Introduction

Vũ Thị Hồng Nhạn

(vthnhan@vnu.edu.vn)

Dept. of Software Engineering, FIT, UET

Vietnam National Univ., Hanoi

Contents

- Professional **software development**
  - What does it mean by software engineering

- Software engineering ethics
  - A brief introduction to ethical issues that affect software engineering
- Case studies
  - An introduction to three examples that are used in later lectures

Professional software development

Answer 10 frequently asked questions about SE

1. What is software?

- **Software** is a **computer program** and it's associated with **documents**
- But **computer program** may be part of **a system**, so there are several programs within a system
- The **documentation** is important when we're doing SE
  - We're not just writing a program
  - We're also creating the documents and the other information we need to use over a period of time
- **Software products** can be of 2 kinds, we can have...
  - Off-the shell products or apps for general markets
  - Customer software for particular customer, which is build exactly based on the customer's specification

2. What are the attributes of good software?

- Software should do what customers wants
- & should deliver the functionality they want at a reasonable level of performance
- Should be usable, maintainable, and reliable

3. What is SE?

- **SE** as **an engineering discipline** that is concerned with all aspects of software production
  - not just coding but **everything** from understanding what the user needs to maintaining that software over the lifetime
- Engineering means that systematic and disciplined and managed approaches are used for the production of software

4. What are the fundamental SE activities?

- There are 4 activities that are always part of professional SE
  - Software specification
  - Software development
  - Software validation
  - Software evolution

5. What is the difference between

SE & computer science?

- SE is **part** of a more general computer science course
  - Or fundamentally computer science is **the theory** of software
  - Whereas SE is about **the practicalities**, is about building of software
  - and it happens that sometimes the theory and the practice don't exactly match

6. What is the difference between

Software & SE?

- We usually talk about Software systems, so what is the difference between software & SE

- SE is concerned with all aspects of computer-based systems development

  - including the **hardware,** the **software,** and **process engineering**

- Software is a more focused activity

  - focuse only on **the software,** but obviously it's closely associated and integrated with the SE process

7. Key Challenges facing in SE?

- Firstly, facing with increasing diversity,
  - Software is in all sorts of devices from smart watches up to space stations
  - We need to understand **the characteristics** of the device and the computer that we're using → to create the most appropriate software for that
- Secondly, the challenge of responsiveness or delivering the software more quickly
  - Because business & people want things to happen quickly (not wait for several years)
- Finally, delivering trustworthy software
  - Software that we can trust will be reliable, resilient, and counter external cyberattack

8. What are the costs of SE?

- This is a hard question to answer!
  - Because it varies quite dramatically for all sorts of systems
  - If we’re building a critical system in an aircraft , it’s gonna cost an awful lot per line of code than we’re building an app that’s offered for free in Android store or itunes
- Roughly 60% of software costs are development costs, 40% are testing costs (updating as things change so the software remains useful for the buyer of that software)

9. What are the best...

engineering techniques & methods

- All software projects have to be professionally managed and developed, **different techniques** are appropriate for different types of systems

- E.g., **games** should always be developed using a series of prototypes, whereas **safety critical control systems** require a complete and analyzable specification

- Therefore you cannot say that one method is better than another

10. What differences has the web made to SE?

- Fundamentally the difference **the web** has made is *that has allowed us to develop and deliver* **software** *in a new way*
- We can deliver **software services** rather than systems that are installed on a user's own computer
  - We can access these services remotely
- It allows us to build a new kind of distributed system to deliver the functionality that the user needs
- Because of **the demands** of the web and **the need** to build highly distributed systems, this has led to the advances in SE
  - New ways of **reusing software**, a new programming language & development techniques can be used

Why SE matters?

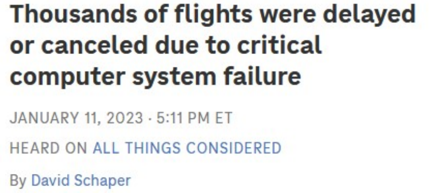
Why?

- Software is everywhere and our society is now totally dependent on software-intensive system

Power infrastructure
telecommunications

Financial system

Transport

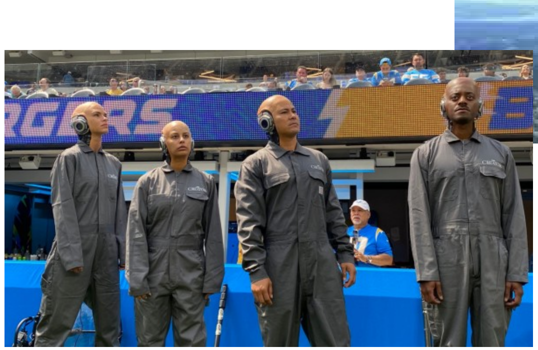


Thousands of flights were delayed or canceled due to critical computer system failure

JANUARY 11, 2023 · 5:11 PM ET

HEARD ON ALL THINGS CONSIDERED

By David Schaper

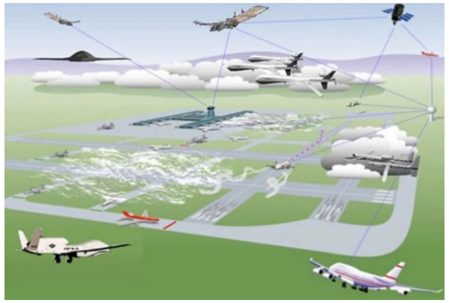


Why?

- Feeding and housing a growing world population
- Coping with the problems that come from people living longer and providing a high quality of life for our elderly citizens
- Dealing with international terrorism, scourge on all of our societies
- We need SE, efficient & effective techniques for developing complex software systems to maintain our society and to continue to provide high quality lives for our citizens
- We need to ensure these systems are reliable and secure and meet the needs of their users

Software products

- Generic products
  - Stand-alone systems that are marketed and sold to any customer who wishes to buy them
  - E.g. PC software such as graphics programs, project management tools, CAD(Computer-Aided Designs) software, software for specific markets such as appointment systems for dentists
- Customized products
  - Software that is commissioned by a specific customer to meet their own needs
  - E.g., embedded control systems, air traffic control software, traffic monitoring systems



Product specification

- Generic products
  - The specification of what the software should do is owned by *the software developer*
  - and decisions on software change are made by *the developer*
- Customized products
  - The specification of what the software should do is owned by *the customer*
  - and they make *decisions on software changes*

Essential attributes of good software

Product characteristic	Description
Maintainability	Software should be written in such a way so that it can evolve to meet the changing needs of customers. This is a critical attribute because software change is an inevitable requirement of a changing business environment
Dependability & security	Software dependability includes a range of characteristics including reliability, security, and safety. Dependable software shouldn't cause physical or economic damage in the event of system failure. Malicious users shouldn't be able to access or damage the system
Efficiency	Software shouldn't make wasteful use of system resources such as memory or processor cycles. Efficiency therefore includes responsiveness, processing time, memory utilization, etc.
Acceptability	Software must be acceptable to the type of users for which it is designed. This means that it must be understandable, usable, and compatible with others systems that they use

SE

- SE is an engineering discipline that is concerned with all aspects of software production from the early stages of system specification through to maintaining the system after it has gone into use

- Engineering discipline

  - Using appropriate theories and methods to solve problems bearing in mind organizational and financial constraints

- All aspects of software products

  - **Not just** technical process of development

  - **Also,** project management and development of tools, methods, etc. to support software production

Software process activities

- **Software specification**, Where customers and engineers define the software that is to be produced and the constraints on its operation
- **Software development**, where the software is designed and programmed
- **Software validation**, where the software is checked to ensure that it is what the customer requires (i.e. do the right things)
- **Software evolution**, where the software is modified to reflect changing *customer and market requirements*

General issues that affect software

- Heterogeneity
  - Increasingly, **systems** are required to operate as *distributed systems* across networks that include different types of computer and mobile devices
- Business & social change
  - **Business & society** are changing incredibly quickly as emerging economies develop and new technologies become available
  - They need to be able to change their existing software and to rapidly develop new software

General issues that affect software…

- Security & trust
  - As **software** is intertwined with all aspects of our lives, it is essential that we can trust that software
- Scale
  - Software has to be developed **across** a very wide range of scales, **from** very small embedded systems in portable or wearable devices **through to** Internet-scale, cloud-based systems that serve a global community

SE diversity

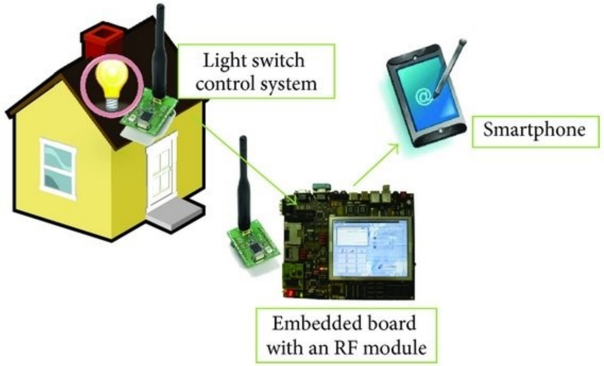
- There are **many different types** of software systems and there’s no universal set of software techniques that is applicable to all of these

- **The SE methods** and **tools** used **depend on** the type of application being developed, the requirements of the customer and the background of the development team

Application types

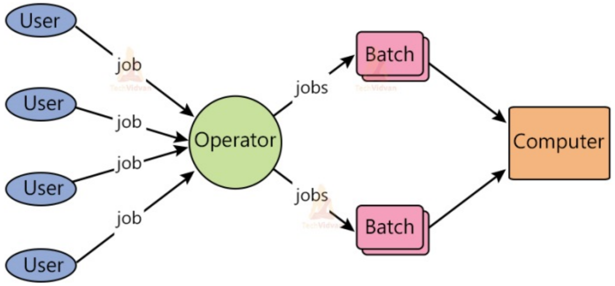
- Stand-alone applications
  - These are **application systems** that run on a local computer, such as a PC
  - They include all necessary functionality and do not need to be connected to a network
- Interactive transaction-based applications
  - Applications that execute **on** a remote computer and are accessed by users from their own PCs or terminals
  - These include web applications such as e-commerce applications
- Embedded control systems
  - These are software control systems

that control and manage hardware devices



Application types…

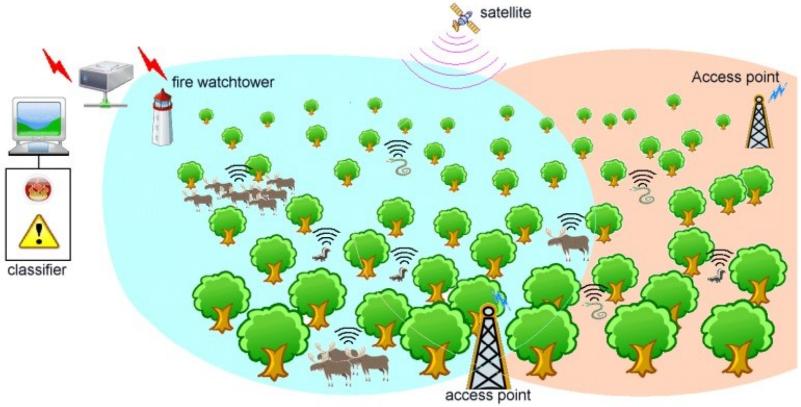
- Batch processing systems
  - These are business systems that are designed to process data in large batches
  - They process large numbers of individual inputs to create corresponding outputs



- Entertainment systems
  - These are systems that are primarily for personal use and which are intended to entertain the user
- Systems for modeling and simulation
  - These are systems that are developed by scientists and engineers to model physical processes or situations, which include many separate interacting objects

Application types...

- Data collection systems
  - These are systems that collect data from their environment using a set of sensors and send that data to **other systems** for processing



Forest fire detection

- Systems of systems
  - These are that are composed of a number of other software systems

SE fundamentals

- Some fundamental principles applied to all types of software systems, irrespective of the development techniques used
  - Systems should be developed using a managed and understood development process. Of course, different processes are used for different types of software
  - Dependability and performance are important for all types of system
  - Understanding and managing the software specification and requirements are important
  - **Where appropriate**, you should reuse software that has already been developed rather than rewriting new software

Internet SE

- **The web** is now **a platform** for **running application** and organizations are increasingly developing web-based systems rather than local systems
- Web services allow application functionality to be accessed **over the web and internet**
- Cloud computing is an approach to the provision of computer services where **applications** run remotely on the cloud
  - users do not buy software but pay according to use

Web-based SE

- **Web-based systems** are complex distributed systems **but** the fundamental principles of SE discussed previously are applicable to them (the same way they are applied to other types of system)

Web SE

- Software reuse
  - **Software reuse** is the dominant approach for constructing web-based systems
  - When building these systems, you think about how you can assemble them from pre-existing software components & systems
- Incremental & agile development
  - **Web-based systems** should be developed and delivered **incrementally**
  - It's now generally recognized that it's impractical to specify all the requirements for such systems in advance

Web SE…

- Service-oriented systems
  - **Software** may be implemented using **service-oriented SE**, where the software components are stand-alone web services
- Rich interfaces
  - **Interface development technologies** such as AJAX and HTML5 have emerged that support the creation of rich interfaces within a web browser

SE Ethics

What is Ethics?

- Ethics & codes of ethics related to professions in computing & science is fairly simple
  - it’s behavior that we consciously make, that has positive or negative impacts on society and citizens and the environment (cultural & natural) and so on
- Generally there are 2 kinds of ethics
  - **Minimalism** doesn’t cause **harm**
  - **Idealism**: a fun side of computing which says **I can** actually prevent harm, maybe actually do some positive things and then sometimes get labeled as **ethical**

SE ethics

- SE involves wider **responsibilities** than simply the application of technical skills
- Software engineers must behave in an honest and ethically responsible way if they are to be able to be respected as **professionals**
- **Ethical behavior** is more than simply upholding the law but involves following a set of principles that are morally correct

Issues of professional responsibility

- Confidentiality (bảo mật)
  - Engineers should normally respect the confidentiality of their employers or clients irrespective of **whether or not** a formal confidentiality agreement has been signed
- Competence (chức vụ/năng lực/thẩm quyền)
  - Engineers shouldn’t misrepresent their level of competence
  - They shouldn’t knowingly accept work which is out with their competence

Issues of professional responsibility

- **Intellectual property rights**
  - Engineers should be aware of local laws governing the use of intellectual property such as patents, copyright, etc.
  - They should be careful to ensure that their intellectual property of employers and clients is protected
- **Computer misuse**
  - Software engineers shouldn't use their technical skills to misuse other people's computers
  - Computer misuse ranges from relatively trivial (game playing on an employer's machine, say) to extremely serious (dissemination of viruses)

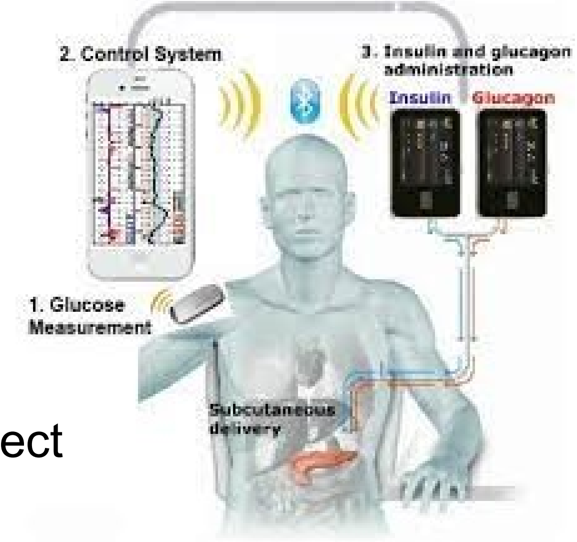
Case studies

case studies

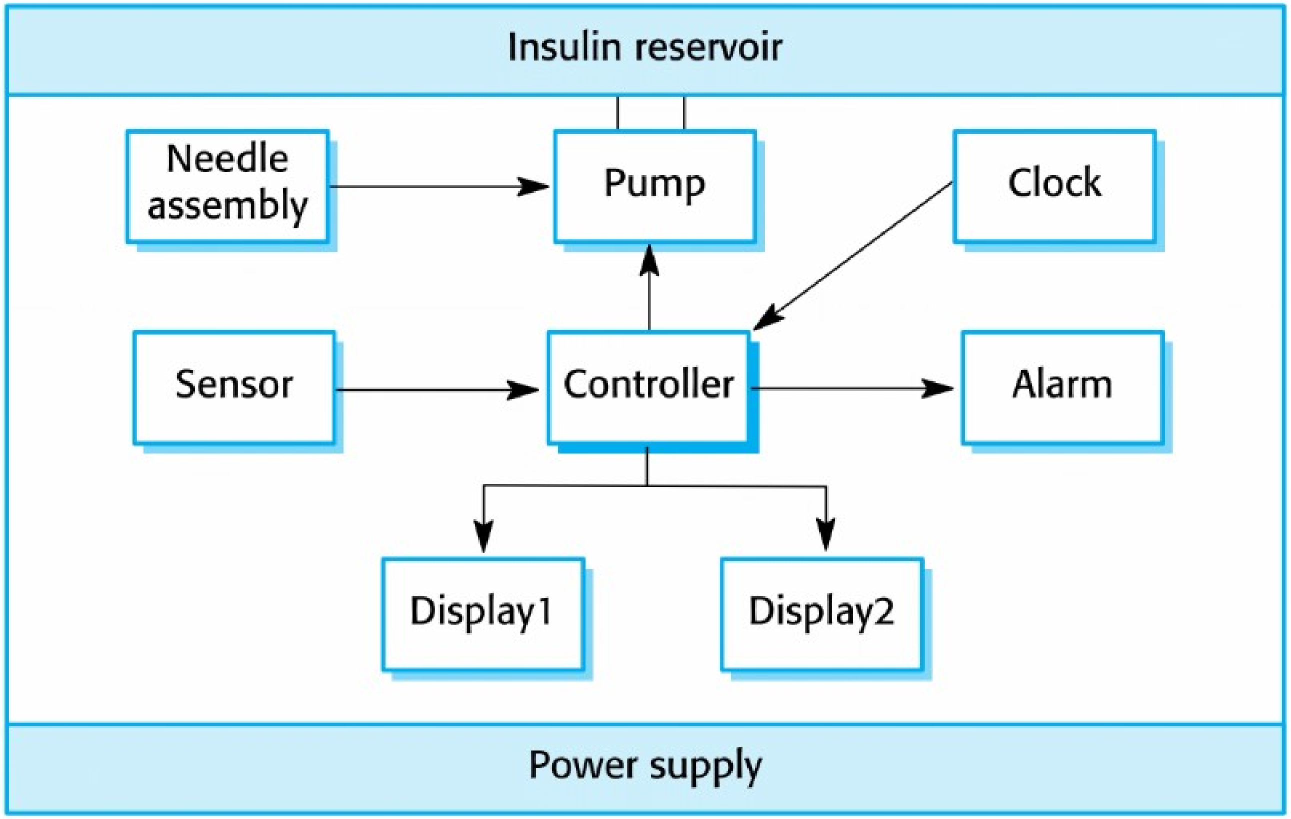
- A personal insulin pump
  - **An embedded system** in an insulin pump used by diabetics to maintain blood glucose control
- A mental health care patients management system
  - Mentcare**: a system** used to maintain records of people receiving care for mental health problems
- A wilderness weather station
  - **A data collection system** that collects data about weather conditions in remote areas
- iLearn: a digital learning environment
  - service-oriented systems

Insulin pump control system

- 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 blood sugar levels can lead to *brain malfunctioning, coma and death*; high-blood sugar levels have long-term consequences such as eye and kidney damage



Insulin pump hardware architecture



Essential high-level requirements

- The system shall be **available to deliver insulin** when required

- The system shall **perform reliably** and deliver the correct amount of insulin to counteract the current level of blood sugar

- The system must therefore be designed and implemented to ensure that the system **always meets** these requirements

Mentcare

A patient information system for mental health care

- A patient information system to support mental health care is a medical information system
  - that maintains information about patients suffering from mental health problems and the treatments that they have received
- **Most mental health patients** do not require dedicated hospital treatment
  - but need to attend specialist clinics regularly where they can meet a doctor who has detailed knowledge of their problems
- To make it easier for patients to attend, these clinics are not just run in hospitals
  - they may be held in local medical practices or community centers

Mentcare

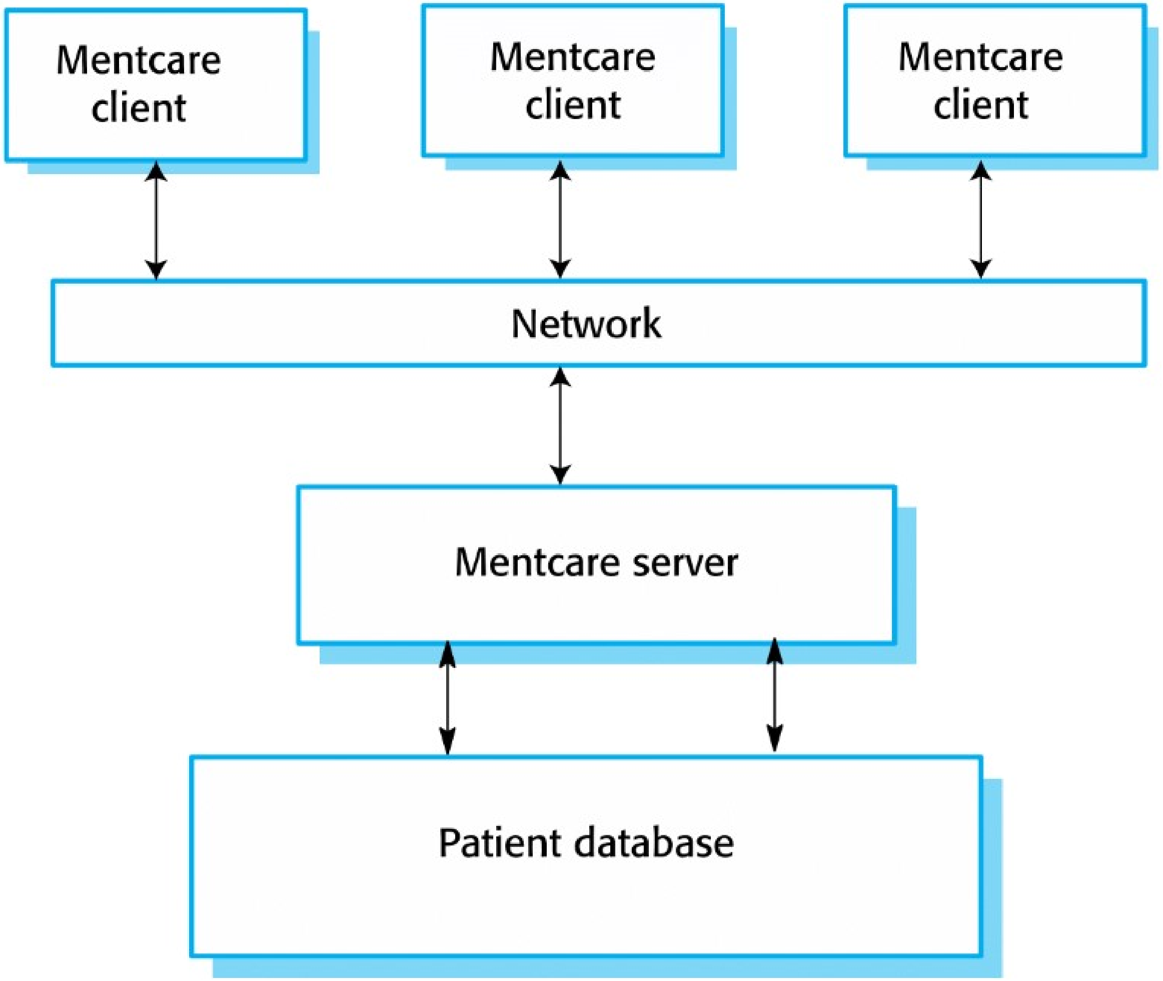
- Mentcare is an information system that is intended for use in clinics
- It makes use of a centralized database of patient information
  - but has also been designed to run on a PC,
  - so that it may be accessed and used from **sites** that don’t have secure network connectivity
- When **the local systems** have secure network access, they use patient information in the database,
  - they can download and use copies of patient records when they are disconnected

Mentcare goals

- To generate management information that allows health service managers to assess performance against local and government targets

- To provide medical staff with timely information to support the treatment of patients

The organization of the Mentcare system



Key features of the Mentcare system

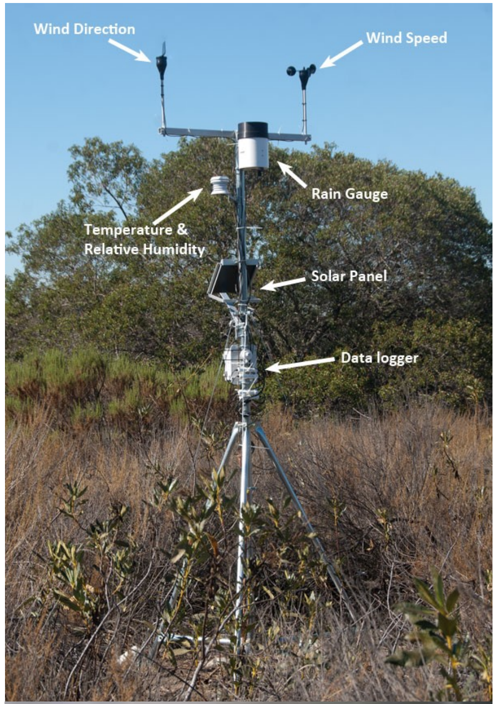
- Individual care management
  - Clinicians can support records for patients, edit the information in the system, view patient history, etc.
  - The system supports data summaries so that doctors can quickly learn about their key problems and treatments that have been prescribed
- Patient monitoring
  - 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/admitted to hospital,
  - the drugs prescribed and their costs, etc.

Mentcare system concerns

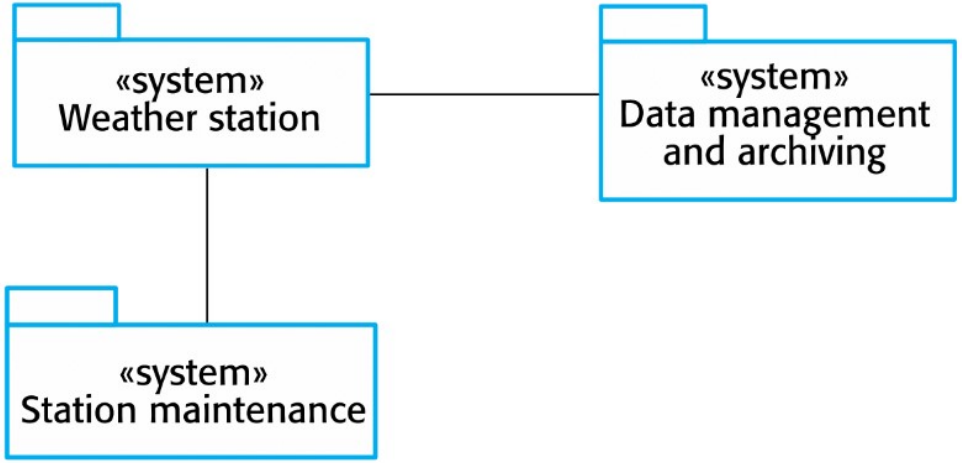
- **Privacy**
  - It is essential that patient information is confidential and is never disclosed to anyone apart from authorized medical staff and the patient themselves
- **Safety**
  - Some mental illness cause patients to become suicidal or a danger to other people
  - Whenever possible, the system should warn medical staff about potentially suicidal or dangerous patients
  - The system must be available when needed, otherwise patient **safety** may be compromised **and** it may be impossible to prescribe the correct medication to patients

Wilderness weather station

- The government of a country with a large area of wilderness decides to deploy several hundred weather stations in **remote areas**
- **Weather stations** collect data from a set of instruments that...
  - measure weather parameters such as the wind speed and directions, the ground and air temperatures, the barometric pressure and the rainfall over a 24 hour period
  - Each of these instruments is controlled by **a software system** that takes parameter readings periodically and **manages** the data collected from the instruments



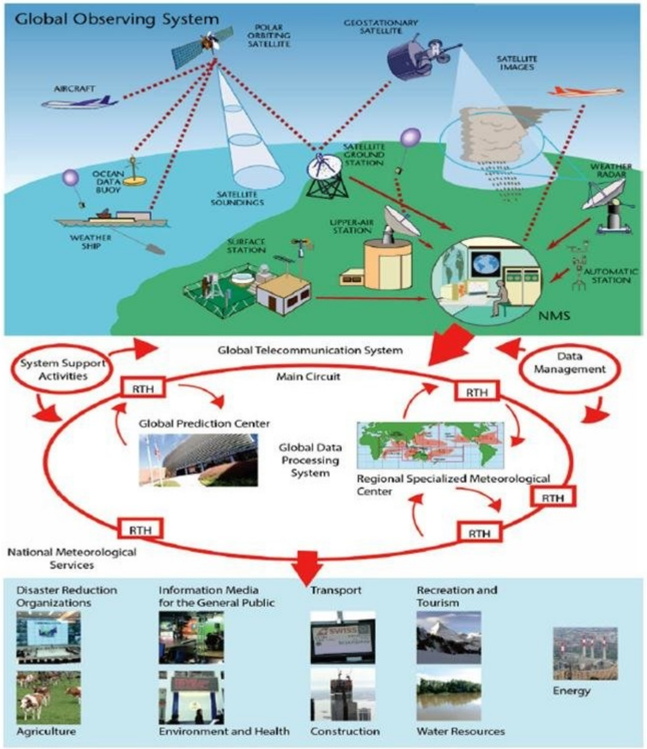
The weather station’s environment



«system»
Weather station

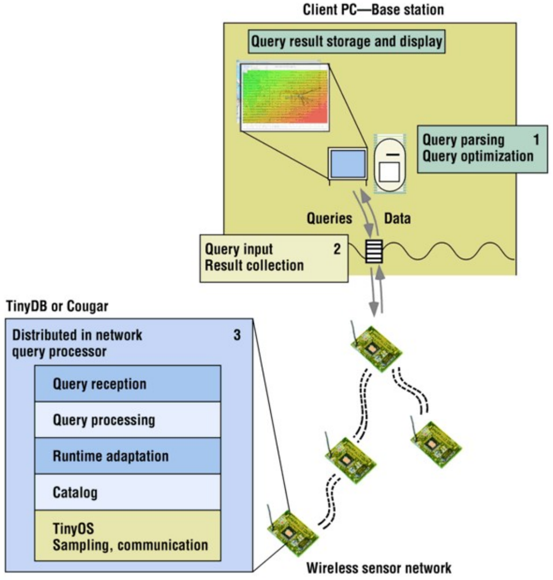
«system»
Data management
and archiving

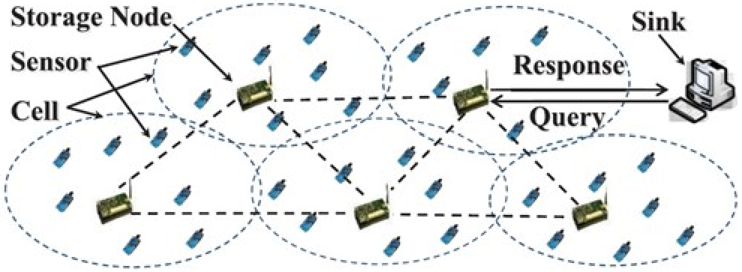
«system»
Station maintenance



Weather information system

- The weather station system
  - This is responsible for **collecting** weather data, carrying out some initial data processing and transmitting it to the data management system
- The data management and archiving system
  - This system **collects** the data from all of the wilderness weather stations, carries out data processing and analysis and **archives** the data
- The station maintenance system
  - This system can communicate by **satellite** with all wilderness weather stations to monitor the health of these systems and provide reports of problems





iLearn: a digital learning environment

- A digital learning environment is a framework in which
  - a set of general-purpose and specialty designed **tools** for learning may be embedded
  - plus a set of applications that are geared to the needs to 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
  - These can be general applications such as spreadsheets
  - learning management applications such as a Virtual Learning Environment to mange homework submission and assessment, games and simulations

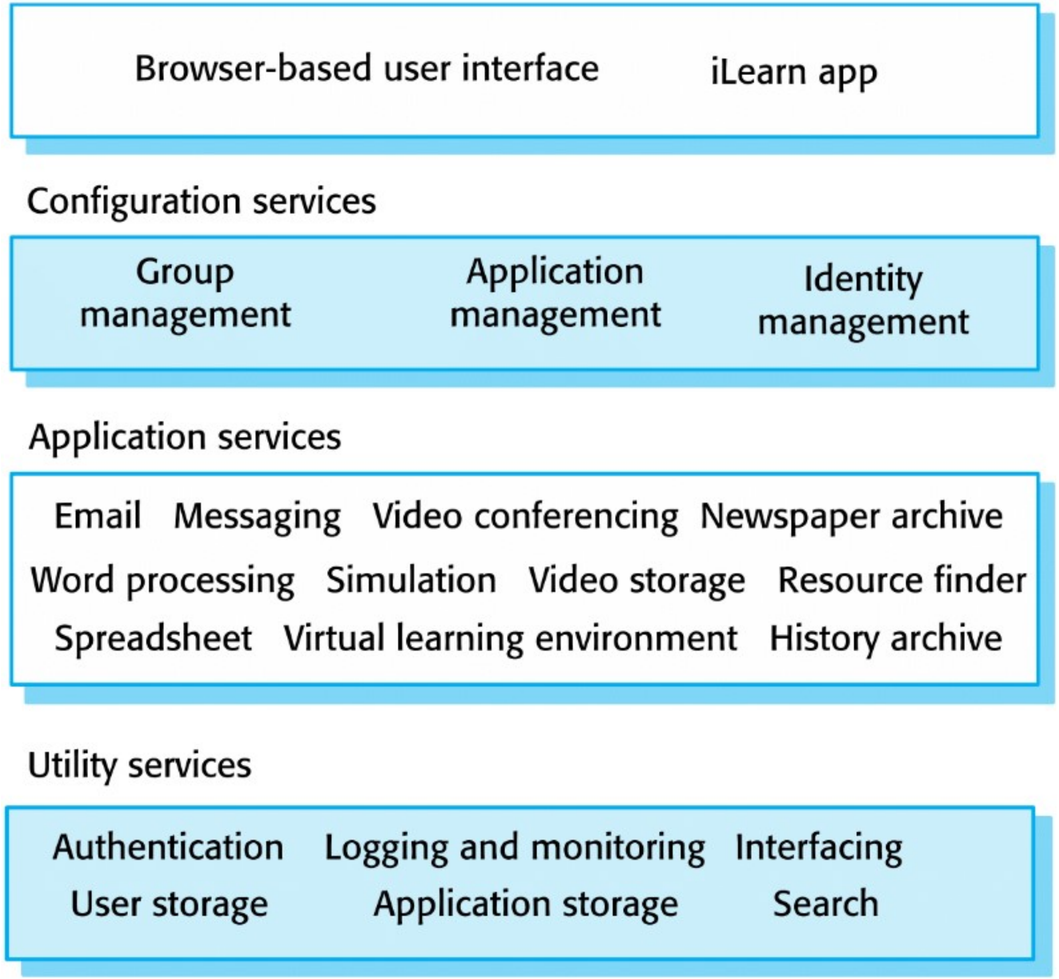
iLearn: service-oriented systems

- The system is a service-oriented system with **all system components** considered to be a replaceable service
- This allows the system to be updated incrementally as **new services** become available
- It also makes it 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 services

- **Utility services** that provide basic application-independent functionality and which may be used by other services in the system
- **Application services** that provide specific applications such as email, conferencing, photo sharing etc. and access to specific educational content such as scientific films or historical resources
- **Configuration services** that are 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

iLearn architecture



iLearn service integration

- **Integrated services** are services which offer an API and which can be accessed by other services through that API
  - Direct service-to-service communication is therefore possible
- **Independent services** are services which are simply accessed through a browser interface and which operate independently of other services
  - **Information** can only be shared with other services through explicit user actions such as copy and paste;
  - **re-authentication** may be required for each independent service

Key points

- Software engineering is an engineering discipline that is concerned with all aspects of **software production**
- Essential software product attributes are maintainability, dependability and security, efficiency and acceptability
- The high-level activities of specification, development, validation and evolution are part of all software process
- The fundamental notions of SE are universally applicable to all types of system development