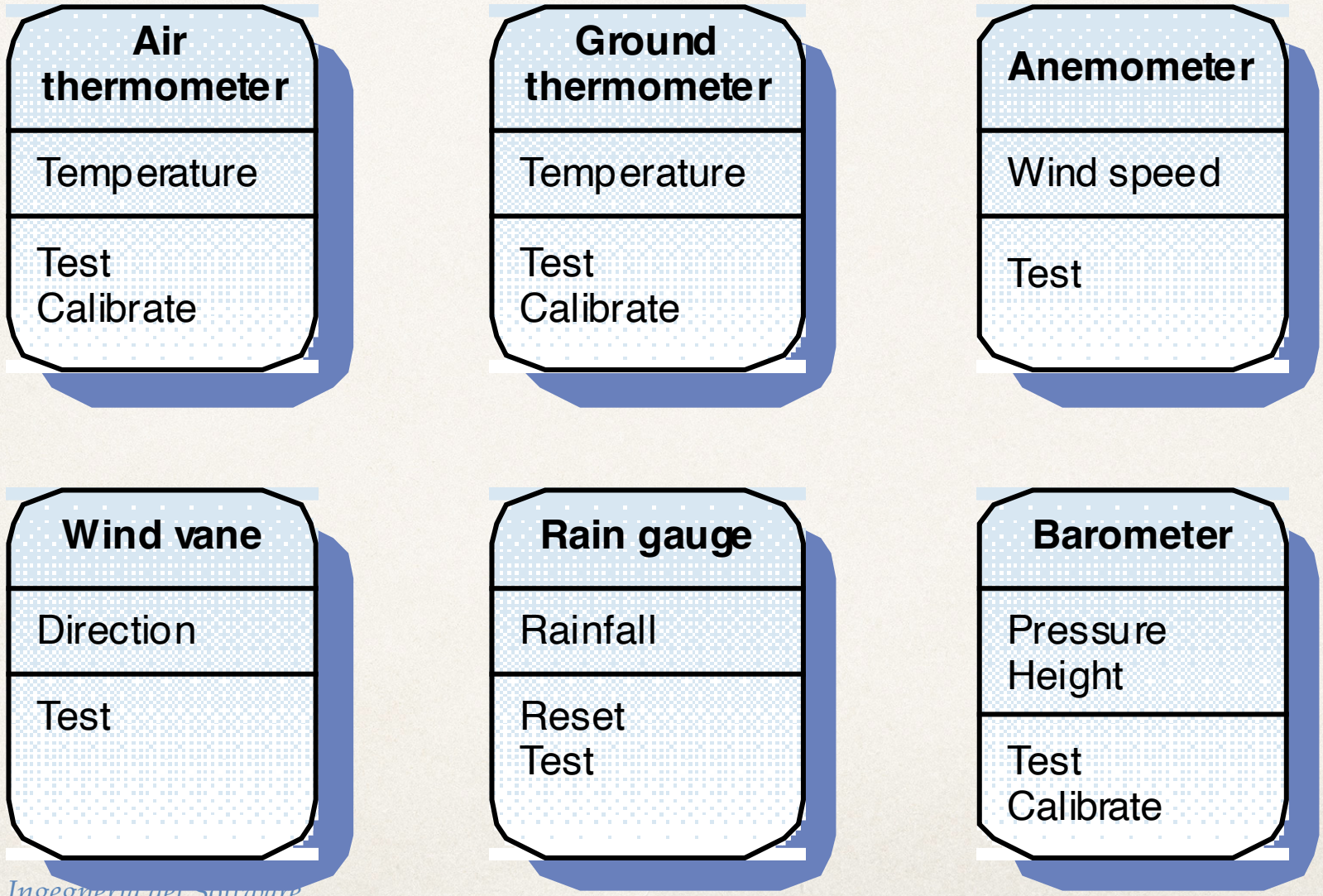
La classe "stazione meteo"



Progettazione degli oggetti

- Gli oggetti per il controllo dell'hardware corrispondono direttamente ai sensori o attuatori connessi al sistema
- Contengono i dettagli sul controllo hardware
- Cambiamenti dell'hardware possono essere introdotti in modo semplice, re-implementando la stessa interfaccia di questi oggetti

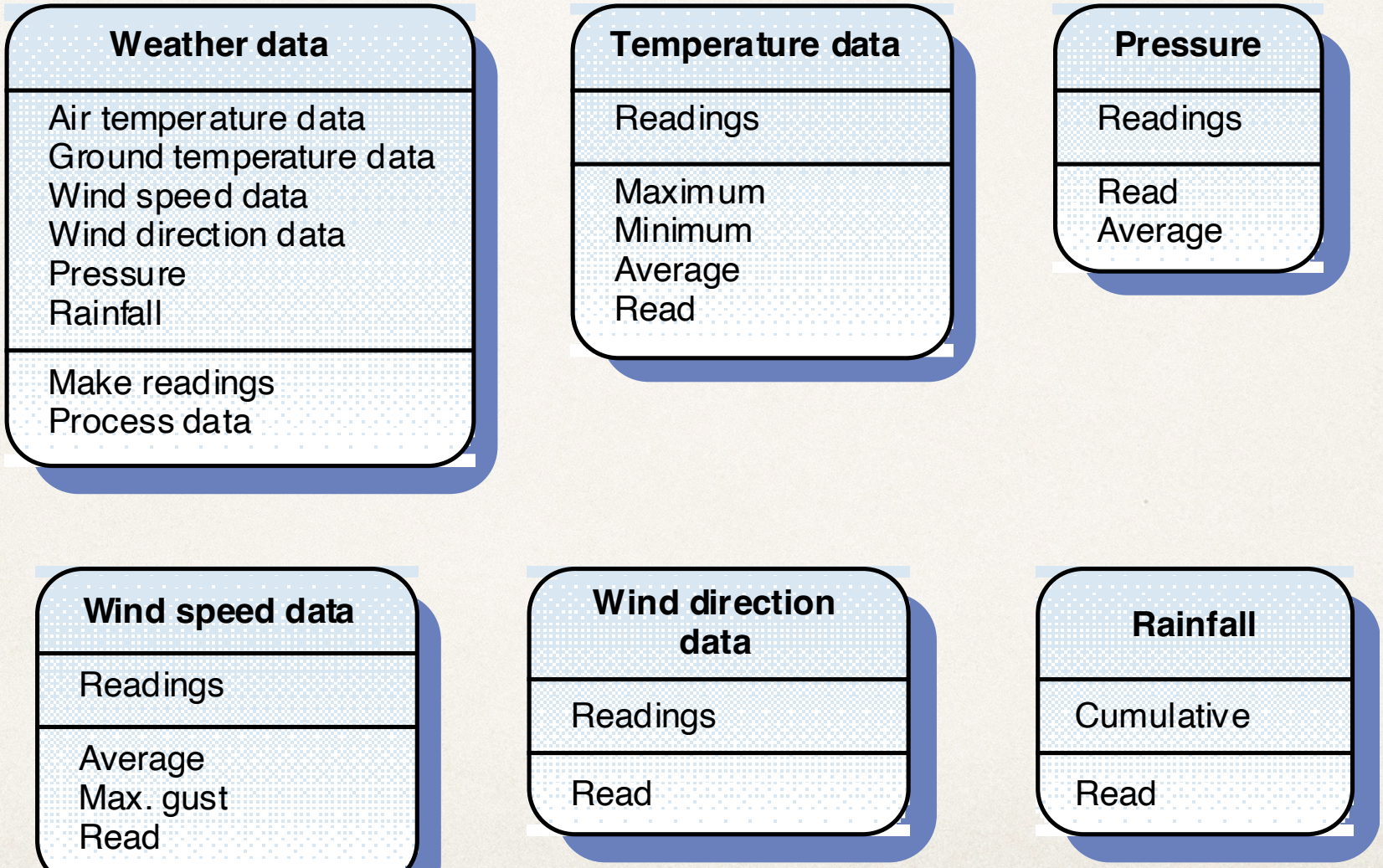
Oggetti per il controllo dell'hardware



Dati raccolti dalla stazione meteo

- Temperatura dell'aria e del suolo
  - valori massimo, minimo e medio
- Velocità del vento
  - valori massimo, minimo e medio
- Direzione del vento
  - ogni 5 minuti durante il periodo di raccolta dati
- Pressione
  - pressione barometrica media
- Precipitazioni
  - valore cumulativo

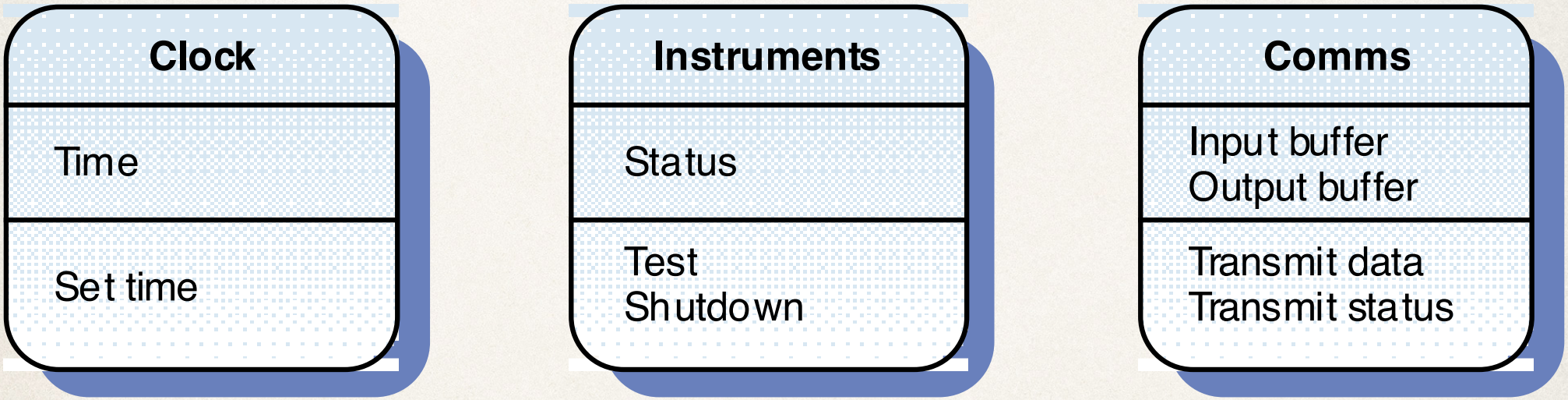
Oggetti che corrispondono ai dati meteorologici



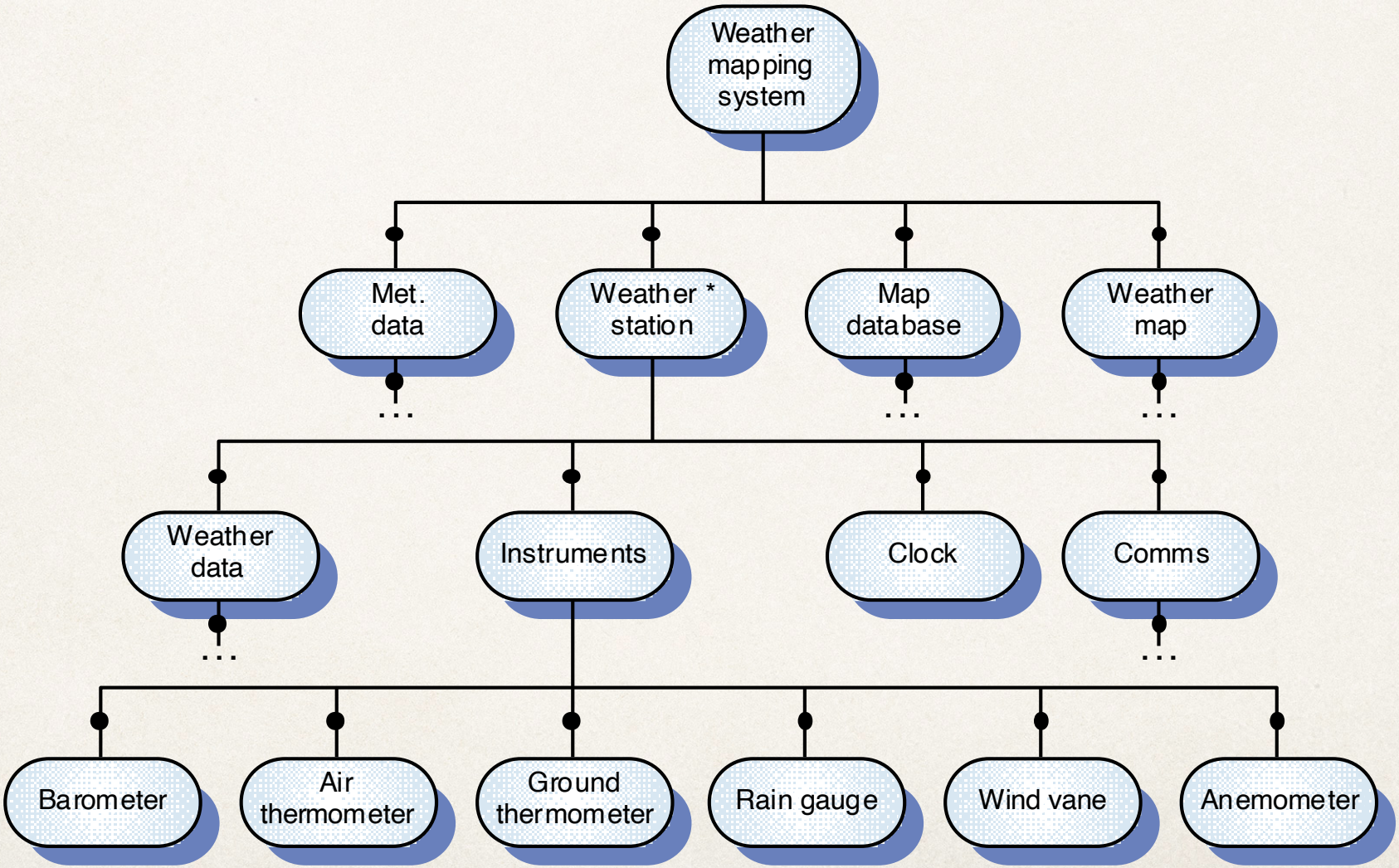
Dati meteorologici

- Tutti i dati meteorologici possono essere incapsulati in uno stesso oggetto
- Gli attributi di un oggetto “dati meteo” sono a loro volta degli oggetti
- L’operazione “Process_data” viene chiamata quando l’informazione meteo sta per essere trasmessa. Calcola l’informazione richiesta usando le informazioni grezze collezionate

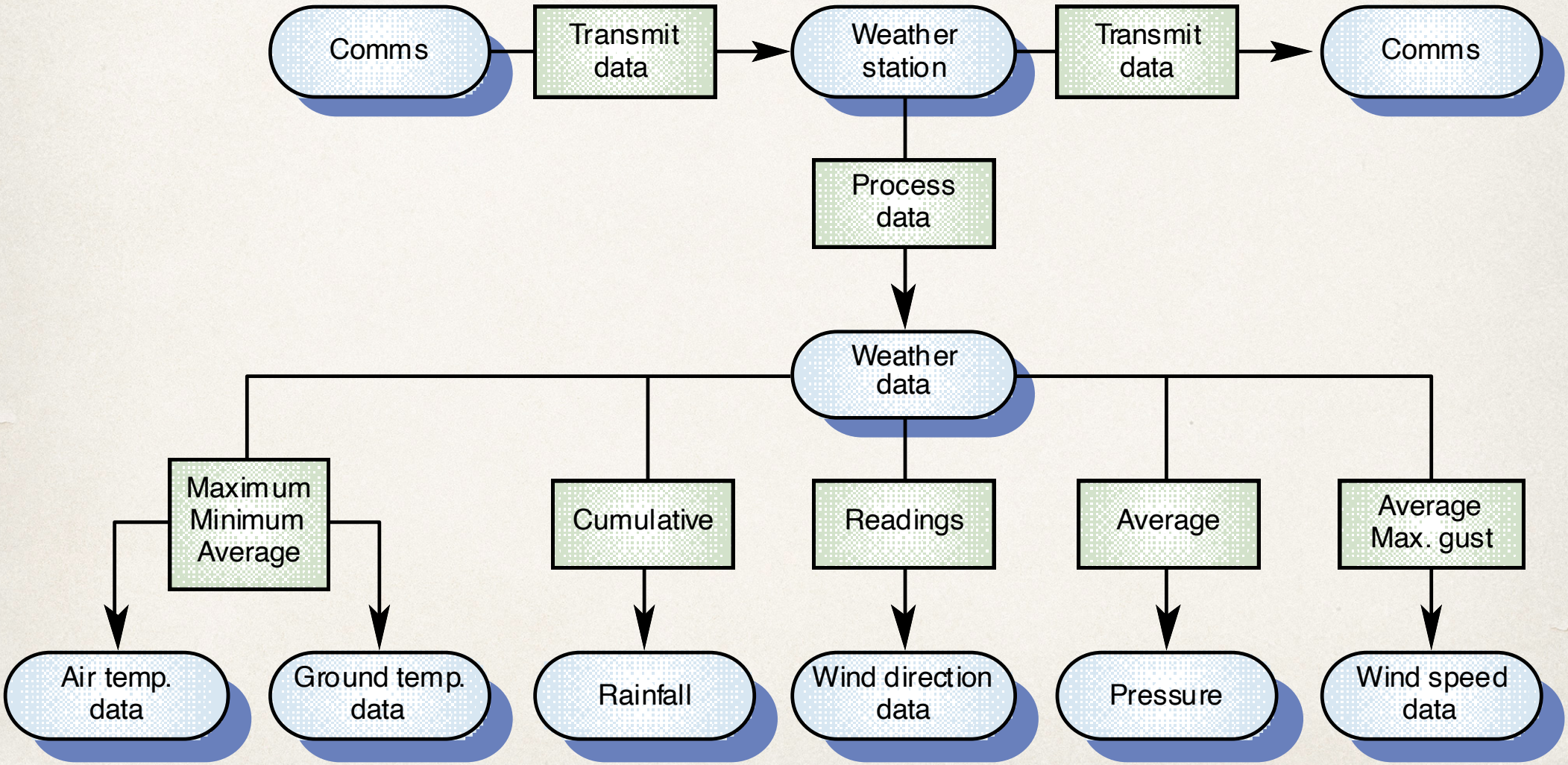
Altri oggetti della stazione meteo



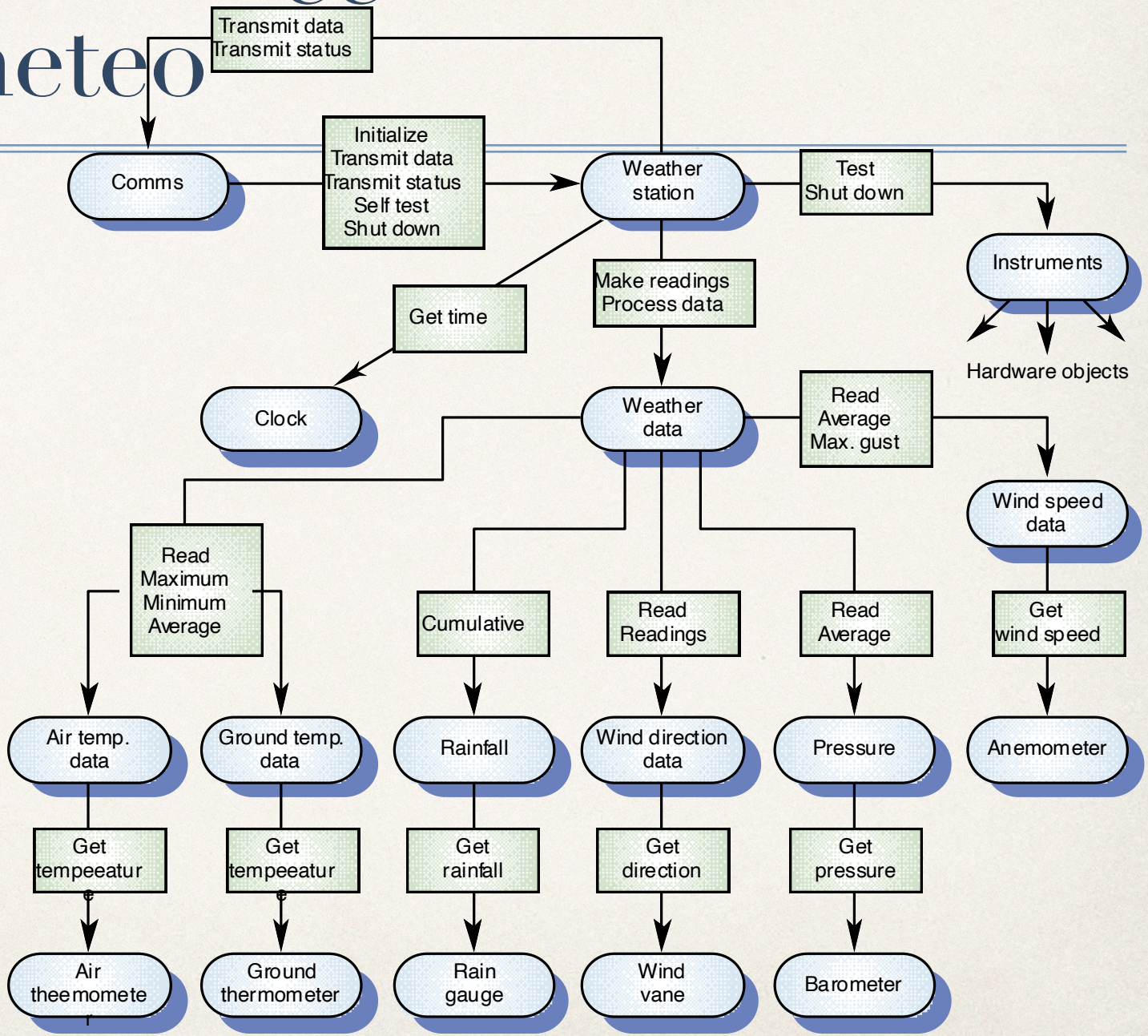
Gerarchia di aggregazione degli oggetti (struttura statica)



Interazione tra oggetti (struttura dinamica)



Interazione tra oggetti della stazione meteo



Progettazione delle interfacce

- Significa definire i tipi degli attributi e le firme dei metodi
- Bisogna evitare di dare informazioni sulla rappresentazione (interna) dei dati
- Deve essere data una specifica precisa dei metodi, usando ad esempio un PDL

Interfaccia Java

```
interface Weather_station {
private String  station_identifier ;
private Weather_data_rec Weather_data ;
private Instrument_status inst_status ;

public Weather_station () ;
public void Transmit_data (computer_id dest) ;
public void Transmit_status (computer_id dest) ;
public void Self_test () ;
public void Shut_down () ;

public    String Identifier () ;
public void Put_identifier (String  Id) ;
public instrument_status Inst_status () ;
public void Put_instr_status(Instrument_status isd) ;
public Weather_data_rec Weather_data () ;
public void Put_weather_data (Weather_data_rec wdr) ;
} ;
```

Oggetti concorrenti

- Gli oggetti possono essere visti come entità a se stanti, adatti all'implementazione concorrente
- Il modello di passaggio di messaggi tra oggetti può essere implementato direttamente se gli oggetti girano su processori diversi in un sistema distribuito