
# Introduction

Mariano Ceccato

mariano.ceccato@univr.it

# Lecturer

Associate professor, PhD

Department of Computer Science, room 1.53
Email: mariano.ceccato@univr.it
Web: https://profs.scienze.univr.it/~ceccato/

2003-2019 Fondazione Bruno Kessler, Trento
2016-2019 CTO: 2aspire, startup on digital security
2020-now: University of Verona
Research interest:

- Automated software testing
- Security testing (Web apps, Android apps, blockchain smart contracts, REST Api)
- Empirical software engineering

# Course

- Fondamenti di ingegneria del software
- Progettazione e validazione di sistemi software

# Course schedule

- Thursday 16:30-18:30 – Room T.03
- Friday 13:30-16:30 – Room I

# Course organization

**Theory:**

- Software process
- Agile methodologies
- Requirement engineering
- Software design (UML)
- Software architectures
- Software testing
- Code refactoring
- Project management

**Lab + Homework:**

- Exercises on agile
- Exercises on architectures
- Exercises on testing

**Course Project:**

- Real world case study program
- Perspective of software engineer
- Maintenance project to be delivered to the software owner

# Course exam

- Written exam (on the theory): 22 points
- Project: 6 points
  - To be delivered 7 days before the date when you take the exam
  - In case you fail the exam, the project keeps valid
- Homework: 4 points
  - To be delivered by the deadline set during the lab

# Notice

- The topic is potentially huge:
  - High level overview on most of the topics
  - Detailed focus on selected topics

# Software

- The economies of ALL developed nations are dependent on software
  - National & international institutions, companies, infrastructures
  - Entertainment, communication
- Software is intangible
  - Can quickly become very complex and difficult to maintain
- Software engineering is concerned with theories, methods and tools for professional software development.

# Software cost

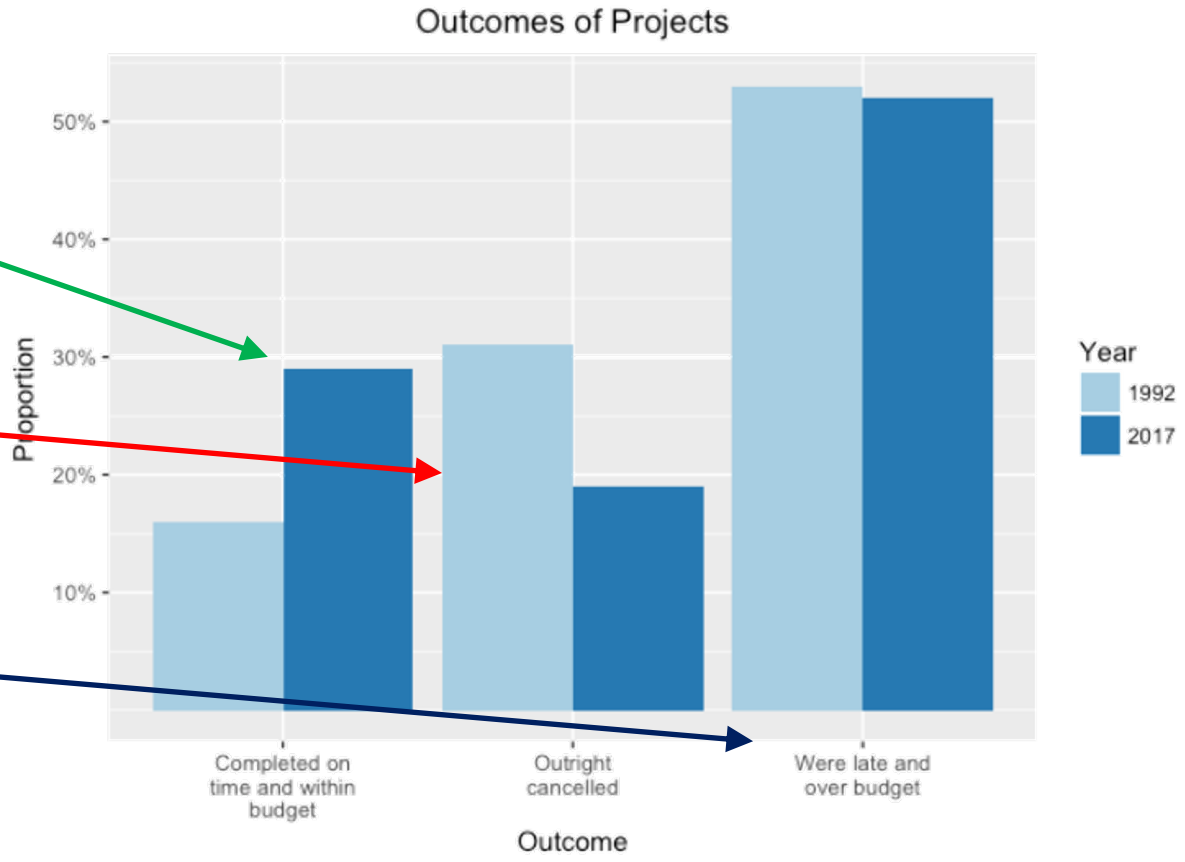
- The costs of software are often greater than the hardware cost
- Maintenance cost > development develop.
  - Systems with a long life, maintenance costs >>> development costs
- Software engineering is concerned with cost-effective software development

# Current practice

- Report of Standish Group
  - IT research and consulting company located in West Yarmouth, Massachusetts.
  - They are most famous for their research on why IT projects fail

Survey of 50,000 projects conducted in 2017 around the world revealed:

- 29 percent of IT projects were successful, coming in on time and on budget
- 19 percent failed completely, canceled prior to completion or delivered but never used
- 52 percent were "challenged," arriving over budget, late (over time), or with less-than-required features



# Common problems

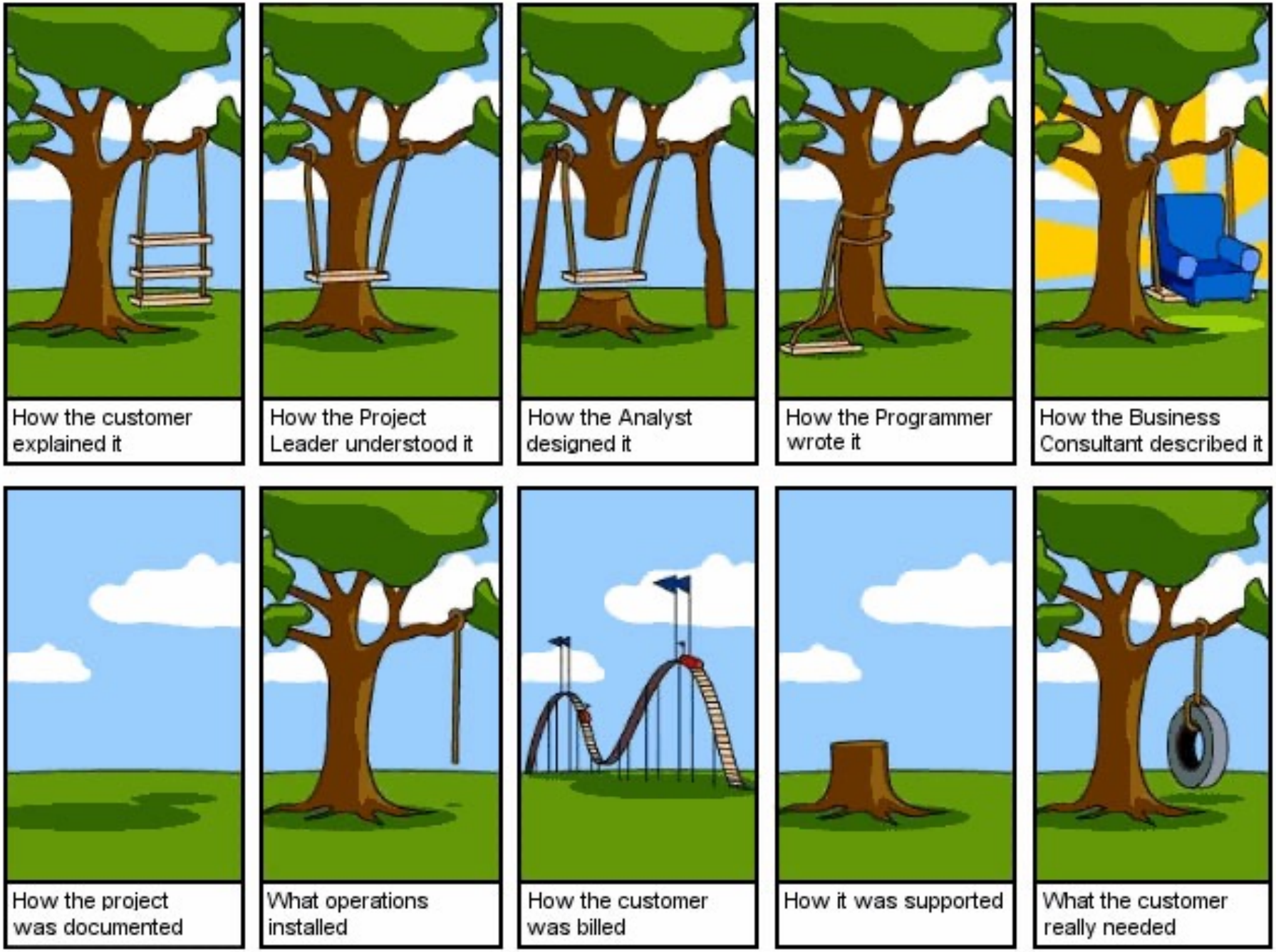
- Costs larger than forecasted
  - Over budget
- Delivering after the contract deadline
  - Over time
- Final system with partial features
- Software with low quality
  - Usability concerns, defects
- Cancelled projects
- End-users are not fully satisfied
  - The system does not satisfy user needs

# Alleged reasons

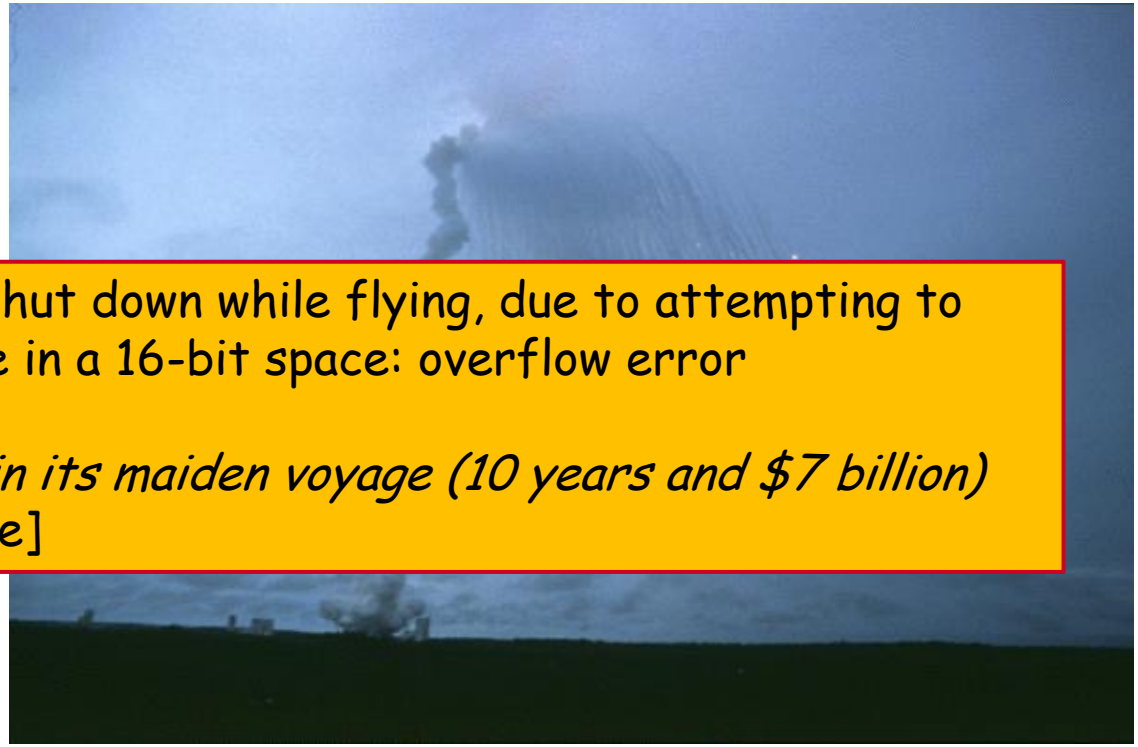
- Intrinsic complexity of the problem
- Technological problems
  - Decision on programming language, infrastructure, framework,

# Real reasons

- Communication problems between software engineers and stakeholders
- Difficulties in understanding the end-user problems
- Difficulties in understanding the business domain
  - Context in which the software operates
    - E.g., the concept of *purchase order* and *invoice* and their relation in an invoicing system
- Management and organizational problems
  - Planning, allocation of resources, people management
- Difficulties in team working


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

# 1996 - Arianne 5



Propulsion system shut down while flying, due to attempting to write a 64-bit value in a 16-bit space: overflow error

*Ariane 5 explodes in its maiden voyage (10 years and $7 billion)* [NY Times Magazine]

**37 after take-off**

# The cause

```
L_M_BV_32 := TBD.T_ENTIER_32S ((1.0/C_M_LSB_BV) * G_M_INFO_DERIVE(T_ALG.E_BV));
if L_M_BV_32 > 32767 then
     P_M_DERIVE(T_ALG.E_BV) := 16#7FFF#;
elsif L_M_BV_32 < -32768 then
     P_M_DERIVE(T_ALG.E_BV) := 16#8000#;
else
     P_M_DERIVE(T_ALG.E_BV) := UC_16S_EN_16NS(TDB.T_ENTIER_16S(L_M_BV_32));
end if;

P_M_DERIVE(T_ALG.E_BH) := UC_16S_EN_16NS (TDB.T_ENTIER_16S ((1.0/C_M_LSB_BH) *
G_M_INFO_DERIVE(T_ALG.E_BH)));
```

- For efficiency reasons, an “if” statement to check the value size was removed (it is present in preceding lines)
- Even if the direct cause is a programming mistake, the real cause of the disaster can be recognized as a poor management of the software project

# Nato conference 1968

# Software engineering

- Engineering discipline concerned with all aspects of software production
- Engineering discipline
  - Using appropriate theories and methods to solve problems bearing in mind organizational and financial constraints.
- All aspects of software production
  - From the early stages of system specification through to maintaining the system after it has gone into use
  - Project management and the development of tools, methods etc. to support software production.

Engineering is about getting results of the required quality within schedule and budget. This often involves making compromises: engineers cannot be perfectionists. People writing programs for themselves, however, can spend as much time as they wish on the program development.

# Software engineering

- We need to be able to produce reliable and trustworthy systems economically and quickly
  - Respect budget constraint
  - Meet time deadline
  - Deliver expected feature
  - High quality of software

# Topics in the course

- Software process
- Agile methodologies
- Requirement engineering
- Software design
- Software architectures
- Software testing
- Code refactoring
- Project management

# Case studies

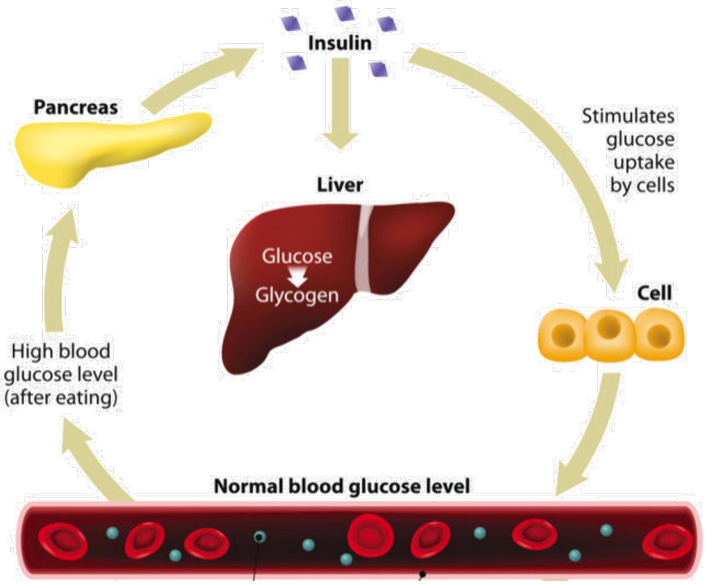
# 1. Integrated system

- **Problem:**
- Diabetes is a common disease related to pancreas malfunction
  - Commonly treated with self tests of blood sugars + insulin injection few times per day
  - Suboptimal:  irregular measurements could cause too low/high blood sugars

# 1. Integrated system

**System: personal insulin pump**

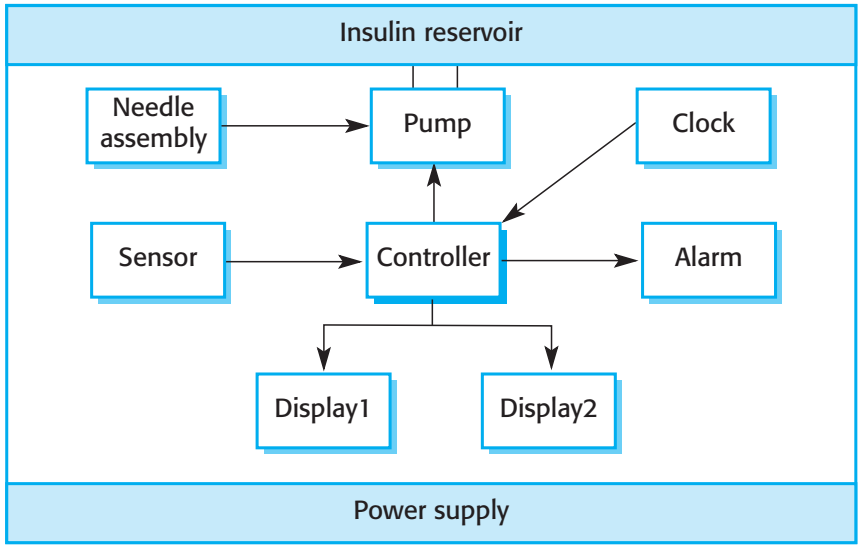
- An embedded system in an insulin pump used by diabetics to maintain blood glucose control
- Collects data from a blood sugar sensor and calculates the amount of insulin required to be injected
  - Calculation based on the rate of change of blood sugar levels.
  - Sends signals to a micro-pump to deliver the correct dose of insulin.
- Safety-critical system as low/high blood sugars must be avoided.



# Personal insulin pump

**System:**

- An embedded system in an insulin pump used by diabetics to maintain blood glucose control
- Collects data from a blood sugar sensor and calculates the amount of insulin required to be injected
  - Calculation based on the rate of change of blood sugar levels.
  - Sends signals to a micro-pump to deliver the correct dose of insulin.
- Safety-critical system as low/high blood sugars must be avoided.




**Critical requirements:**

1. The system shall be available to deliver insulin when required
2. The system shall perform reliably and deliver the correct amount of insulin to counteract the current level of blood sugar.

# 2. Information system

- Mentcare: A patient information system for mental health care
  - Information about patients
  - Treatments they received
- Patients do not require dedicated hospital treatment but need to attend specialist clinics regularly and meet a doctor who has detailed knowledge of their problems
- To facilitate patients to attend, clinics are not only in hospitals, but also in local medical practices or community centers
  - Central database
  - Access by remote hosts (laptops and smartphones) that can download local copy of patient data

# Users

- Patients might be irrational
  - Lose prescriptions or drugs (accidentally or intentionally)
  - Get lost in clinics (suddenly)
  - Forget about scheduled meetings with doctors

- Users:
  - Medical staff: doctors, nurses, health visitors (nurses who visit patients at their homes)
  - Nonmedical staff:
    - Receptionists, who make appointments
    - Medical records staff, who maintain the records system,
    - Administrative staff, who generate reports

# Key features

- Individual care management
  - Clinicians can create records for patients, edit the information in the system, view patient history, etc.
  - The system supports data summaries so that doctors can quickly learn about the key problems and treatments that have been prescribed.

- Patient monitoring
  - The system monitors the records of patients that are involved in treatment and issues warnings if possible problems are detected.

- Administrative reporting
  - The system generates monthly management reports showing the number of patients treated at each clinic, the number of patients who have entered and left the care system, number of patients sectioned, the drugs prescribed and their costs, etc.

# Low-compliant concerns

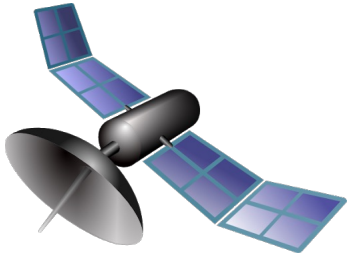
- Privacy:
  - Patient information is confidential and is never disclosed to anyone apart from authorized medical staff and the patient themselves.

- Safety:
  - Some mental illnesses may be dangerous (for patients or for others). Wherever possible, the system should warn medical staff about potentially dangerous cases.
  - Decisions should ne logged for a possible police investigation or judicial review, if necessary

# 3. sensor-based data collection system

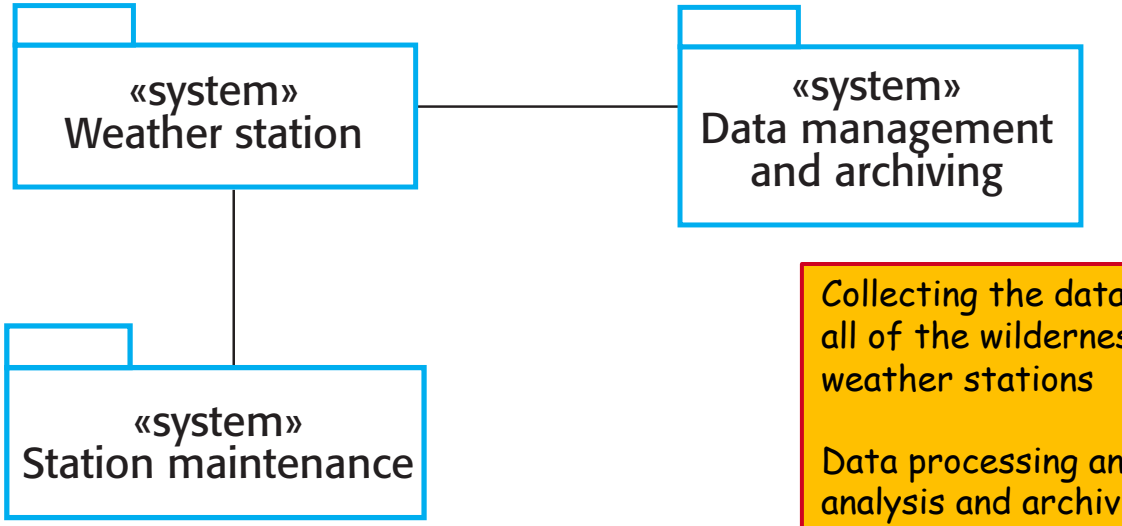
**System: wilderness weather station**

- The government of a country with large areas of wilderness decides to deploy several hundred weather stations in remote areas
- Collect data from a set of instruments that measure temperature and pressure, sunshine, rainfall, wind speed and wind direction



# Infrastructure

Collecting weather data, carrying out some initial data processing (low bandwidth of satellite link) and transmitting it to the data management system




Collecting the data from all of the wilderness weather stations

Data processing and analysis and archiving data

Communicating via satellite with all wilderness weather stations to monitor the health of these systems and provide reports of problems

# Additional features

- Monitor the instruments of power and communication hardware and report faults (due to animals or environment) to the management system
- Manage the system power
  - Ensuring that batteries are charged whenever possible (e.g., sun, wind)
  - Generators are shut down in potentially damaging weather conditions (e.g., high wind)
- Support dynamic reconfiguration
  - Parts of the software are replaced with new versions
  - Backup instruments are switched into the system in the event of system failure

# 4. A support environment

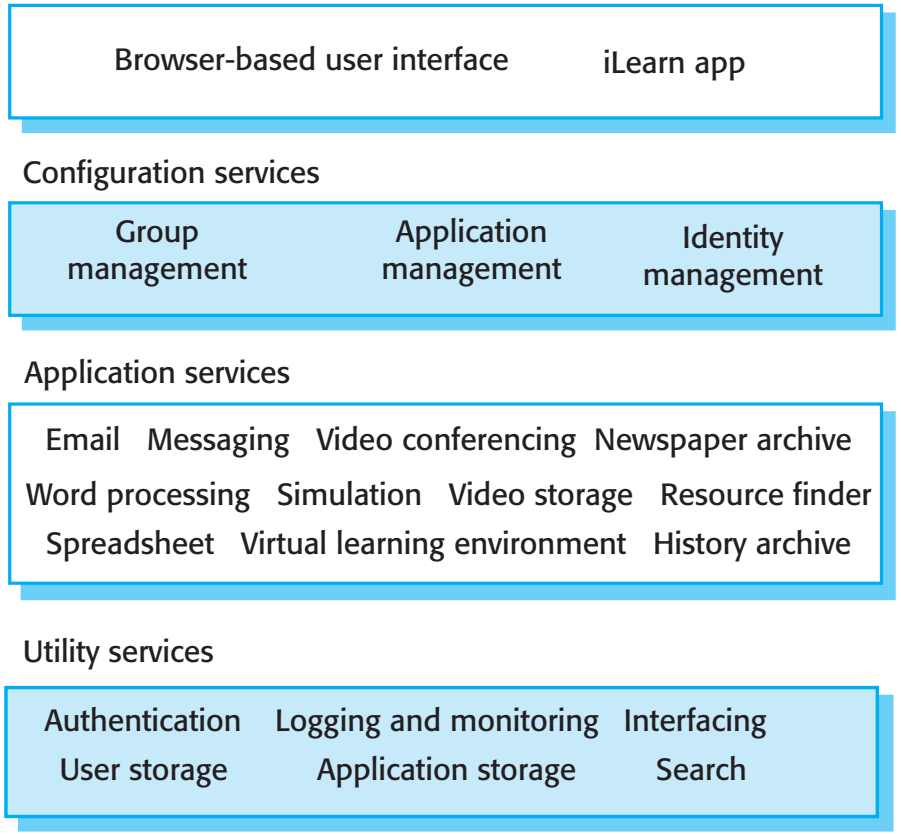
**iLearn: A digital learning environment**

- Framework in which a set of general-purpose and specially designed tools for learning may be embedded, plus a set of applications that are geared to the needs of the learners using the system.
  - The tools included in each version of the environment are chosen by teachers and learners to suit their specific needs
  - General applications such as spreadsheets
  - Learning management applications such as a Virtual Learning Environment (VLE) to manage homework submission and assessment, games and simulations.

# Service oriented

- All system components considered to be a replaceable service
- This allows the system to be updated incrementally as new services become available
- Possible to rapidly configure the system to create versions of the environment for different groups such as very young children who cannot read, senior students, etc.

# iLearn architecture




used to adapt the environment with a specific set of application services and do define how services are shared between students, teachers and their parents.

Provide specific applications such as email, conferencing, photo sharing etc. and access to specific educational content such as scientific films or historical resources

Provide basic application-independent functionality and which may be used by other services in the system