MATA KULIAH	: Rekayasa Piranti Lunak
KODE MATA KULIAH/SKS	: TI1014– 4 SKS
TAHUN	: **2013**
VERSI	: **1.0**

KALBIS Institute
Science • Technology • Business

EDUCATION FOR A BETTER LIFE

KEMAMPUAN AKHIR YANG DIHARAPKAN

Mahasiswa dapat menerapkan prinsip-prinsip Rekayasa Piranti Lunak dalam Desain proyek Rekayasa Piranti Lunak

MATERI POKOK

- Objek dan kelas objek
- Sebuah proses desain berorientasi obyek
- Desain evolusi

Sumber Pustaka

1. A. S. Rosa, Salahuddin M., Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek, Informatika, 20013.
2. Ian Sommerville, Software Engineering 9th Edition, Addison-Wesley, 2011.
3. Roger S. Pressman, Software Engineering: A Practitioner’s Approach Seventh Edition, McGraw-Hill, 2010.
4. Yasin Verdi, Rekayasa Perangkat Lunak Berorientasi Objek, Mitra Wacana Media, 2012.

Object-oriented development

- Object-oriented analysis, design and programming are related but distinct.
- OOA is concerned with developing an object model of the application domain.
- OOD is concerned with developing an object-oriented system model to implement requirements.
- OOP is concerned with realising an OOD using an OO programming language such as Java, C++ or C#.

Characteristics of OOD

- Objects are abstractions of real-world or system entities and manage themselves.
- Objects are independent and encapsulate state and representation information.
- System functionality is expressed in terms of object services.
- Shared data areas are eliminated. Objects communicate by message passing.
- Objects may be distributed and may execute sequentially or in parallel.

Object oriented modelling

- An inherent part of object-oriented development is to develop UML models to represent the system.
- Structural and behavioural UML models have already been introduced in previous lectures on system modelling.
- Diagram types used here are from UML 1 rather than UML 2 but differences are minimal.

An object-oriented design process

- Structured design processes involve developing a number of different system models.
- They require a lot of effort for development and maintenance of these models and, for small systems, this may not be cost-effective.
- However, for large systems developed by different groups design models are an essential communication mechanism.

Process stages

- The principal activities in any OO design process include:
  - Context: Define the context and modes of use of the system;
  - Architecture: Design the system architecture;
  - Objects: Identify the principal system objects;
  - Models: Develop design models;
  - Interfaces: Specify object interfaces.

Weather system description

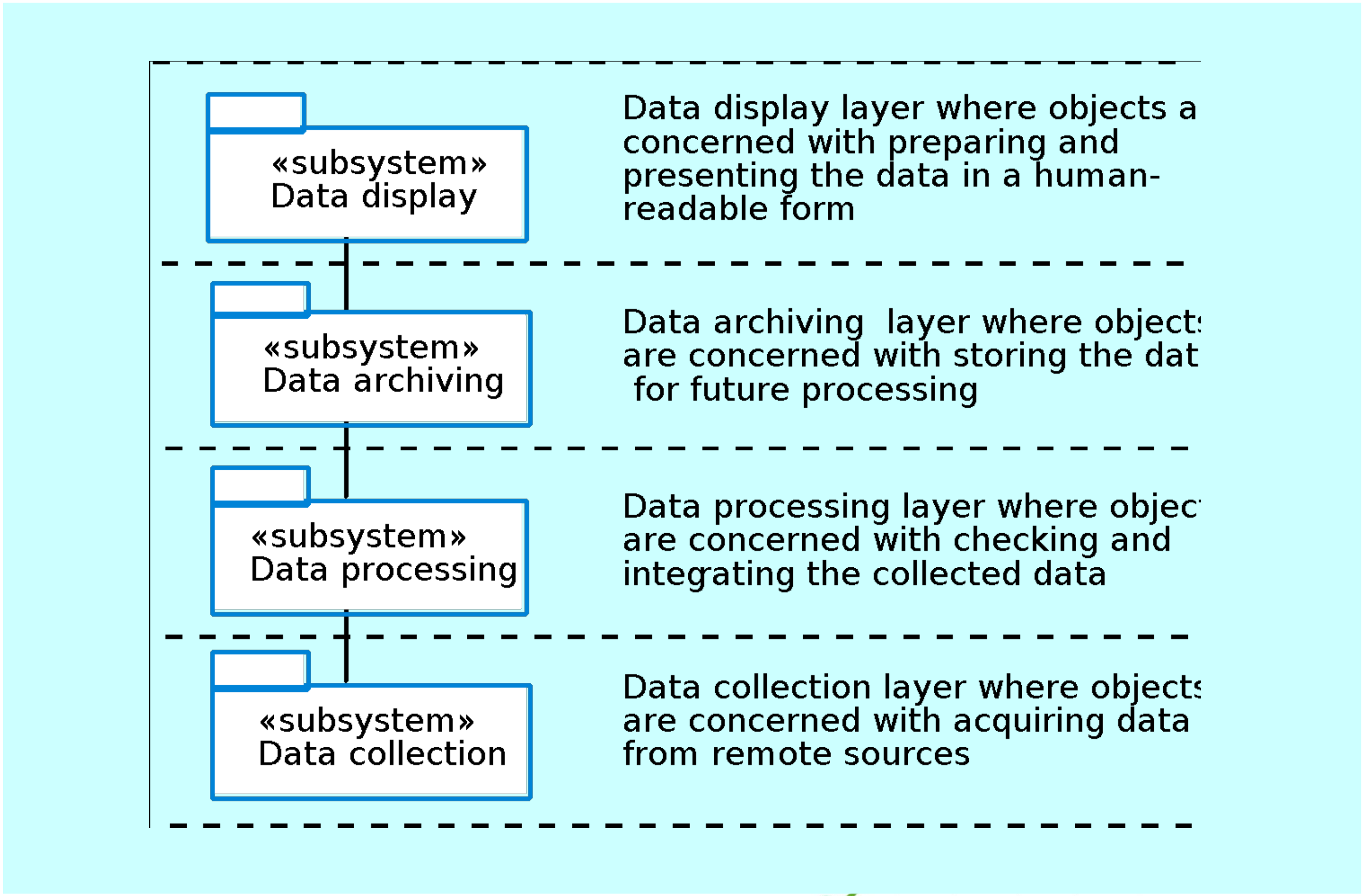
A weather mapping system is required to generate weather maps on a regular basis using data collected from remote, unattended weather stations and other data sources such as weather observers, balloons and satellites. Weather stations transmit their data to the area computer in response to a request from that machine.

The area computer system validates the collected data and integrates it with the data from different sources. The integrated data is archived and, using data from this archive and a digitised map database a set of local weather maps is created. Maps may be printed for distribution on a special-purpose map printer or may be displayed in a number of different formats.

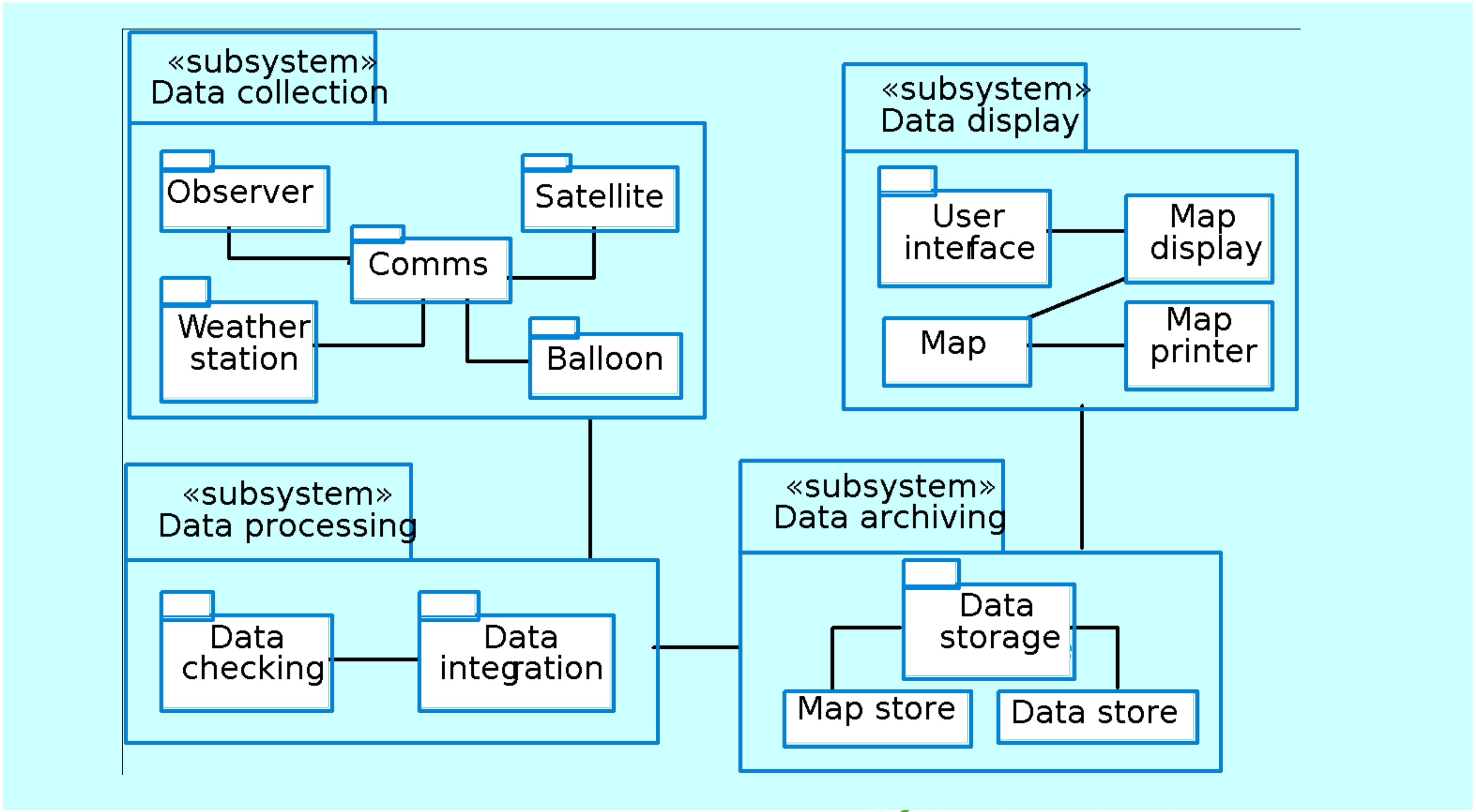
Context

- Develop an understanding of the relationships between the software being designed and its external environment
- System context
  - A static model that describes other systems in the environment. Use a subsystem model to show other systems. Following slide shows the systems around the weather station system.
- Model of system use
  - A dynamic model that describes how the system interacts with its environment. Use use-cases to show interactions

Layered architecture


«subsystem» Data display
Data display layer where objects a concerned with preparing and presenting the data in a human-readable form
«subsystem» Data archiving
Data archiving layer where object are concerned with storing the dat for future processing
«subsystem» Data processing
Data processing layer where objec are concerned with checking and integrating the collected data
«subsystem» Data collection
Data collection layer where object are concerned with acquiring data from remote sources

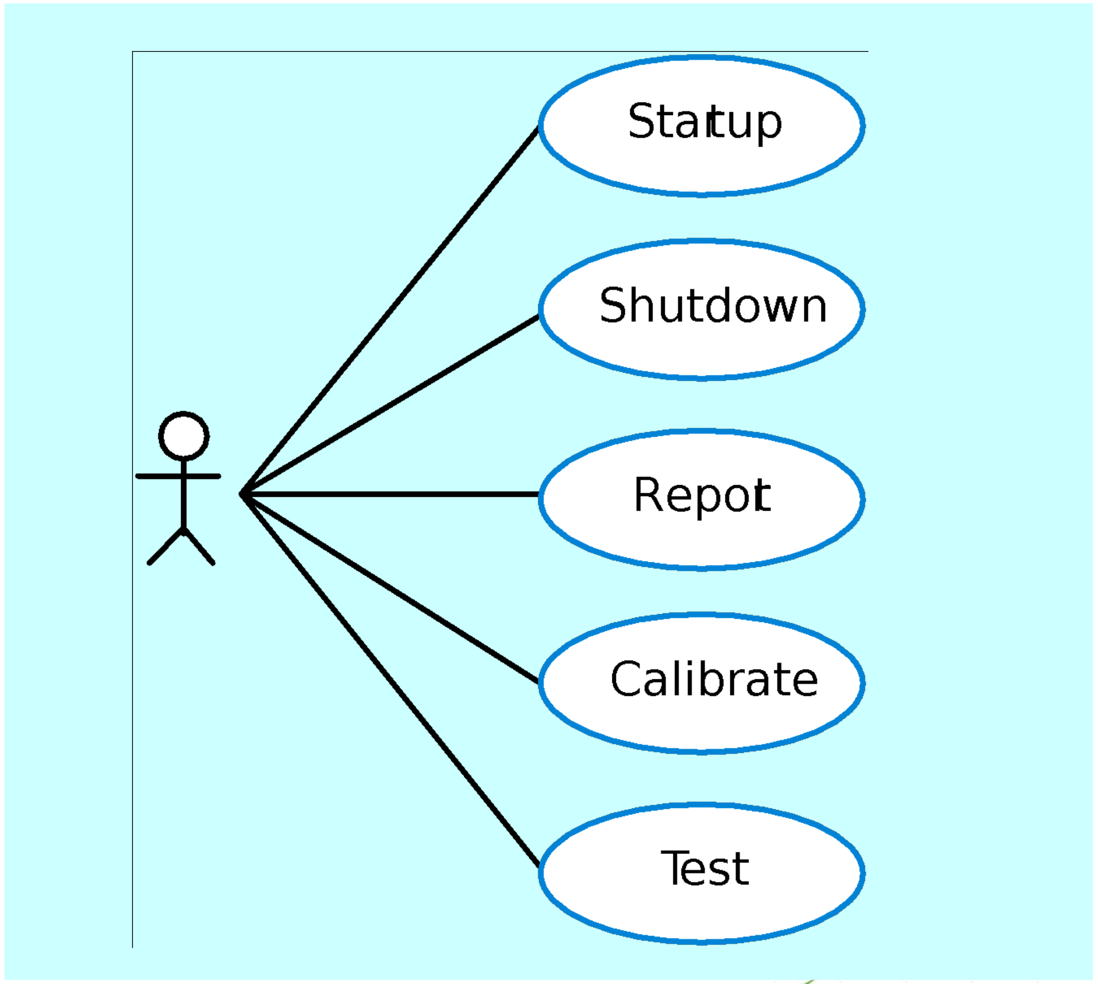
Subsystems in the weather mapping system



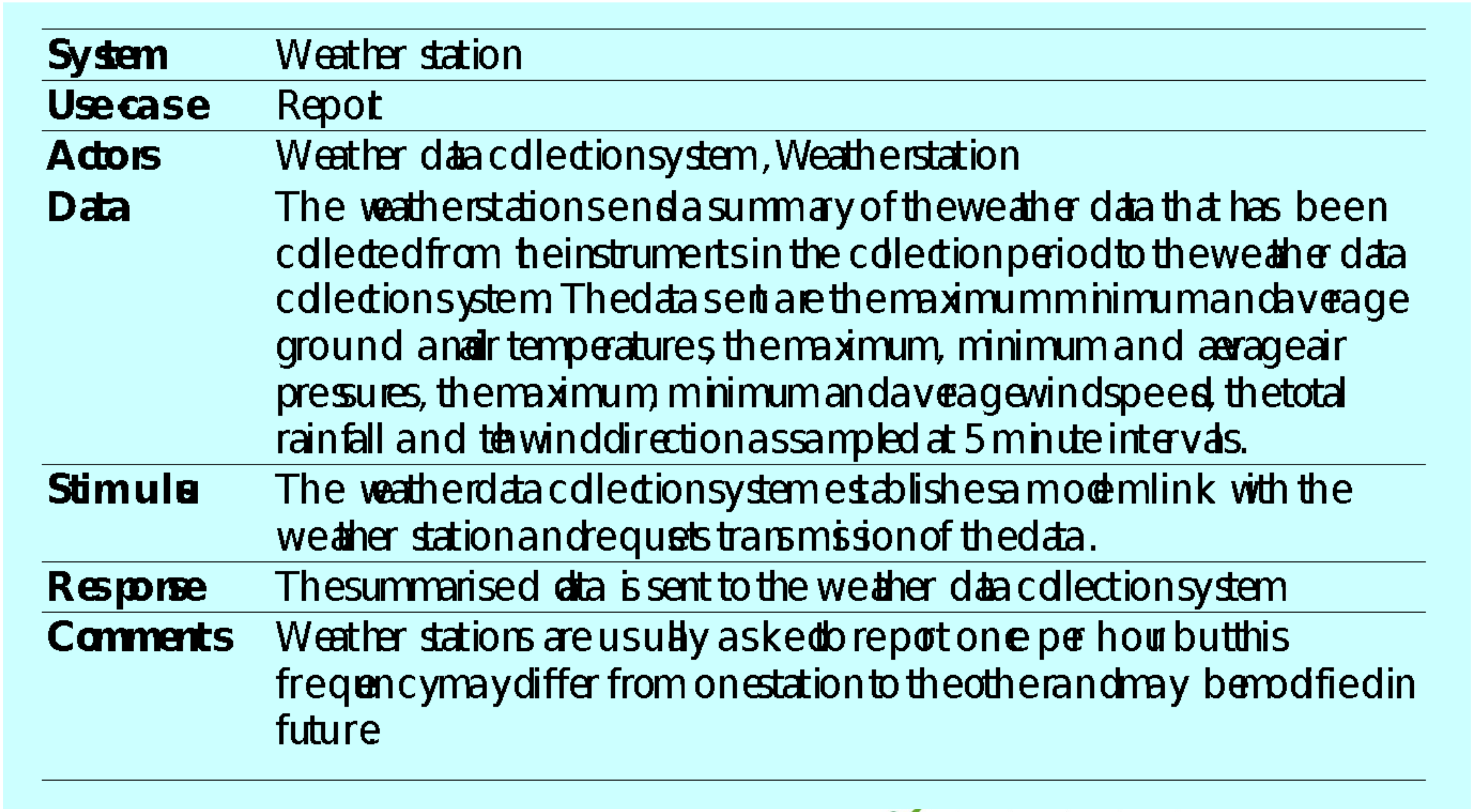
Use-case models

- Use-case models are used to represent each interaction with the system.
- A use-case model shows the system features as ellipses and the interacting entity as a stick figure.

Use-cases for the weather station



Use-case description

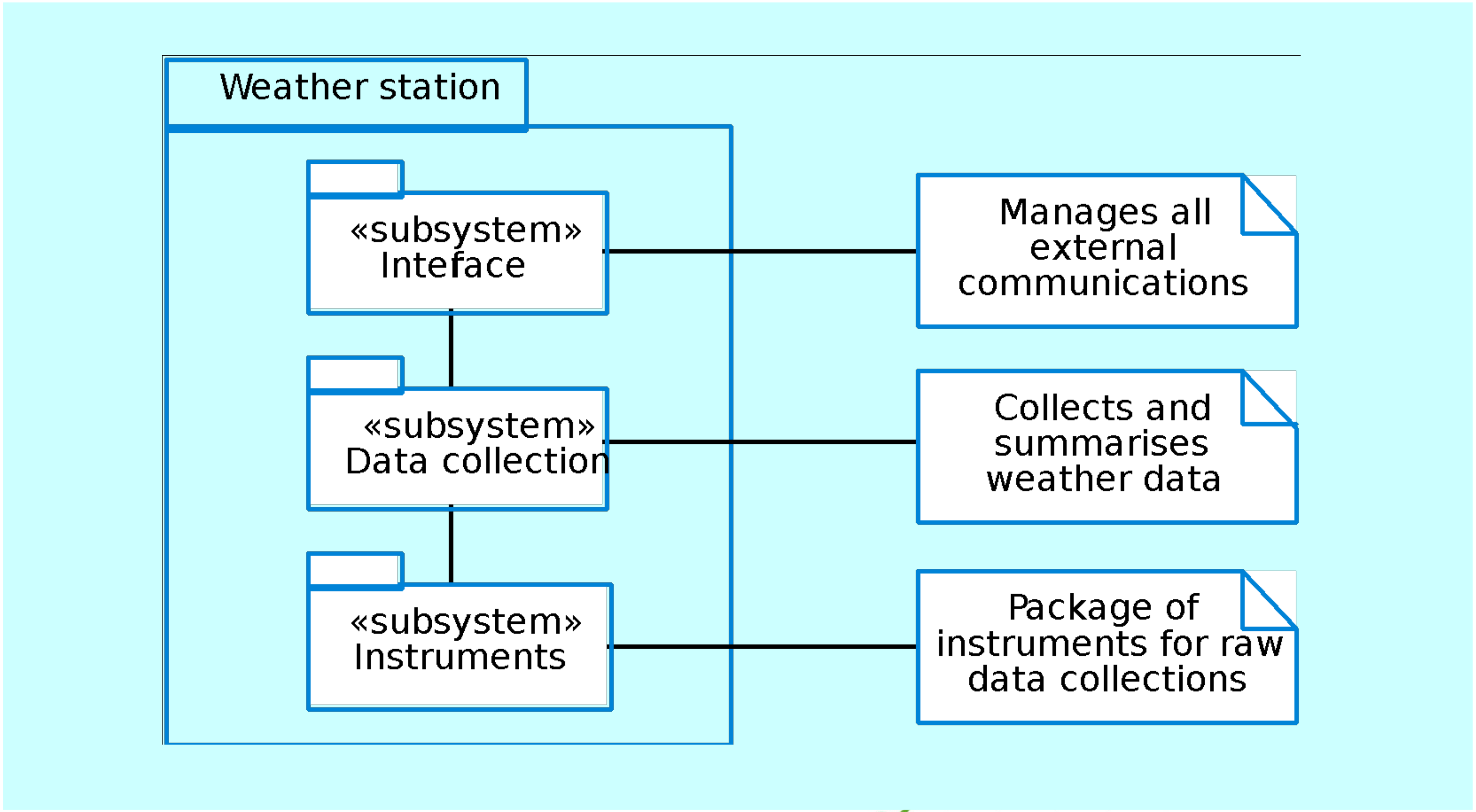


System	Weather station
Use-case	Repot
Actors	Weather data collection system, Weather station
Data	The weather station sends a summary of the weather data that has been collected from the instruments in the collection period to the weather data collection system. The data sent are the maximum minimum and average ground and air temperatures, the maximum, minimum and average air pressures, the maximum, minimum and average wind speeds, the total rainfall and the wind direction as sampled at 5 minute intervals.
Stimulus	The weather data collection system establishes a modem link with the weather station and requests transmission of the data.
Response	The summarised data is sent to the weather data collection system
Comments	Weather stations are usully asked to report once per hour but this frequency may differ from one station to the other and may be modified in future

Architecture

- Once interactions between the system and its environment have been understood, you use this information for designing the system architecture.
- A layered architecture as discussed in Chapter 11 is appropriate for the weather station
  - Interface layer for handling communications;
  - Data collection layer for managing instruments;
  - Instruments layer for collecting data.
- There should normally be no more than 7 entities in an architectural model.

Weather station architecture



Weather station
«subsystem» Inteface
«subsystem» Data collection
«subsystem» Instruments
Manages all external communications
Collects and summarises weather data
Package of instruments for raw data collections

Objects

- Identifying objects (or object classes) is the most difficult part of object oriented design.
- There is no 'magic formula' for object identification. It relies on the skill, experience and domain knowledge of system designers.
- Object identification is an iterative process. You are unlikely to get it right first time.

Approaches to identification

- Use a grammatical approach based on a natural language description of the system (used in Hood OOD method).
- Base the identification on tangible things in the application domain.
- Use a behavioural approach and identify objects based on what participates in what behaviour.
- Use a scenario-based analysis. The objects, attributes and methods in each scenario are identified.

Weather station description

A weather station is a package of software controlled instruments which collects data, performs some data processing and transmits this data for further processing. The instruments include air and ground thermometers, an anemometer, a wind vane, a barometer and a rain gauge. Data is collected periodically.

When a command is issued to transmit the weather data, the weather station processes and summarises the collected data. The summarised data is transmitted to the mapping computer when a request is received.

Weather station object classes

- Ground thermometer, Anemometer, Barometer
  - Application domain objects that are ‘hardware’ objects related to the instruments in the system.
- Weather station
  - The basic interface of the weather station to its environment. It therefore reflects the interactions identified in the use-case model.
- Weather data
  - Encapsulates the summarised data from the instruments.

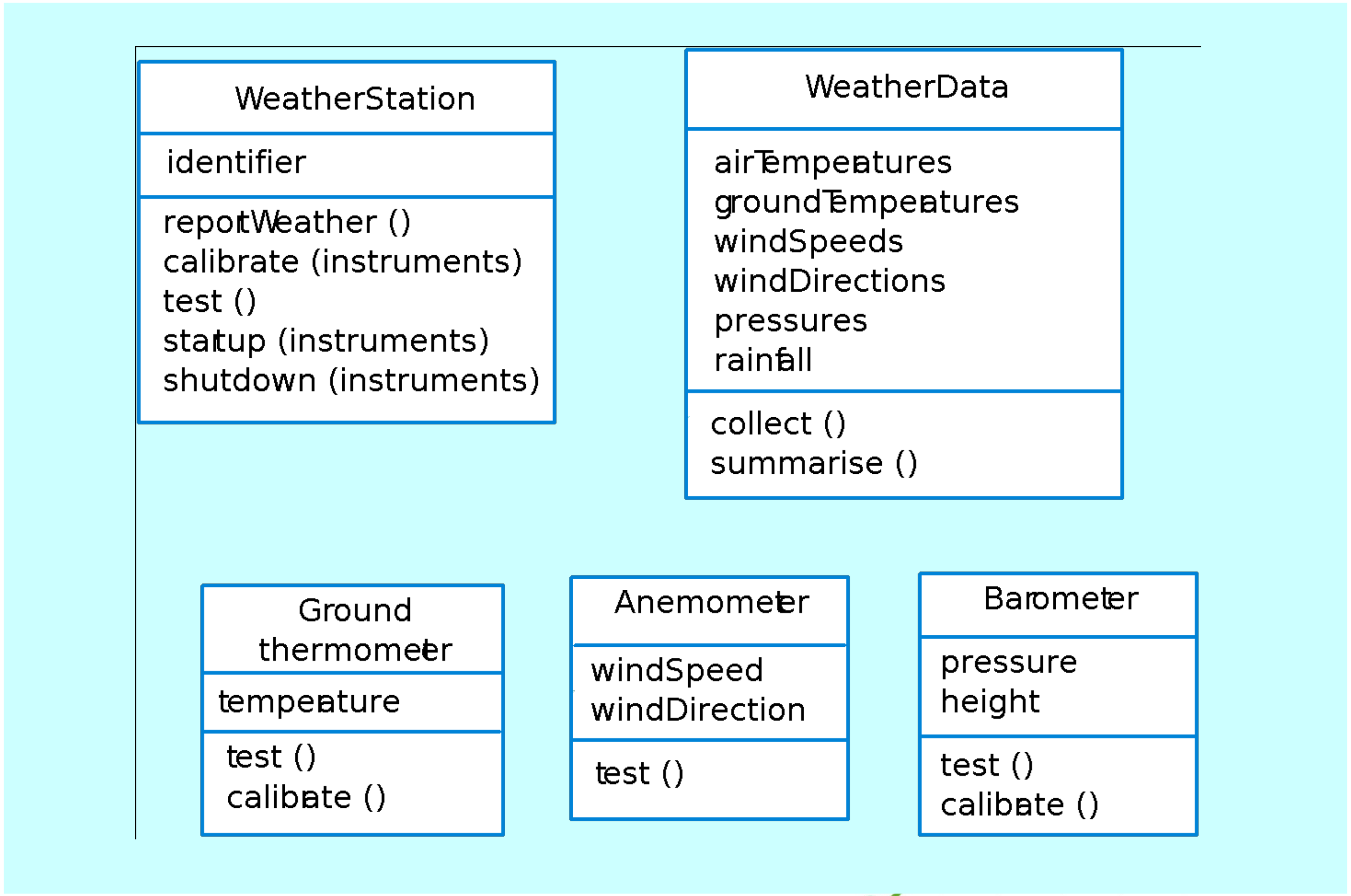
Key points

- Object-oriented development involves adopting an OO approach at all stages from specification through to programming
- OO design involves designing the system using objects as the fundamental abstraction and representing the system as an associated set of models in the UML
- The OO design process involves several stages - discussed here were Context, Architecture and Objects.

Recap

- The stages in the OO design process are:
  - Context: Define the context and modes of use of the system;
  - Architecture: Design the system architecture;
  - Objects: Identify the principal system objects;
  - Models: Develop design models;
  - Interfaces: Specify object interfaces.

Models


WeatherStation
identifier
reportWeather ()
calibrate (instruments)
test ()
startup (instruments)
shutdown (instruments)
WeatherData
airTemperatures
groundTemperatures
windSpeeds
windDirections
pressures
rainfall
collect ()
summarise ()
Ground thermometer
temperature
test ()
calibrate ()
Anemometer
windSpeed
windDirection
test ()
Barometer
pressure
height
test ()
calibrate ()

Further objects and object refinement

- Use domain knowledge to identify more objects and operations
  - Weather stations should have a unique identifier;
  - Weather stations are remotely situated so instrument failures have to be reported automatically. Therefore attributes and operations for self-checking are required.
- Active or passive objects
  - In this case, objects are passive and collect data on request rather than autonomously. This introduces flexibility at the expense of controller processing time.

Design models

- Design models show the objects and object classes and relationships between these entities.
- Static models describe the static structure of the system in terms of object classes and relationships.
- Dynamic models describe the dynamic interactions between objects.

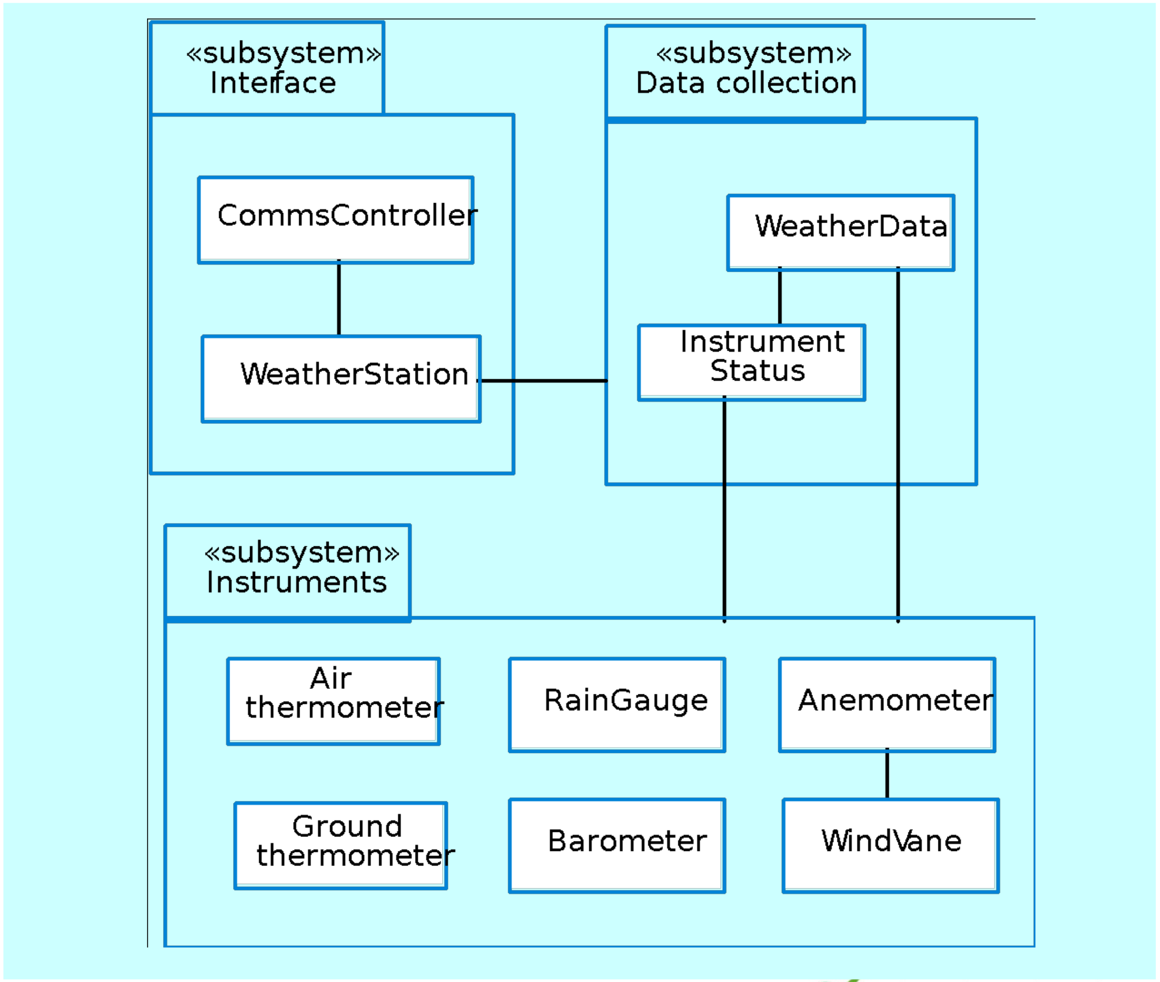
Examples of design models

- Sub-system models that show logical groupings of objects into coherent subsystems.
- Sequence models that show the sequence of object interactions.
- State machine models that show how individual objects change their state in response to events.
- Other models include use-case models, aggregation models, generalisation models, etc.

Subsystem models

- Shows how the design is organised into logically related groups of objects.
- In the UML, these are shown using packages - an encapsulation construct.
- The UML stereotype annotation is used to label packages as sub-systems.

Weather station subsystems



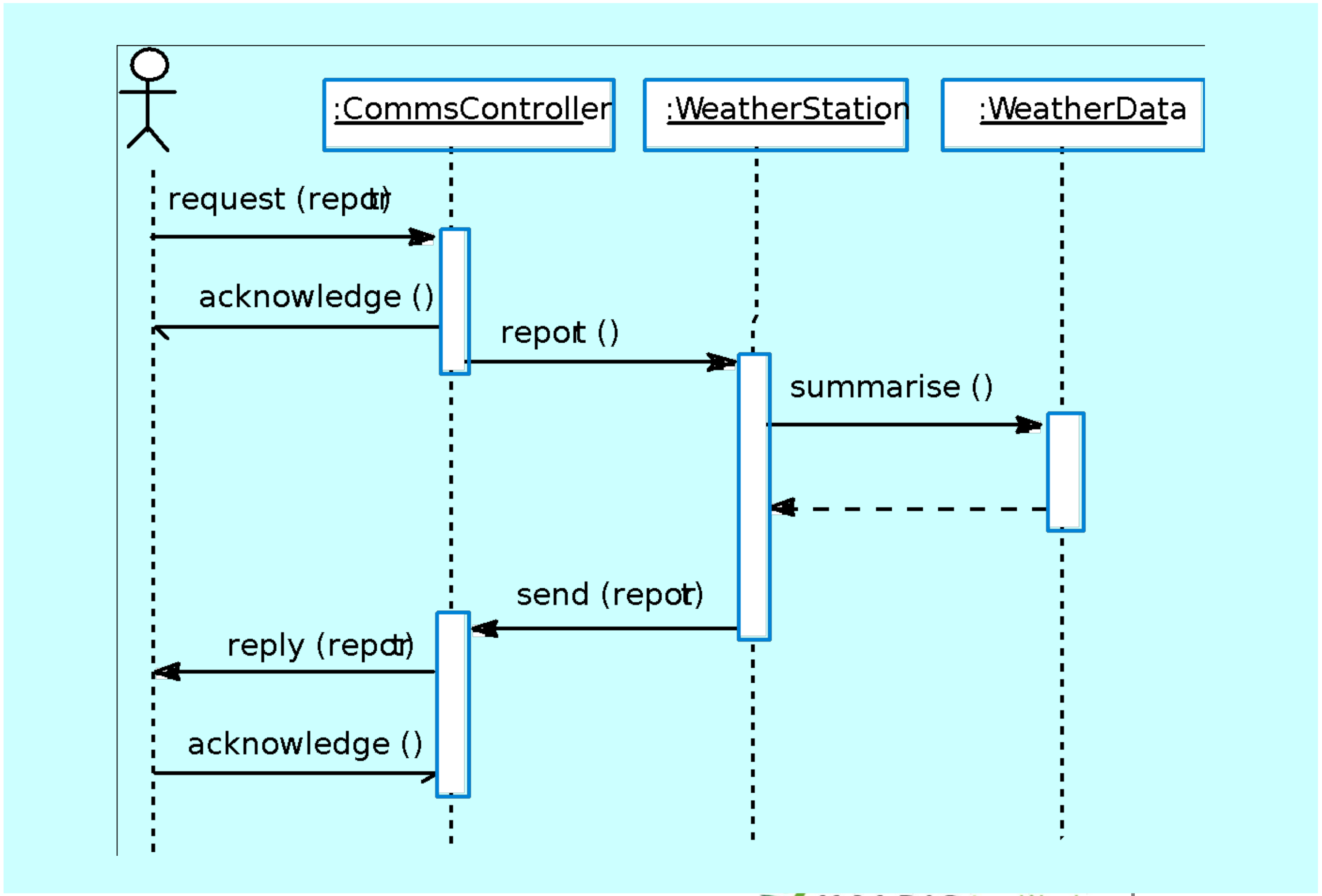
Subsystem decomposition

- Interface subsystem
  - Includes the objects in the system that are concerned with interfacing the weather station to external systems
  - May include other objects from those shown here - e.g. a user interface for testing.
- Data collection subsystem
  - Includes objects that implement the strategies adoped for data collection
  - These are deliberately separated from the actual data collection to allow for changes to these strategies
- Instruments subsystem
  - Includes all objects that interface to the instrument hardware

Sequence models

- Sequence models show the sequence of object interactions that take place
  - Objects are arranged horizontally across the top;
  - Time is represented vertically so models are read top to bottom;
  - Interactions are represented by labelled arrows, Different styles of arrow represent different types of interaction;
  - A thin rectangle in an object lifeline represents the time when the object is the controlling object in the system.

Data collection sequence



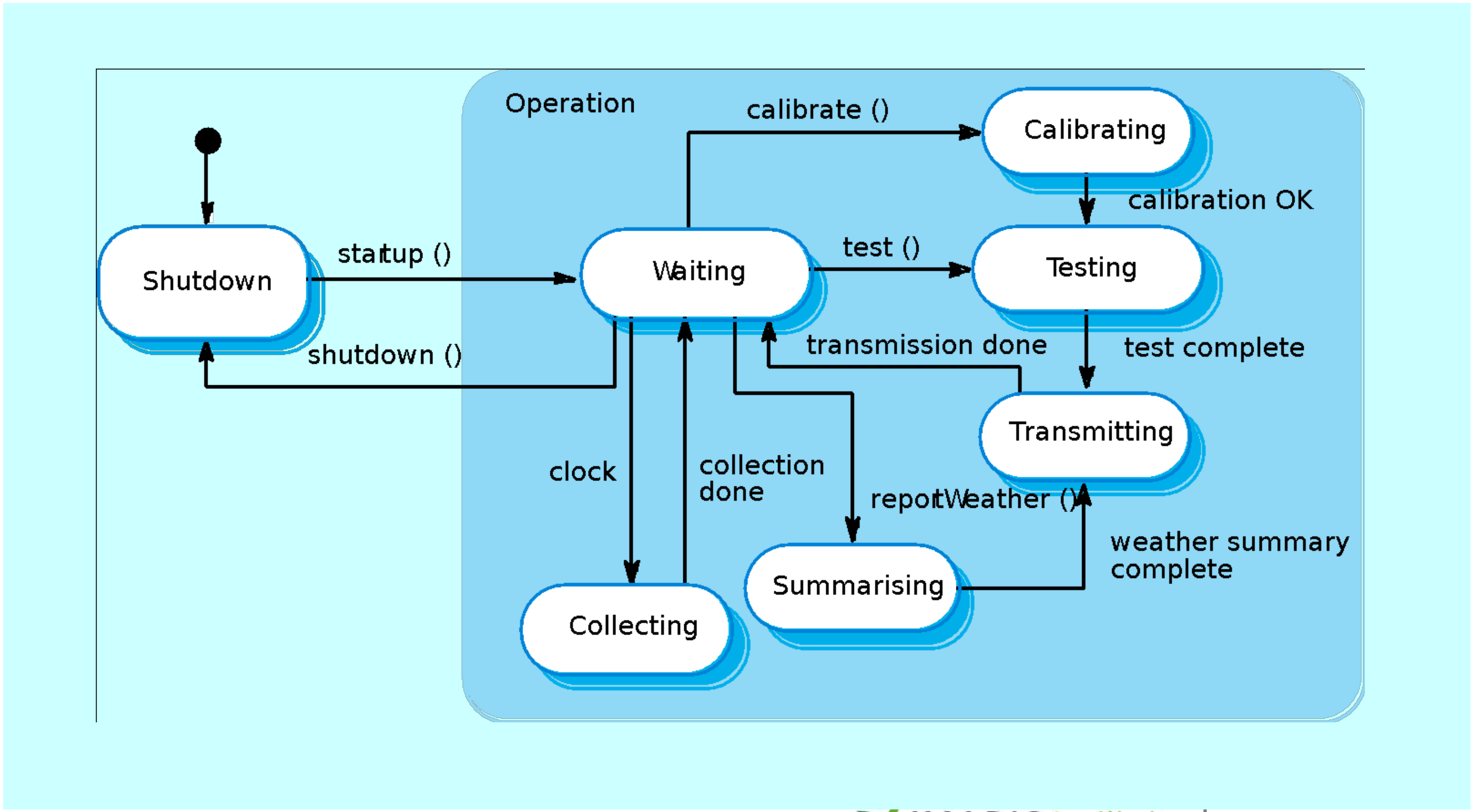
Sequence of operations

- The sequence diagram shows:
  - An external entity, shown as a stick man, (it can be a person or another system) initiates the data collection by sending a request to the interface object
  - The interface object sends a request to the weather station to provide a report for transmission
  - The weather station requests the WeatherData object which maintains all raw weather data to provide a summary that will be included in this report

Statecharts

- Show how objects respond to different service requests and the state transitions triggered by these requests
- The states are represented as rounded rectangles
- State transitions are labelled links between these rectangles

Weather station state diagram



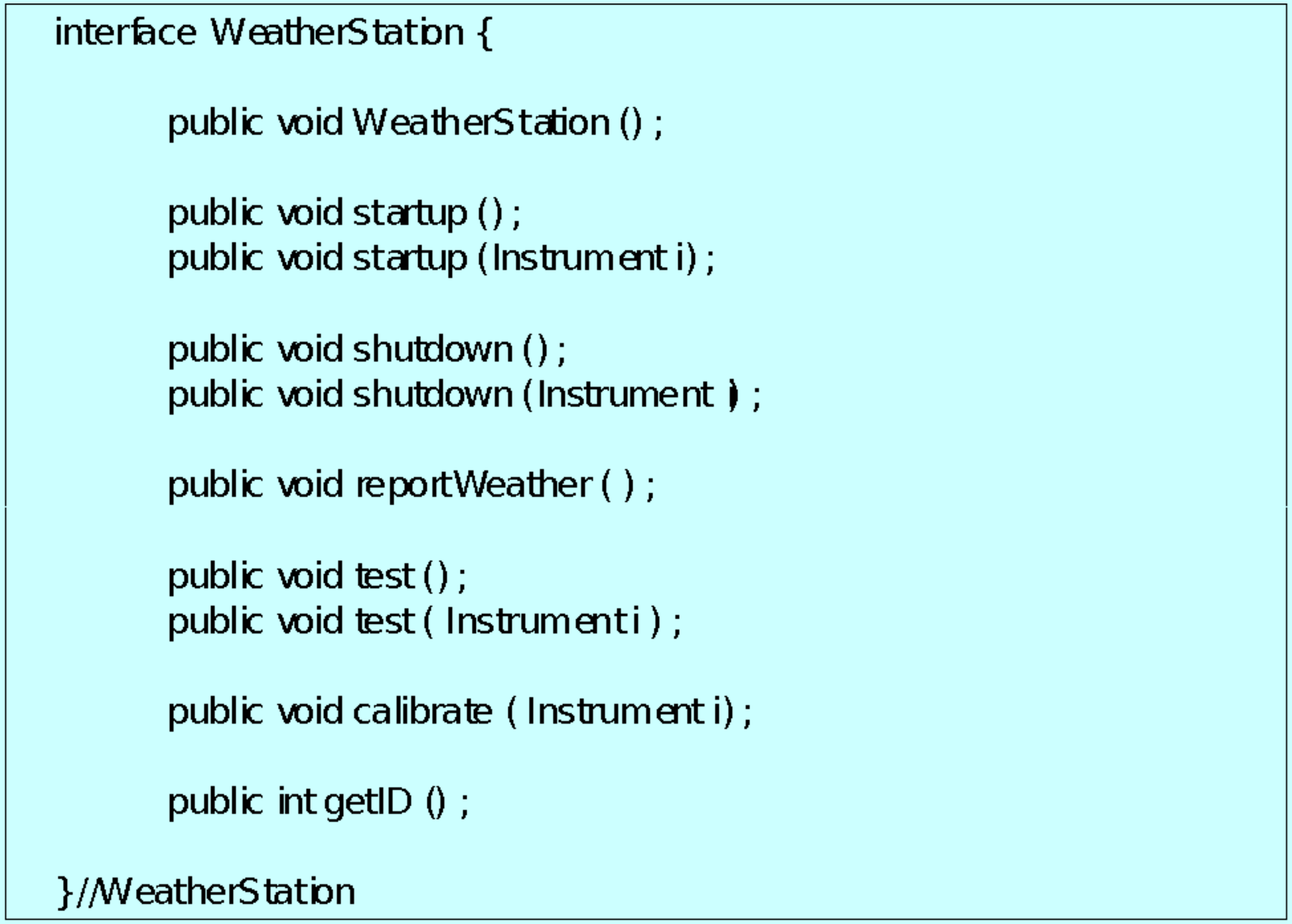
Weather station system states

- If object state is Shutdown then it responds to a Startup() message;
- In the waiting state the object is waiting for further messages;
- If reportWeather () then system moves to summarising state;
- If calibrate () the system moves to a calibrating state;
- A collecting state is entered when a clock signal is received.

Interfaces

- Object interfaces have to be specified so that the objects and other components can be designed in parallel.
- Designers should avoid designing the interface representation but should hide this in the object itself.
- Objects may have several interfaces which are viewpoints on the methods provided.
- The UML uses class diagrams for interface specification but Java may also be used.

Weather station interface



```
interface WeatherStation {

        public void WeatherStation () ;

        public void startup () ;
        public void startup (Instrument i) ;

        public void shutdown () ;
        public void shutdown (Instrument i) ;

        public void reportWeather ( ) ;

        public void test () ;
        public void test ( Instrument i ) ;

        public void calibrate ( Instrument i) ;

        public int getID () ;

}//WeatherStation
```

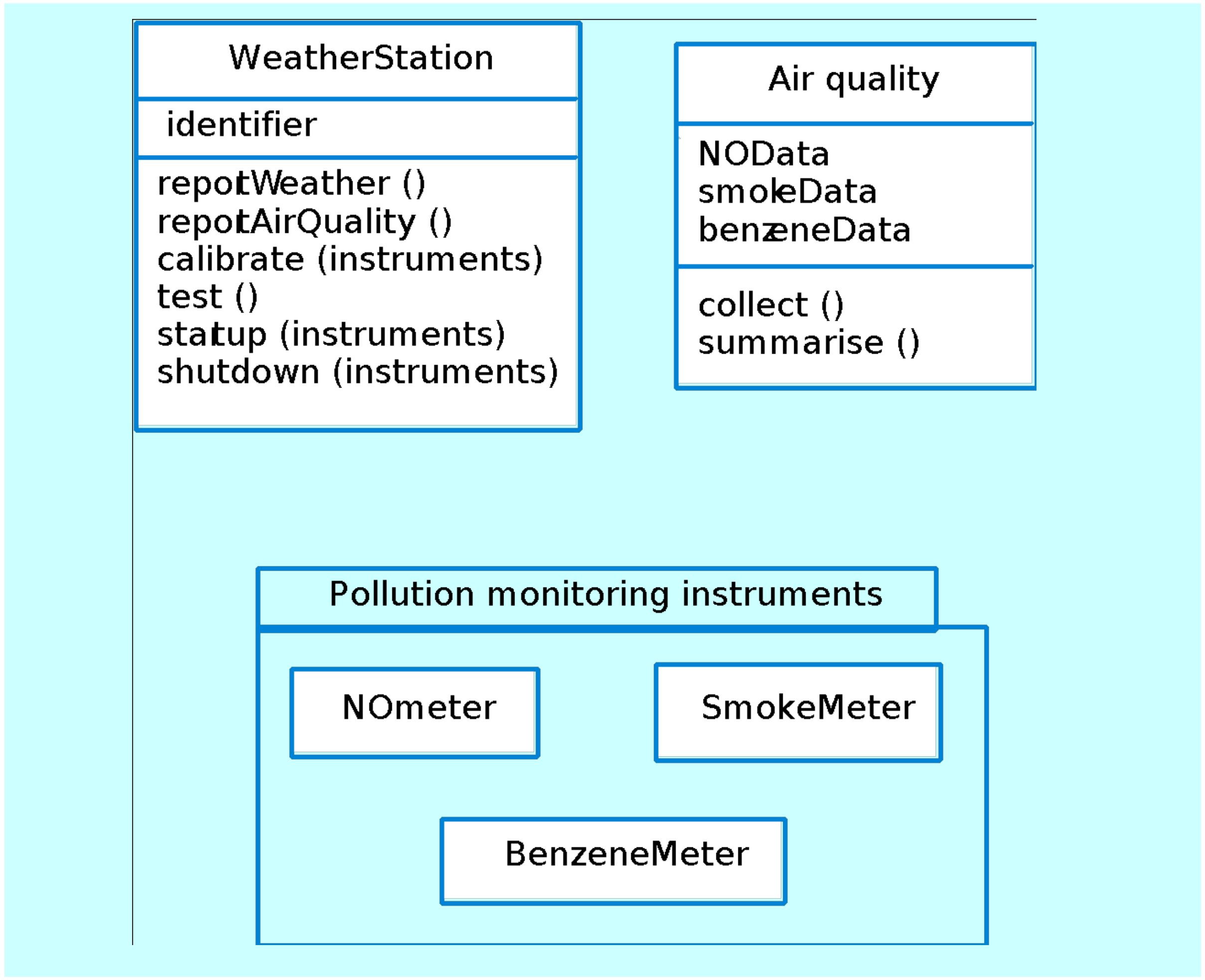
Design evolution

- Hiding information inside objects means that changes made to an object do not affect other objects in an unpredictable way.
- Assume pollution monitoring facilities are to be added to weather stations. These sample the air and compute the amount of different pollutants in the atmosphere.
- Pollution readings are transmitted with weather data.

Changes required

- Add an object class called Air quality as part of WeatherStation.
- Add an operation reportAirQuality to WeatherStation. Modify the control software to collect pollution readings.
- Add objects representing pollution monitoring instruments.

Pollution monitoring



Summary

- A range of different models may be produced during an object-oriented design process. These include static and dynamic system models.
- Object interfaces should be defined precisely using e.g. a programming language like Java.
- Object-oriented design potentially simplifies system evolution.

KALBIS Institute
Science • Technology • Business
EDUCATION FOR A BETTER LIFE