CMPE 312

SOFTWARE ENGINEERING

Chapter 1- Introduction

Topics covered

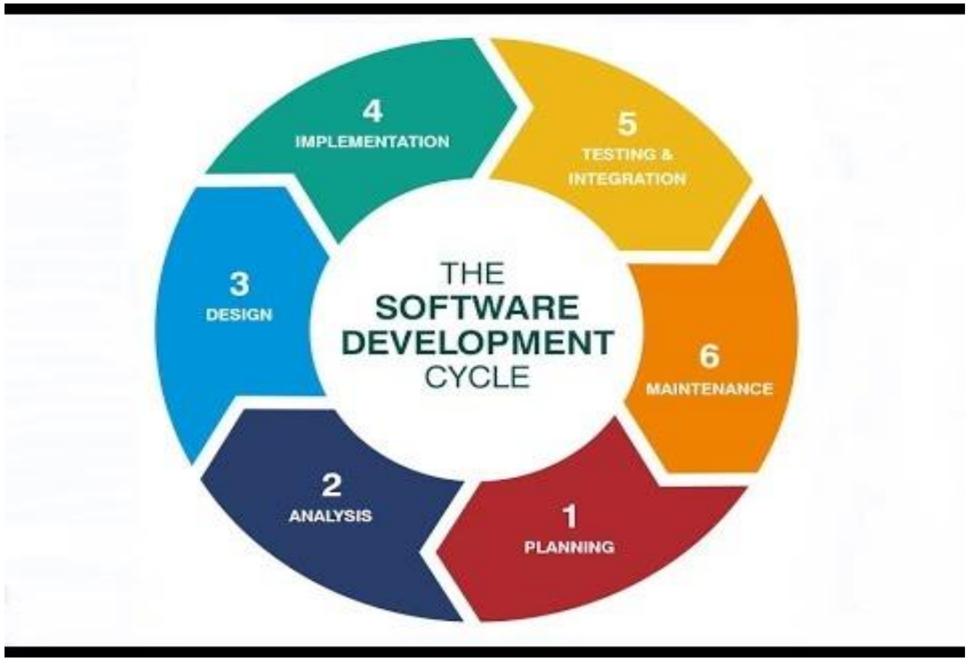
- Professional software development
  - What is meant by **software engineering?**
- Software engineering ethics
  - A brief introduction to **ethical issues** that affect software engineering.
- Case studies
  - An **introduction to FOUR examples** that are used in later chapters in the book.

Software engineering

- The <u>economies of ALL developed nations/countries</u> are dependent on software.
- More and more systems are software controlled anymore…
- Software engineering is related <u>with **theories, methods and tools**</u> for professional software development.
- Software expenditure/costs represents <u>a significant portion of Gross national product (GNP)</u> in all developed countries.

This is your customer! Customer Firm --requires a SW product from your company--
This is you! A SW company --you work for / your own--

Software costs



- ✧ The costs of software on a PC are often **greater** than the hardware cost.
- ✧ MAINTENANCE cost is generally more than DEVELOPING cost of a SW prj.
  - For systems with a long life, maintenance costs may be several times development costs.
- ✧ **SOFTWARE ENGINEERING is about PRODUCING cost-effective software products.**

Software project failure

- ✧ Due to increased competition in markets, we know that the SW products complexity is increasing day by day.
- ✧ New software engineering techniques help us:
  - To create larger systems,
  - To create more complex systems,
  - Satisfy the demands changes in many sectors.
- ✧ Therefore, many sectors expectations:
  - SWs **have to be built** and **delivered** more **quickly**;
  - **Larger** and **more complex** SWs are required;
  - SWs must **have new capabilities/features.**
    - These features were **previously considered** to be **impossible**.

Software project failure

✧ ***Due to failure to use SE methods***

- Software products are too costly!
- Today, many companies have started to develop their SWs for performing their business processes.
- They want to develop their own SWs without using SE *methods, tools* and *techniques*. But this is wrong way!
- Also, since this is not their expertise, so they cannot use SE methods,tools & techniques very well.
  - For example. Health sector: hospitals, clinics, etc.
- As a result, their SWs produced are *often more expensive* and *less reliable* than it should be.

5 Signs your Project is Going Wrong

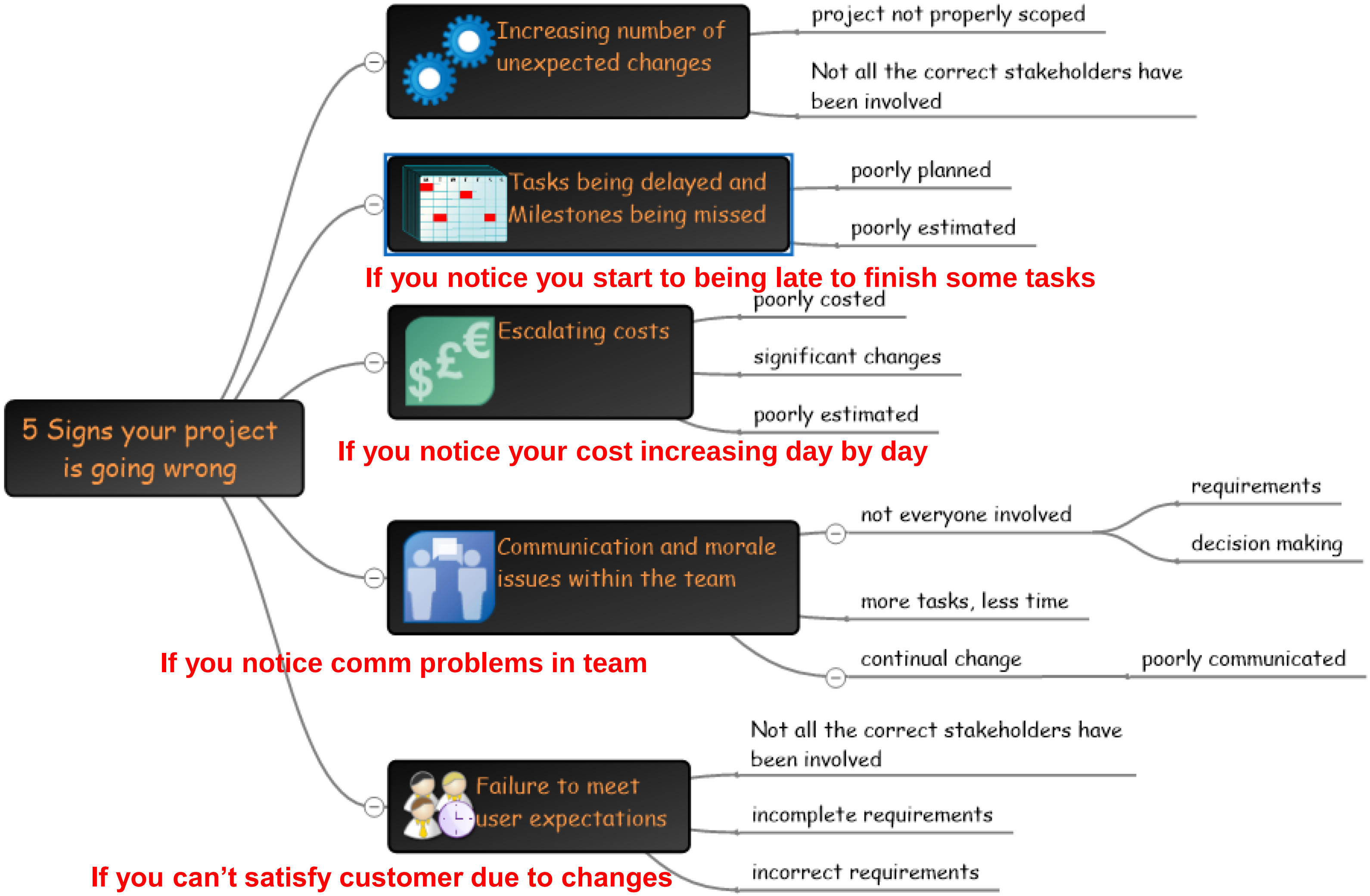
If you notice that too much changes are required…

If you notice you start to being late to finish some tasks

If you notice your cost increasing day by day

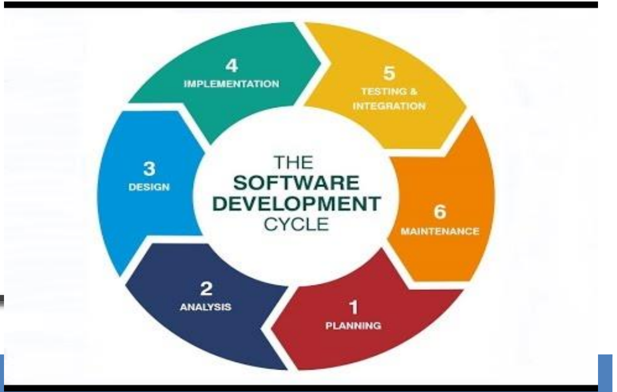
If you notice comm problems in team

If you can't satisfy customer due to changes


5 Signs your project is going wrong
Increasing number of unexpected changes
project not properly scoped
Not all the correct stakeholders have been involved
Tasks being delayed and Milestones being missed
poorly planned
poorly estimated
Escalating costs
poorly costed
significant changes
poorly estimated
Communication and morale issues within the team
not everyone involved
requirements
decision making
more tasks, less time
continual change
poorly communicated
Failure to meet user expectations
Not all the correct stakeholders have been involved
incomplete requirements
incorrect requirements

Professional software development

Frequently asked questions about SE



Question	Answer
What is software?	Computer programs and related documentation. Software products may be developed for a particular customer or may be developed for a general market.
What are the attributes **of good software**?	Good software should deliver the required functionality and performance to the user. Good software should be **maintainable, dependable and usable**.
What is software engineering?	Software engineering is an engineering discipline that is concerned with all aspects of software production.
What are the fundamental software engineering activities?	1-Software specification, 2-software development, 3-software validation and 4-software evolution.
What is the difference between software engineering and computer science?	Computer science focuses on **theory and fundamentals**; software engineering is concerned with the practices **of developing and delivering useful software**.
What is the difference between software engineering and system engineering?	System engineering is about with all aspects of computer-based systems development including ***hardware, software and process engineering***. Software engineering is part of this more general process.

Frequently asked questions about software engineering

Question	Answer
What are the key challenges facing software engineering?	Coping with increasing diversity, Demands for reduced delivery times and Developing reliable software.
What are the costs of software engineering?	Roughly 60% of software costs are development costs, 40% are testing costs. **For custom software, evolution costs often exceed development costs.**
What are the best software engineering techniques and methods?	While all software projects have to be professionally managed and developed, <u>different techniques are suitable for different types of system</u>. *For example, games should always be developed using a series of prototypes whereas* ***safety critical control systems*** *require a complete and analyzable specification to be developed. You can't, therefore, say that one method is better than another.*
What differences has the web made to software engineering?	The web has led to the availability of software services and the possibility of developing highly distributed service-based systems. Web-based system development provides to important advances in **<u>programming languages and software reuse</u>**.

Software products

- Generic products
  - Stand-alone systems that are marketed and sold to any customer who wishes to buy them.
  - Examples – PC software such as graphics programs, project management tools; CAD software; software for specific markets such as appointments systems for dentists.
- Customized products
  - Software that is used by a specific customer to meet their own needs.
  - Examples – embedded control systems, air traffic control software, traffic monitoring systems.

Product specification

- Generic products
  - "What the software will do" depends on the software developer.
  - Decisions about software change are made by the developer.
- Customized products
  - "What the software will do" depends on customers (needs).
  - Decisions about software change (requirements) are made by the customers.

Crucial attributes of a good software

Product characteristic	Description
Maintainability	• Software should be written/developed 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 (trustworthy) and Security	• Software dependability includes some characteristics; **1- Reliability, 2- Security & Safety**. • Dependable software **should not cause physical or economic damage** in the event of **system failure**. • Malicious users should not be able to access or damage the system.
Efficiency	• Software should not make wasteful use of system resources such as memory and processor cycles. • **Efficiency therefore includes responsiveness, processing time, memory utilisation**, etc.
Acceptability	• Software must be acceptable by the end **users, which is designed for them**. • This means that it **must be understandable, usable** and **compatible with other systems used** in the customer company.

Software engineering

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

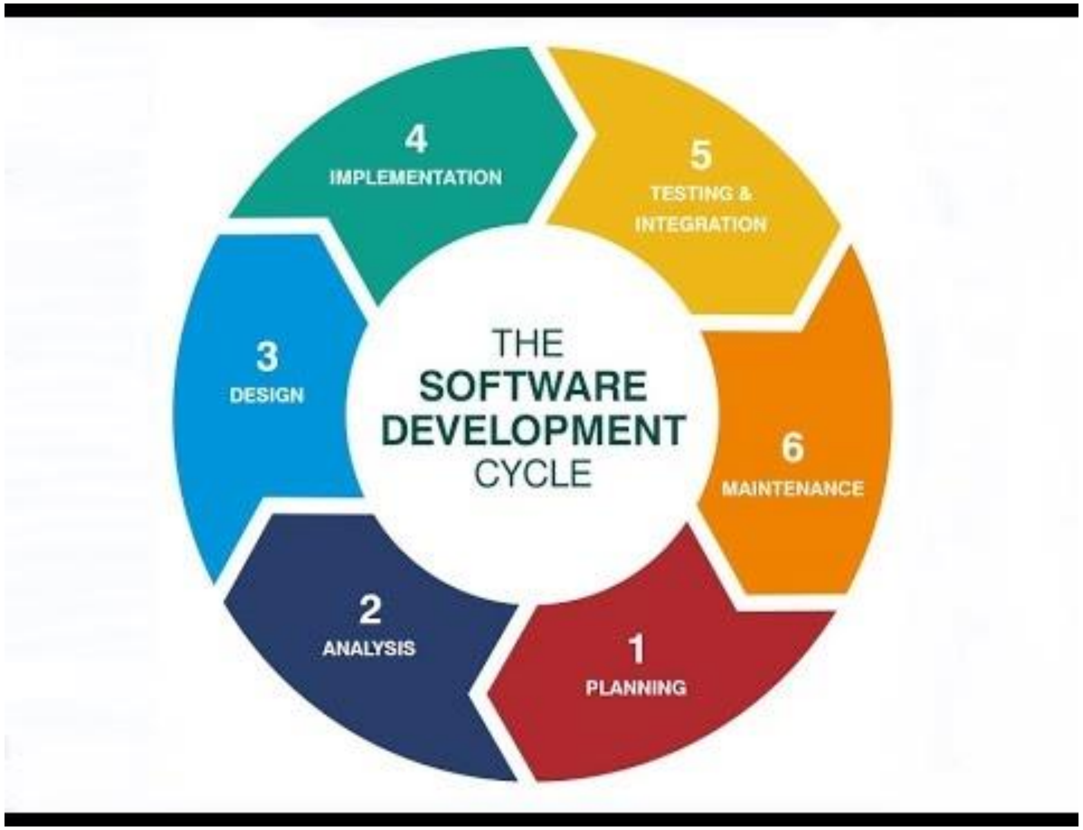
2- Engineering discipline ?

uses **appropriate theories and methods to solve problems** by considering organizational and financial constraints.

3- What are the aspects of software production ?

Don't think of producing software as just program coding!

Also→ project management and the using of SE tools, methods etc. supports to software production.



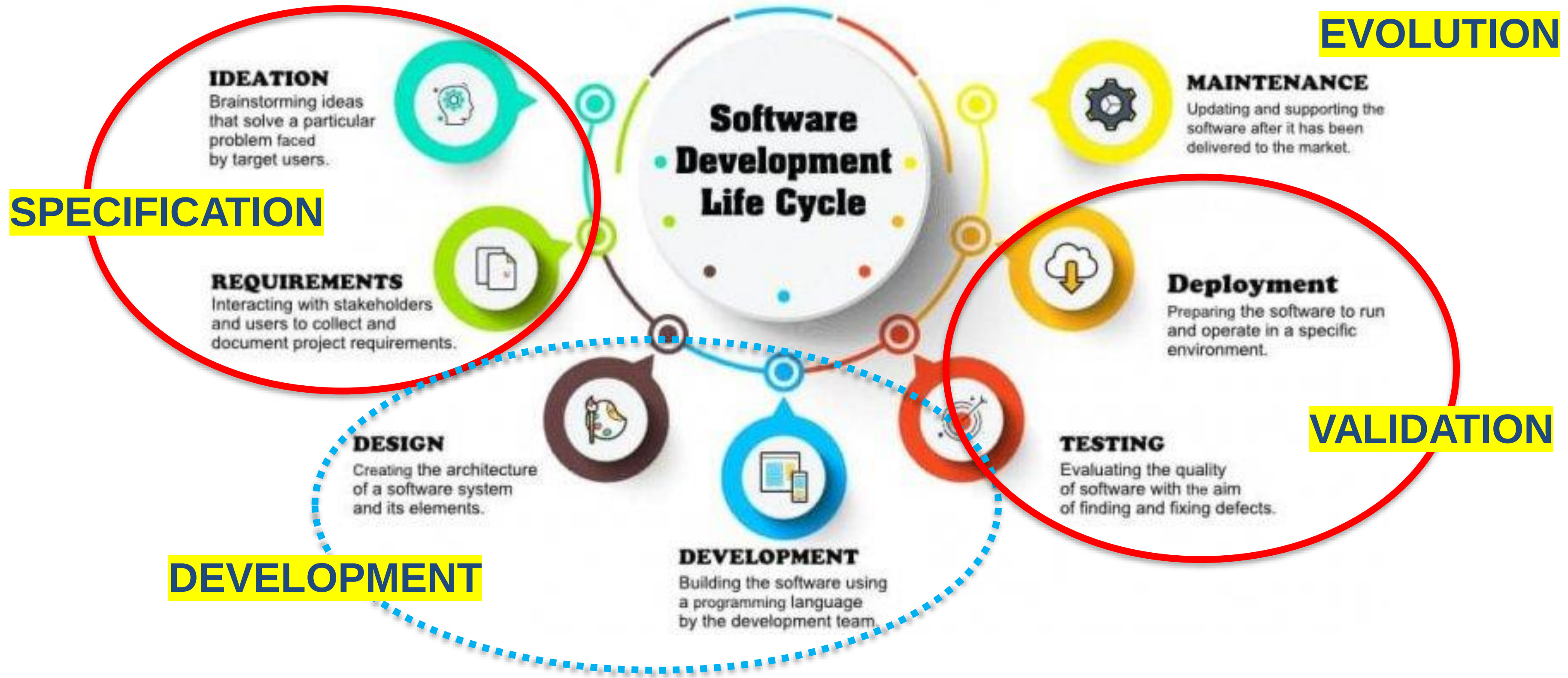
Importance of software engineering

More and more individuals and society **want to trust advanced** software systems.

So, we need to be able to **produce reliable and trustworthy systems** *economically* and *quickly*.

It is usually cheaper, in the long run, **to use SE methods and techniques for software systems** rather than just write the programs as if it was a personal programming project.

For most types of SW systems, **the major costs are the costs of changing the software** after starting use it.


EVOLUTION
SPECIFICATION
VALIDATION
DEVELOPMENT
Software Development Life Cycle
IDEATION
Brainstorming ideas that solve a particular problem faced by target users.
REQUIREMENTS
Interacting with stakeholders and users to collect and document project requirements.
DESIGN
Creating the architecture of a software system and its elements.
DEVELOPMENT
Building the software using a programming language by the development team.
TESTING
Evaluating the quality of software with the aim of finding and fixing defects.
Deployment
Preparing the software to run and operate in a specific environment.
MAINTENANCE
Updating and supporting the software after it has been delivered to the market.

SOFTWARE PROCESS ACTIVITIES

1. Software specification, where customers and engineers define the software details that is to be produced and the constraints on its operation.
2. Software development, where the software is designed and programmed.
3. Software validation, where the software is checked to ensure that it is what the customer requires.
4. Software evolution, where the software is modified to reflect changing customer and market requirements.

General issues that affect software
Reasons for changes in organizations?

1. **Heterogeneity**
   - Increasingly, systems are required to operate/run as distributed systems across networks that *may include* ***different types of computers and mobile devices*** *such as: smart phones, smart watchs, smart TVs etc...*.
2. **Business and social change**
   - Business and society are changing incredibly quickly;
     - as emerging economies develop and
     - as new technologies become available.
   - So, companies/sectors need to change their existing software and need to rapidly develop new software.

General issues that affect software

3. Security and trust

- *Because* softwares affects every aspect of our lives, *it is important that* we need to trust softwares

4. Scale

- Software **has to be developed** in 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.*

Software engineering diversity

- ✧ There are many different types of software systems and there is no universal set of software developing techniques/ methods that is applicable to all of these.
- ✧ Used SE methods and tools change acc to;
  - *1) the type of application being developed*,
  - *2) the requirements of the customer* and,
  - *3) the background of the development team*.

Application types

1. 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**.
2. 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**.
3. Embedded control systems
   - These are **software control systems** **that control and manage hardware devices**. Numerically, there are probably more embedded systems than any other type of system.

Application types

4. Batch processing systems

- These are **business systems** that are **designed to process data in large batches**.
- They **process** **large numbers of individual inputs** to obtain corresponding outputs.

5. Entertainment systems

- These are **systems that** **are primarily for personal use,** and which are intended to entertain the user.

6. 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

7. 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.

8. Systems of systems:

These are **systems that are composed of a number of other software systems**.

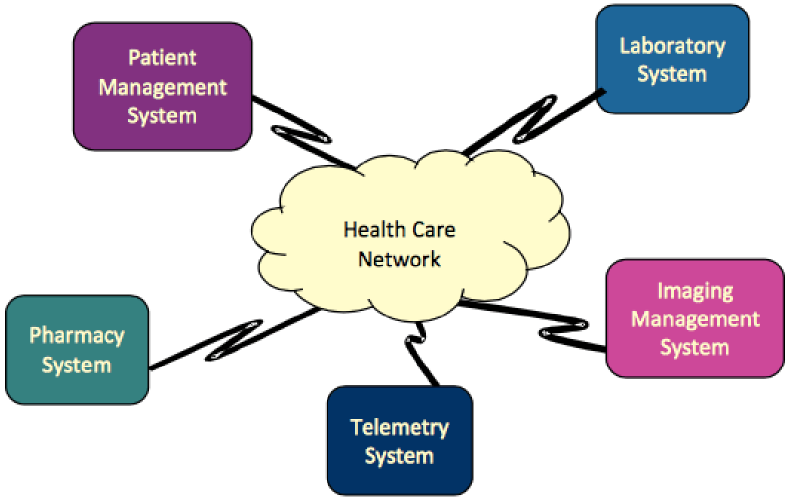


Figure 1: Example Heathcare SoS.

Software Engineering Principles

✧ **Some basic principles** **we can** apply to **all types of software systems** (*not related with development techniques used*):

1. **SELECT A SDLC MODEL:** Systems should be developed by **using a well-defined and managed development process**. Of course, different processes are used for different types of software.

2. **DEPENDABILITY and PERFORMANCE** are important for all types of system.

3. **UNDERSTAND YOUR CUSTOMER WELL:** Understanding and managing the software **specification** and **requirements** (what the software should do) **are important**.

4. **REUSE COMPONENTS:** Where appropriate, **you can reuse software** that has already been developed rather than write a new software.

Internet software engineering

- ✧ 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.
- ✧ **Cloud computing** is an approach to the provision of computer services where applications run remotely on the 'cloud'.
  - Users do not buy licenced software anymore, but they want to pay according to use.

Web-based software engineering

✧ **Web-based systems** are **complex distributed systems**

- however, the **basic principles of SE** mentioned earlier **are also suitable** for Web-based systems as well as other systems.
  - ***Let's remember the SE principles*:**
    - ***A manageble and understandable SW development process is necessary.***
    - ***Dependability and performance are important criterias.***
    - ***Understanding and managing SW requirements is necessary.***
    - ***You can reuse software.***

✧ The **fundamental ideas of SE** apply to web-based systems in the same way that they apply to other types of SW system.

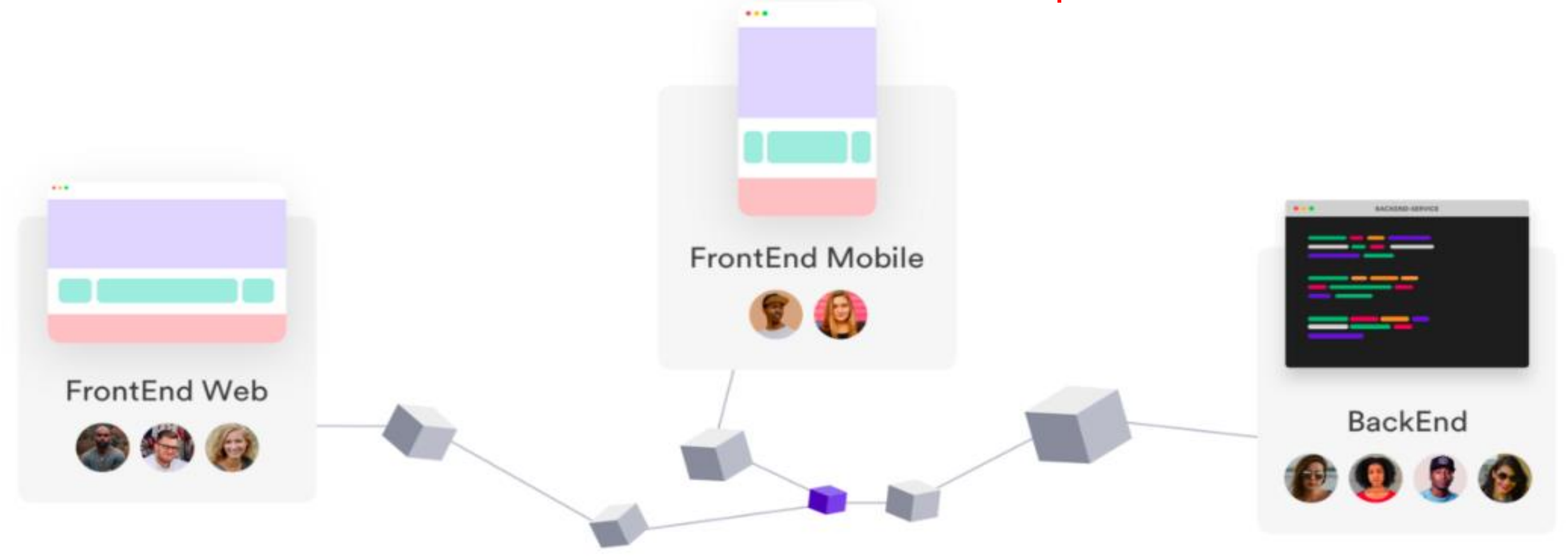
Web software engineering

- **Software reuse**
  - **Software reuse is the dominant approach** for constructing web-based systems.
  - When building a Web system, you think about **how you can combine** the system from pre-existing software components and systems. *(e.g. Credit card payment module + customer auth. module)*
- **Incremental and agile development**
  - Web-based systems **should be developed** and **delivered incrementally** → when it is impossible to **specify entire customer requirements in advanc**e for such systems.

Web software engineering

- **Web Service Oriented**: Software **is implemented using service-oriented** software engineering, where the **software components are** 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.

Service-oriented Software Example



Software engineering ethics

Software engineering ethics

- Software engineering **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 respected as professionals.
- **Ethical behaviour** is more than simply continuation the law but **involves following a set of principles** that are morally correct.

Issues of professional responsibility

✧**Confidentiality (privacy)**

•Engineers **should normally respect the confidentiality of their employers or customers** irrespective of whether or not **a formal confidentiality agreement (NDA)** has been signed.

✧**Competence (yetkinlik)**

- Engineers in a company **should not misrepresent** their level of competence.
- They **should not accept a work** which is out with their competence.

Issues of professional responsibility

- **Intellectual Property Rights** (Fikri Mülkiyet Hakları)
  - Engineers **<u>should be aware of local laws</u>** governing the use of intellectual property such as *patents, copyright*, etc.
  - They **<u>should be careful to ensure that</u>** the intellectual property of *employers and clients* is protected.
- **Computer misuse**
  - Software engineers **<u>should not use their technical skills</u>** to misuse others' computers (kötüye kullanmamalıdır).
  - **<u>Computer misuse ranges;</u>** from relatively trivial (*game playing on an employer's machine, say*) to extremely serious (*dissemination of viruses*).

ACM/IEEE Code of Ethics

- The **professional societies in the US** have cooperated to **produce a code of ethical practice**.
- Members of these organisations **sign up to the code of practice** when they join.
- The **Code contains eight Principles** related to the behaviour of and decisions **made by** professional software engineers, including practitioners, educators, managers, supervisors and policy makers, as well as trainees and students of the profession.

Rationale/reason of the code of ethics

- *Computers have a central and growing role in commerce, industry, government, medicine, education, entertainment and society at large.*
- *Software engineers are those who contribute by direct participation or by teaching, to the analysis, specification, design, development, certification, maintenance and testing of software systems.*
- *Because of their roles in developing software systems, software engineers have significant opportunities to do good or cause harm, to enable others to do good or cause harm, or to influence others to do good or cause harm.*
- *To ensure, as much as possible, that their efforts will be used for good, software engineers must promise themselves to making SE a beneficial and respected profession.*

The ACM/IEEE Code of Ethics

Software Engineering Code of Ethics and Professional Practice
(Yazılım Mühendisliği Etik Kuralları ve Mesleki Uygulama)

ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices

PREAMBLE
The short version of the code summarizes aspirations at a high level of the abstraction; the clauses that are included in the full version give examples and details of how these aspirations change the way we act as software engineering professionals. Without the aspirations, the details can become legalistic and tedious; without the details, the aspirations can become high sounding but empty; together, the aspirations and the details form a cohesive code.

Software engineers shall commit themselves to making the analysis, specification, design, development, testing and maintenance of software a beneficial and respected profession. In accordance with their commitment to the health, safety and welfare of the public, software engineers shall adhere to the following Eight Principles:

Ethical principles

1. PUBLIC - SEs shall act consistently with the public interest.

2. CLIENT AND EMPLOYER - SEs shall act in a manner that is in the best interests of their client and employer consistent with the public interest.

3. PRODUCT - SEs shall ensure that their products and related modifications meet the highest professional standards possible.

4. JUDGMENT - SEs shall maintain integrity and independence in their professional judgment.

5. MANAGEMENT – SE managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.

6. PROFESSION – SEs shall advance the integrity and reputation of the profession consistent with the public interest.

7. COLLEAGUES - SEs shall be fair to and supportive of their colleagues.

8. SELF - SEs shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.

Ethical dilemmas (etik ikilemler)

- ✧ Disagreement in principle with the policies of senior management in the company you worked for.
- ✧ Your employer acts in an unethical way and releases a safety-critical system without finishing the testing of the system (e.g. a medical SW).
- ✧ Participation in the development of military weapons systems or nuclear systems.

Case studies

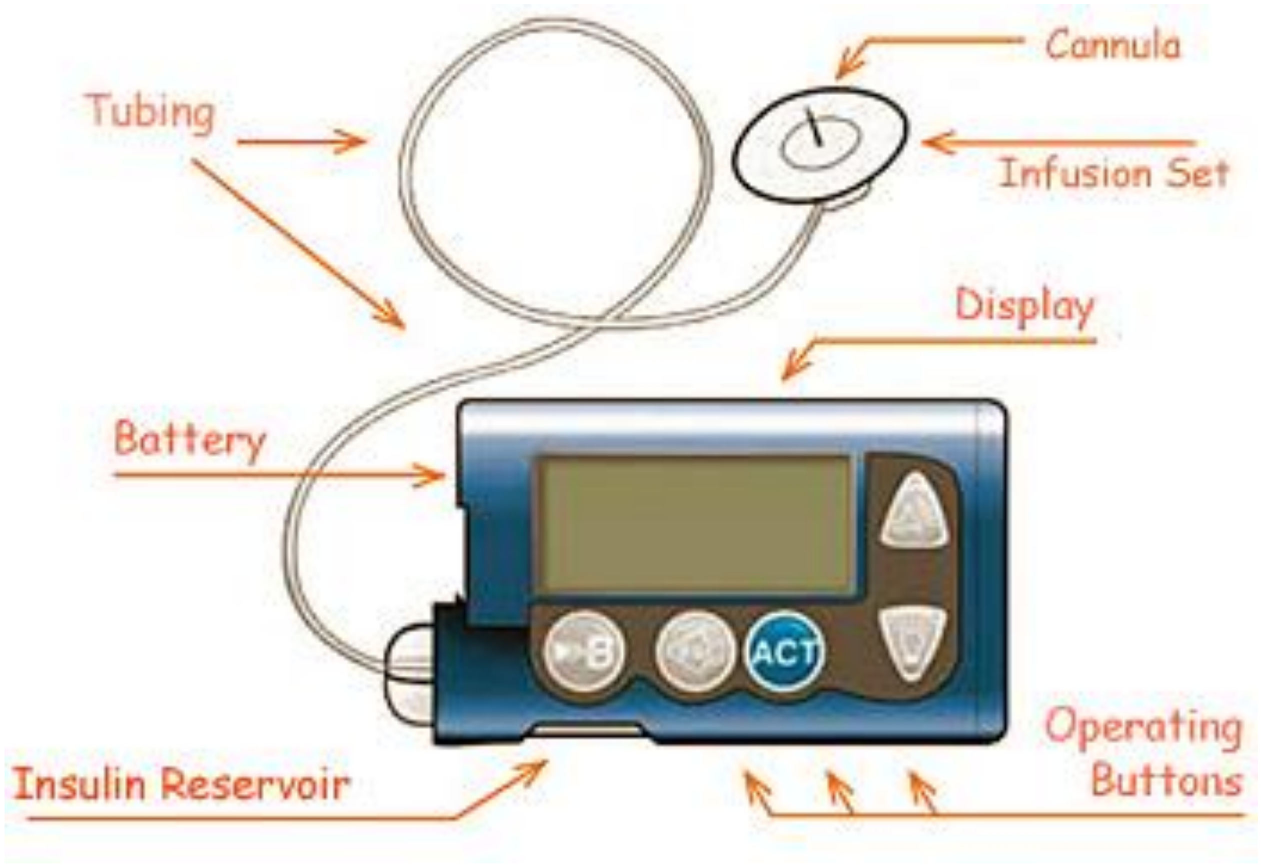
Case studies

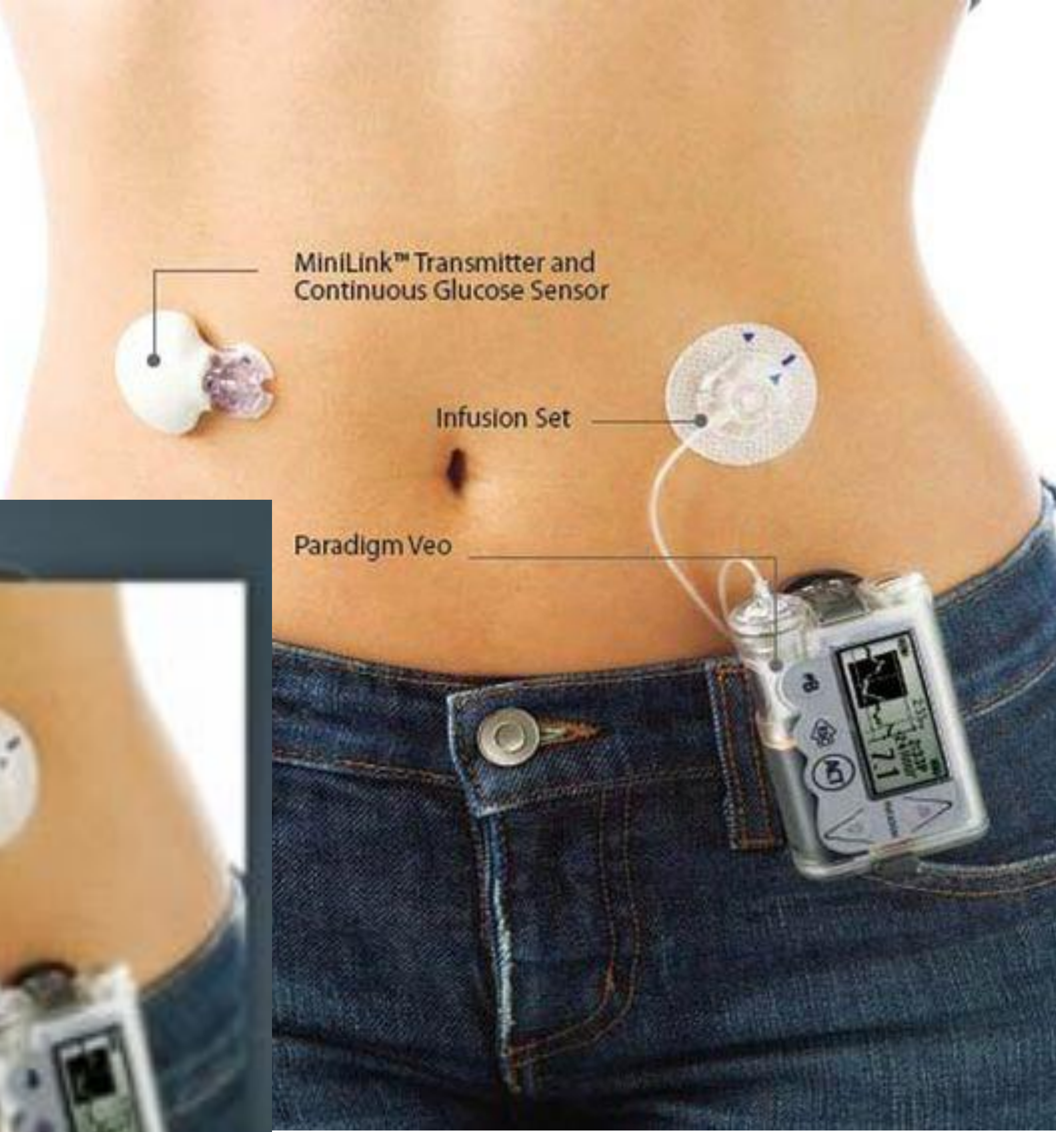
- ✧ A personal insulin pump
  - An embedded system in an insulin pump used by diabetics to maintain blood glucose control.
- ✧ A mental health case patient management system
  - **Mentcare**. A system used to keep records of people receiving care for mental health problems.
- ✧ A wilderness weather station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

1-Insulin pump control system

- ✧ **Collects data from** a blood sugar **sensor** and **calculates** the **amount of insulin required** to be injected.
- ✧ Calculations are **based on the rate of change** of ***blood sugar levels.***
- ✧ **Sends signals to a micro-pump** to deliver the correct dose of insulin.
  - ▪ **low blood sugars** can lead to **brain malfunctioning, coma and death**;
  - ▪ **high-blood sugar levels have** long-term consequences **such as eye and kidney damage**.

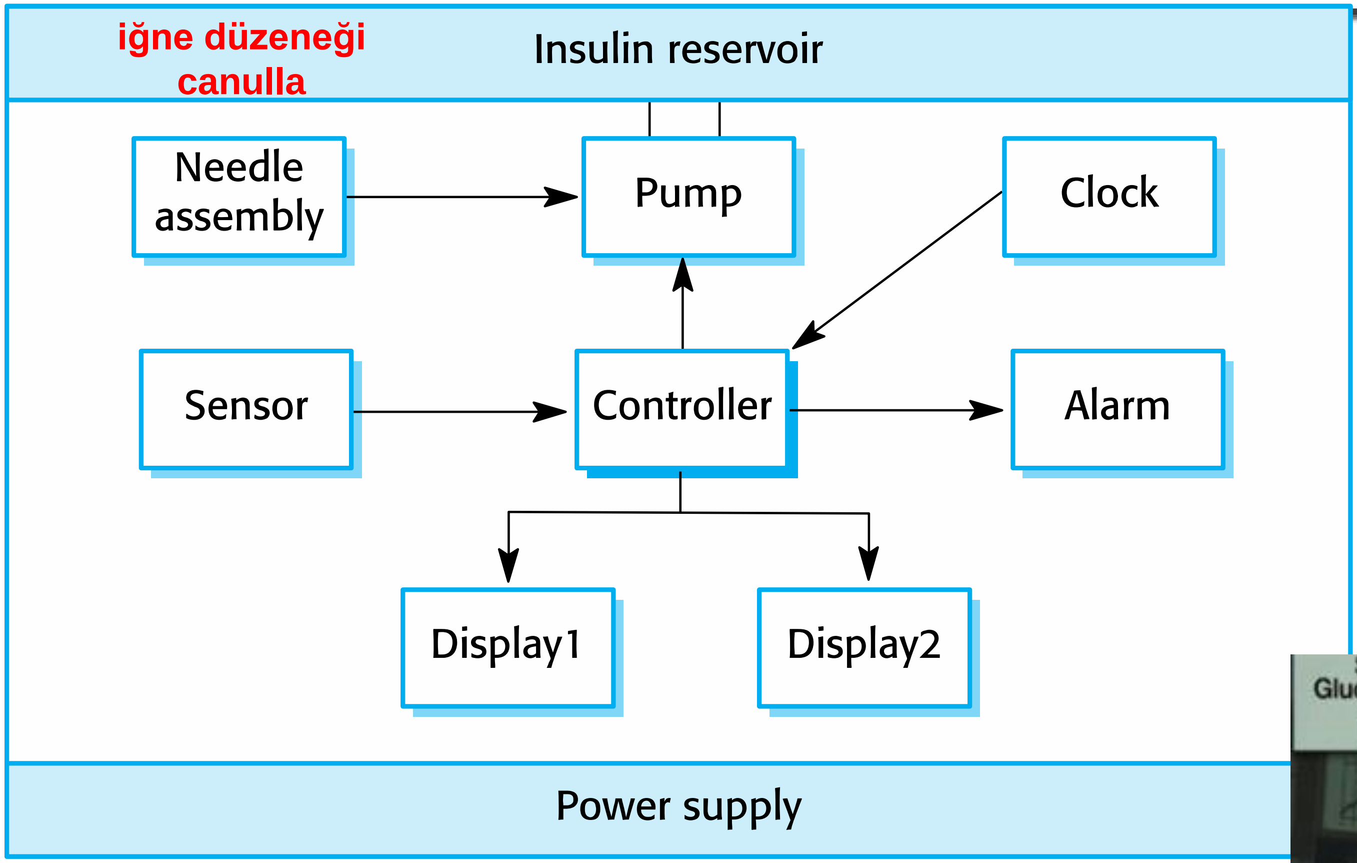
Software Engineering
GLOBAL EDITION
PEARSON


Tubing
Cannula
Infusion Set
Display
Battery
ACT
Operating Buttons
Insulin Reservoir

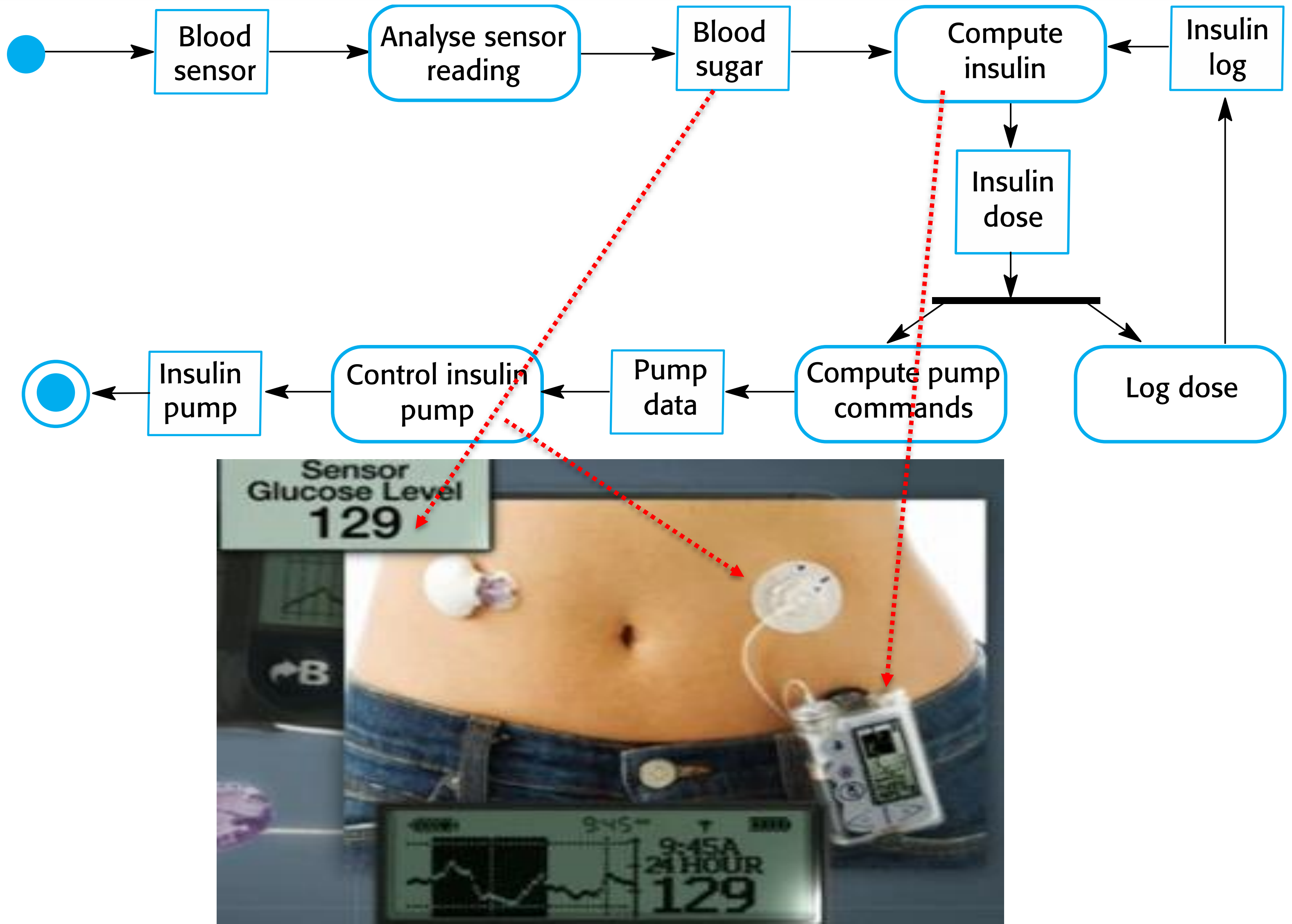

MiniLink™ Transmitter and Continuous Glucose Sensor
Infusion Set
Paradigm Veo

Sensor Glucose Level 129
9:45A
24 HOUR
129

Insulin pump hardware architecture



Activity diagram of the insulin pump



Essential high-level requirements
Customer Expectations/Functional Requirements:

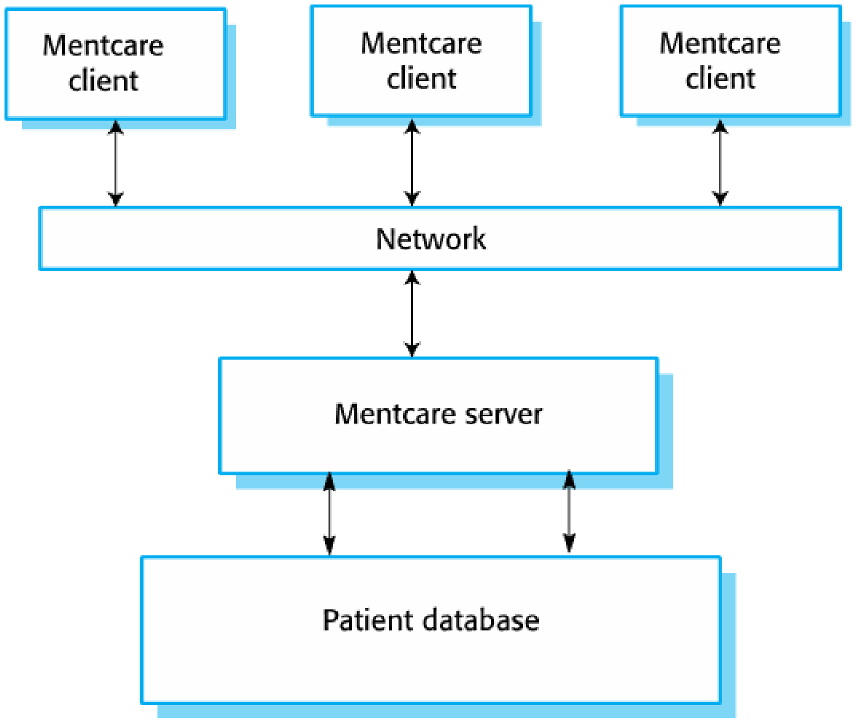
- ✧ The **system shall be available** to deliver insulin when insulin is required.
- ✧ The system shall perform **reliably and always deliver the CORRECT amount of insulin** to balance the current blood sugar level (*otherwise, irreversible undesirable consequences may occur*).
- ✧ The **system must therefore be designed and implemented** to ensure that the system always meets these requirements.

2-Mentcare: A patient information system for mental health care (a core SW prj)

- It is a **Medical Information System** To **Support Patients' Mental Health Care**
  - **Mentcare** that keeps information about the **patients suffering from mental health problems** and the **treatments** that they have received.
- Most of these patients **do not require a dedicated hospital treatment,** but,
  - they **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 also be held in **local medical practices or community centres** (toplum merkezleri,sağlık ocağı vb).

SO→A DISTRIBUTED SYSTEM INFRASTRUCTURE IS NECESSARY!

Mentcare



- Mentcare **<u>is an IS </u>**that is intended for use in clinics.

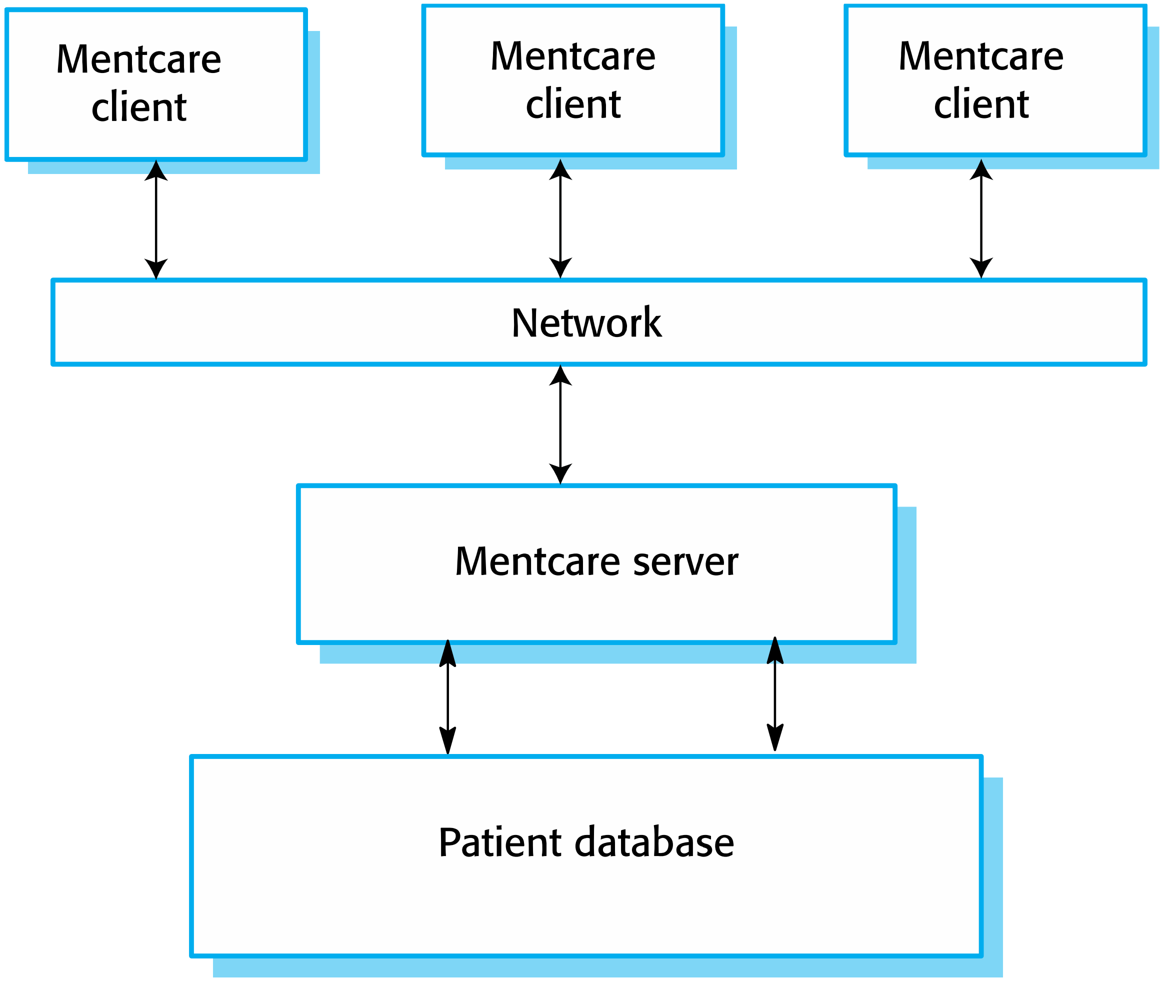
- It makes **<u>use of a centralized database of patient information</u>** but **<u>has also</u>** been designed to **<u>run on a PC</u>**, so that it may be accessed and used from sites that do not have secure network connectivity.

- **<u>When the local systems have secure network access</u>**, they use patient information in the database, **<u>but they can download and use local copies of patient records</u>** when they are disconnected.

Mentcare system 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



An example for client-server architectural model

Key features of the Mentcare system

- **Individual care management**
  - Clinicians can create records for patients, edit the patient information in the system, view patient history, etc.
  - The system supports/show 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 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.

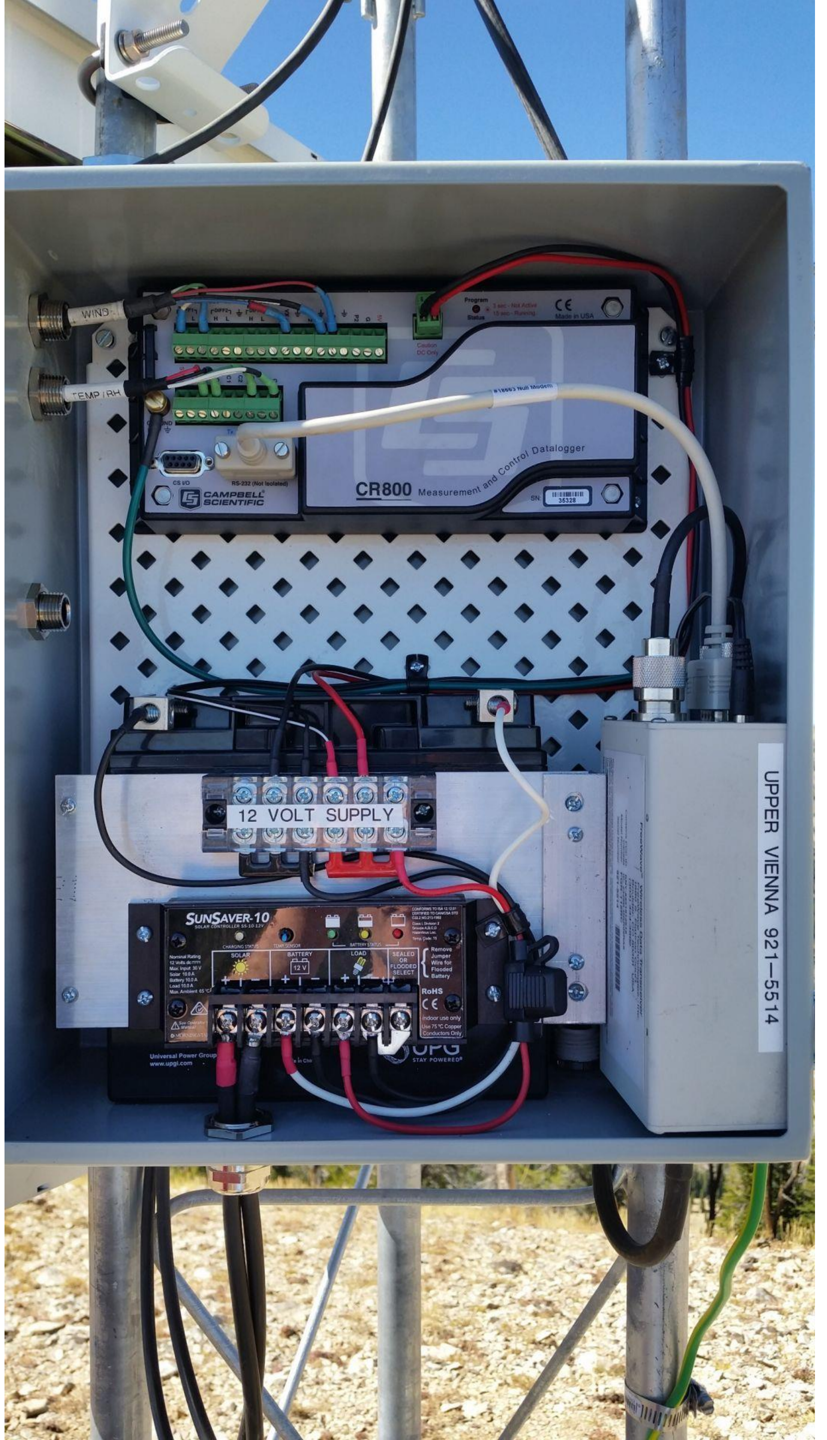
Mentcare system concerns

- **Privacy**
  - It is essential that **patient information is confidential** and
  - It is **never shared to anyone except authorised** medical staff and the **patient themselves**.
- **Safety**
  - Some **mental illnesses cause patients to become suicidal** or a danger to other people.
  - Wherever possible, **the system should warn medical staff** about potentially suicidal or dangerous patients.
  - The **system must be available when needed**
    - Otherwise, ***safety may be compromised/under risk*** and
    - it may be ***impossible to give the correct medication*** to patients.

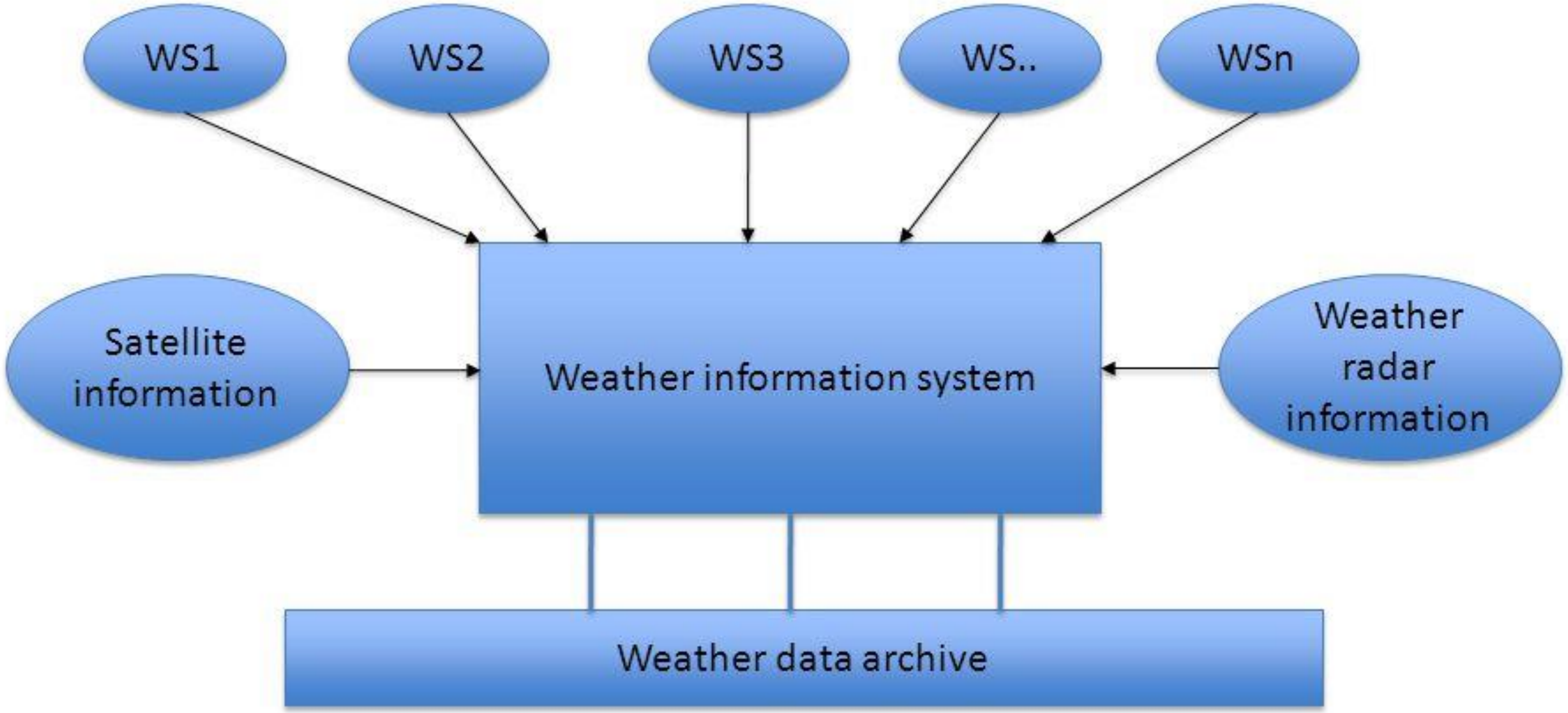
3-Wilderness weather station

- Some countries **have large areas of wilderness** *(geniş vahşi doğa alanları olan ülkeler…)*
- Their government **may decide to placed** **several hundred weather stations** in remote areas.
- Weather stations **collect data from a set of instruments** that measure *temperature* and *pressure*, *sunshine*, *rainfall*, *wind speed* and *wind direction*.
  - The weather station **includes a number of instruments (e.g. sensors)** that measure weather parameters such as the *wind speed and direction*, *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.

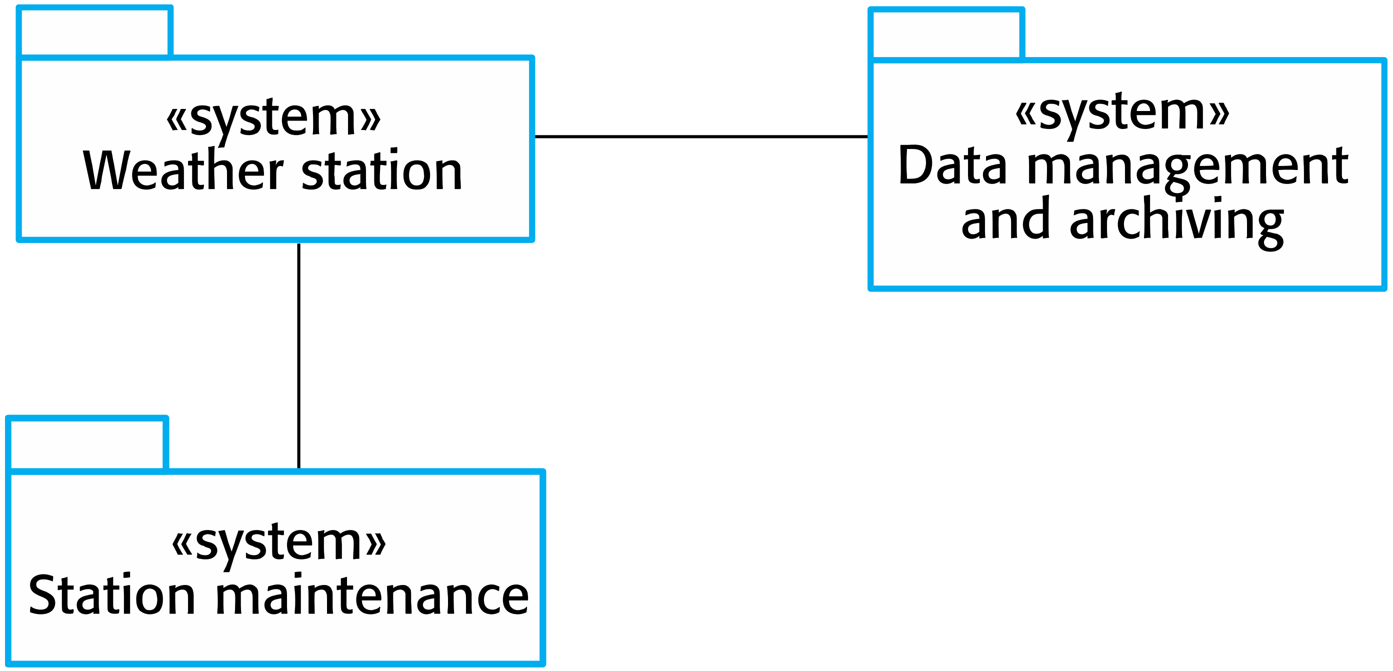
Measure *temperature* and *pressure*, *sunshine*, *rainfall*, *wind speed* and *wind direction*



Overall system organization



The weather station's environment



Bakım Sistemi

<u>It can communicate</u> with all weather stations **<u>via satellite</u>**

Weather information system

- The weather station system
  1. This is **responsible for collecting weather data**,
  2. Carrying out some initial **data processing** and
  3. **Transmitting it to** the **Data Management System**.
- The Data Management and Archiving System
  1. This system **collects the data from all of the weather stations**,
  2. Perfoms data processing and analysis
  3. then **archives the data**.
- The Station Maintenance System
  - This system **can communicate** with all stations **by satellite** to monitor **the health of these systems** and **provide reports** of problems.

Additional software functionality

✧ Monitor the **instruments**, **power** and **communication hardware** and **report faults** to the management system.

✧ **Manage the system power**, ensuring that batteries are charged whenever the environmental conditions permit but **also that generators are shut down** in potentially damaging weather conditions, such as high wind.

✧ **Support dynamic reconfiguration** where parts of the **software are replaced with new versions** and where backup instruments **are switched into** the system in the event of system failure.

4- iLearn: A digital learning environment

- ✧ iLearn **is a digital learning environment**
- ✧ **It is a framework** which **may involve** a set of general-purpose and specially designed tools for learning
- ✧ The tools **may be embedded and a set of applications** that are presented to the learners using the system.
- ✧ The **tools included** in each version of the environment
- ✧ The **tools are chosen** by teachers and learners to meet their specific needs.
  - These can be general applications ***such as spreadsheets***, learning management applications such as a ***Virtual Learning Environment (VLE)*** to ***manage homework submission*** and ***assessment***, ***games*** and ***simulations***.

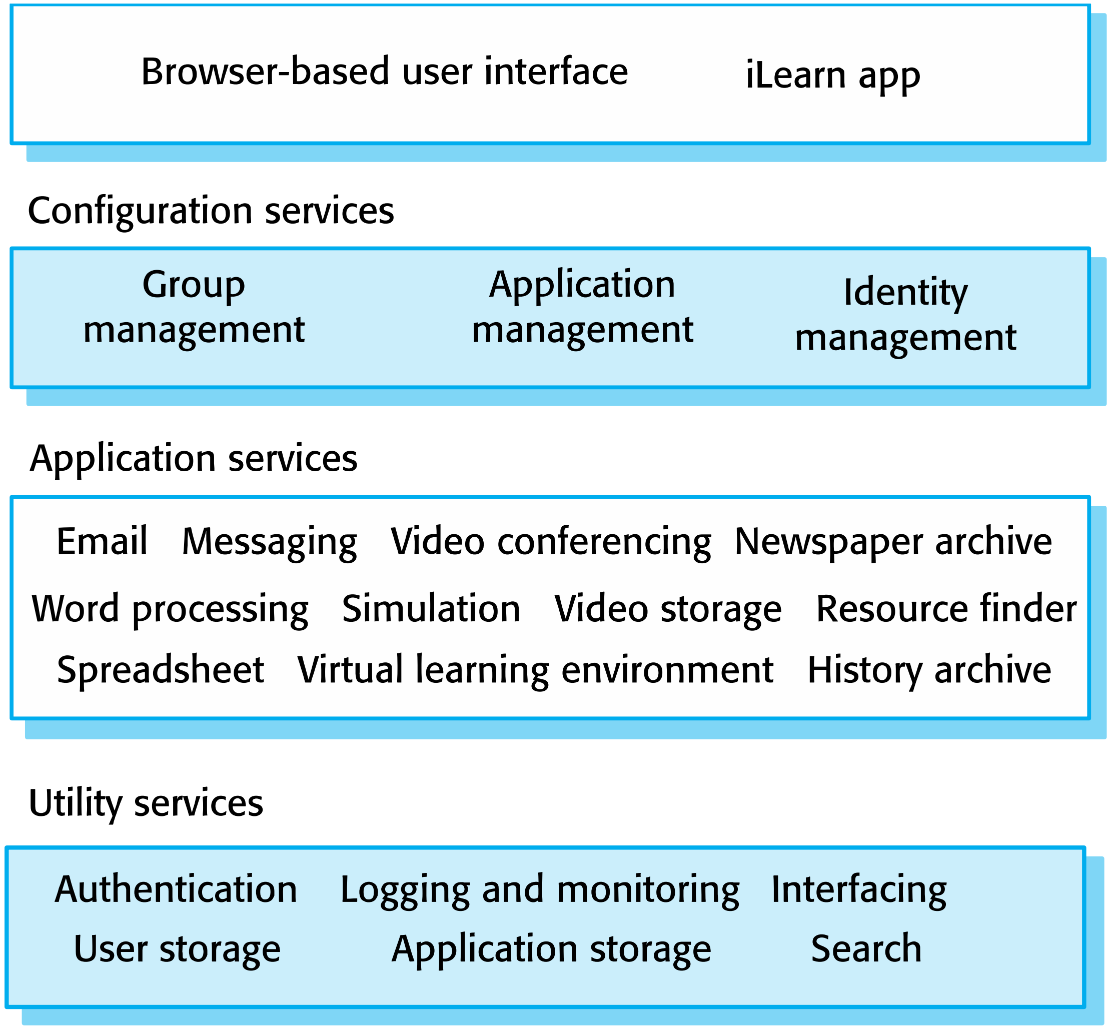
Service-oriented systems

- iLearn **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 graduate stds, undergraduate stds, instructors,* etc.

iLearn services

- ✧ ***Utility services*** that provide basic application-independent functionality and which may be used by other services in the system *(e.g. search, loging,).*
- ✧ ***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 *(e.g. email, simulation, video storage, spreatsheet, etc.).*
- ✧ ***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*. *(e.g. group management, applications management, identity management, etc.).*

iLearn layered base architecture



iLearn service integration

- ***Integrated services*** are services
  - which offer an API (application programming interface) 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 some 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 processes.
- ✧ The fundamental notions of software engineering are universally applicable to all types of system development.

Key points

- ✧ There are many different types of system, and each system requires appropriate SE tools and techniques for their development.
- ✧ The fundamental ideas of SE are applicable to all types of software system.
- ✧ Ses have responsibilities to the engineering profession and society. They should not simply be concerned with technical issues.
- ✧ Professional societies publish codes of conduct (davranış kuralları) which shows the standards of behaviour expected of their members.