ENS1161 Computer Fundamentals

Module 11
Embedded Systems, IoT and Cloud Computing

Moving forward..

- Last module:
  - Networking and the Internet
- Focus of this module:
  - Embedded systems, systems that use the Internet

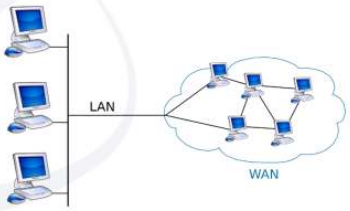


Image: Harald Mühlböck, 2011
https://commons.wikimedia.org/wiki/File:LAN_WAN_scheme.svg

Module Objectives

On completion of this module, students should be able to:

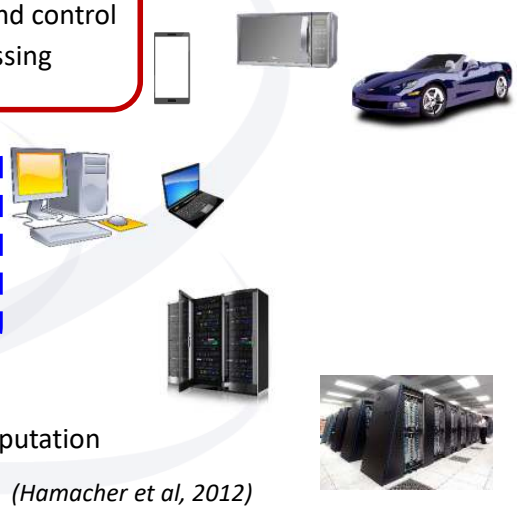
- Describe what embedded systems are, their key design considerations and principles of operation.
- List the common types and applications of embedded systems.
- Describe what IoT is, its impact and usage considerations, and explain how embedded systems form a crucial element of IoT.
- Describe the evolution of cloud computing, list the levels of cloud computing services and explain their principles of operation.

Introduction

- **Module Scope**
  - Embedded systems
  - Internet of Things
  - Cloud Computing

Categories of computers *(recap Module 1)*

- **Embedded computers**
  - Integrated into a larger device or system to monitor and control
  - Used for a specific purpose rather than general processing
    - E.g. in appliances, phones, cars, etc.
- **Personal computers**
  - General purpose computers for a single user
    - E.g. Desktop, laptop, notebook computers
- **Servers and Enterprise Systems**
  - Larger computers meant to be shared by many users
    - E.g. application servers, database servers
- **Supercomputers and Grid Computers**
  - Highest performance, used for highly demanding computation
    - E.g. weather forecasting, complex design simulations



(Hamacher et al, 2012)

General purpose vs embedded systems *(recap Module 1)*

- **General purpose computers**
  - Designed to be used for a variety of applications
    - E.g. word processing, presentation, simulation, entertainment
  - Based on general purpose processor chips
  - 'Normal' computer
- **Embedded systems**
  - **Integrated into a larger device or system to perform a specific function**
  - **Generally use single chip integrated microcontroller devices**
  - Used to provide the 'smarts' in a range of appliances and products
    - E.g. microwaves, air-conditioners, TVs, cars, phones, routers
  - General purpose computers use these special purpose processors for subsystems
    - E.g. graphics coprocessor, DMA controller, keyboard controller
- Both have same general architecture and principles of operation

Embedded Systems

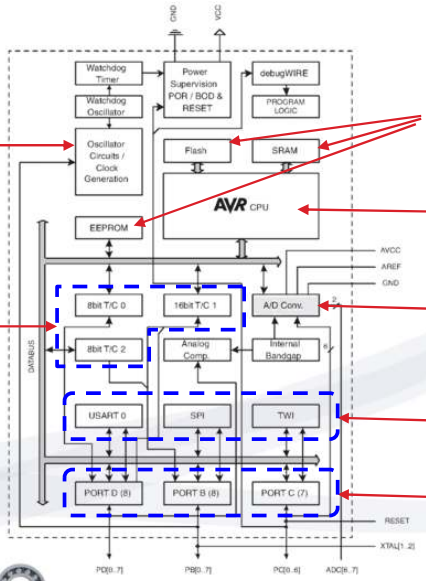
- Embedded systems employ computer control for a specific purpose
  - E.g. microwaves, medical monitoring systems, video game controllers, industrial control
  - As opposed to general purpose computers
- Fundamental components / techniques the same
  - differences due to different design constraints
    - E.g. reliability, robustness
- Embedded systems software interacts closely with hardware
  - Interfacing, polling, interrupts, communication
  - Software operation based on inputs
  - Often designed as State Machine

Embedded Systems Requirements

- Embedded systems have different requirements to general purpose computers
  - Reduced system footprint (chip count and size)
  - Low power
  - Resistance to electromagnetic noise
  - Robust – heat, vibration, shock, etc
  - Generally do not need very high processing power
  - In some cases, real-time response
- Use microcontroller chips with integrated memory and I/O interface circuits or System on a Chip (SoC)

Typical microcontroller structure

ATmega168 Block Diagram



Oscillator / clock circuitry

Memory

Main processor

3 timer / counters

Analog-to-Digital Converter

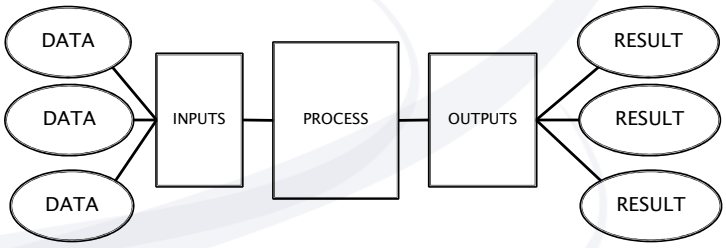
3 different serial interfaces

3 parallel / bit-wise I/O ports

System Behaviour

- **Transformational systems:**
  - have all inputs ready when invoked
  - the outputs are produced after a certain computation period

1. *A data input is taken and it returns an output.*
2. *The order of data inputs is preset.*
3. *Its execution must end.*

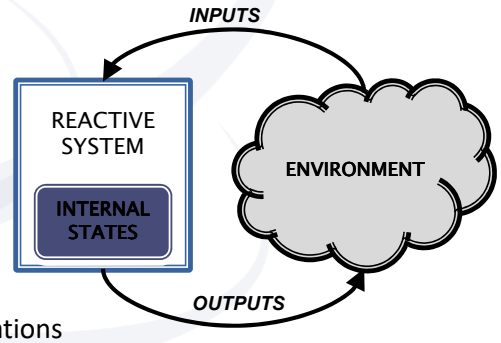


- Typical of data processing environments / applications

System Behaviour

- **Reactive Systems :**
  - inputs arrive in discontinuous and often unpredictable sequences
  - The system exists to interact with some entity or entities in its environment

1. *System continuously interacts with the environment, receiving stimuli and producing outputs in response.*
2. *Order of events in the system is not predictable, determined by external events.*
3. *The execution of reactive systems does not have to end*



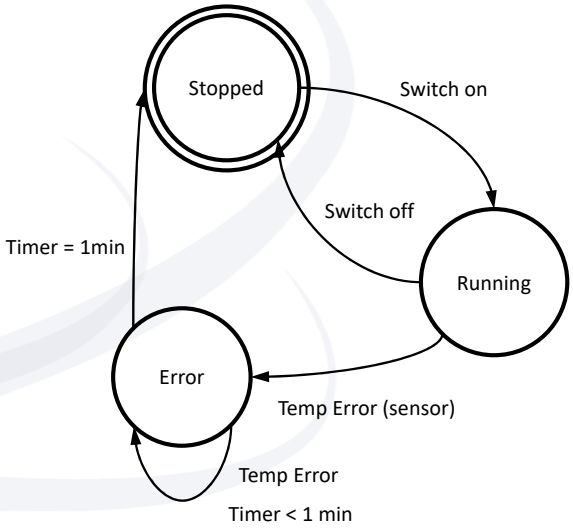
  - Typical of embedded systems environments / applications
  - Commonly designed as **finite state machines**

State Machines

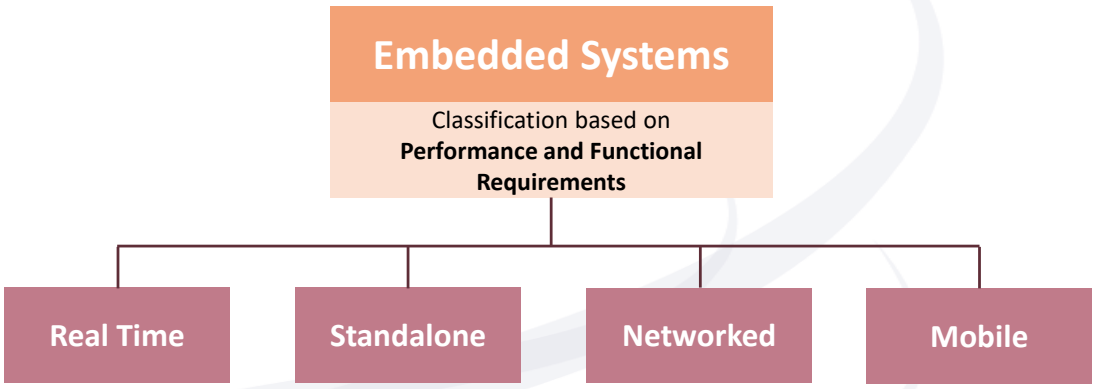
- **System States**
  - A system can only exist in a defined state when:
    - It is waiting for something in the external environment to occur, or
    - It is waiting for a current activity in the environment to change
  - A state must represent some behaviour in the system that is observable and that lasts for some finite period of time
  - Fixed number of states
    - *Finite* state machines
    - Has an initial (or reset) state

State Machines

- Transitions
  - A valid state change is called a transition
  - A transition connects relevant pairs of states
- Fixed number of events
  - Each Event *can* have associated Actions
  - Each Event *can* cause a Transition
- State Diagrams
  - Visual representation
  - A *directed graph*
    - States as nodes
    - Transitions as edges

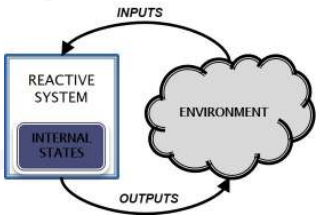


Types of embedded systems

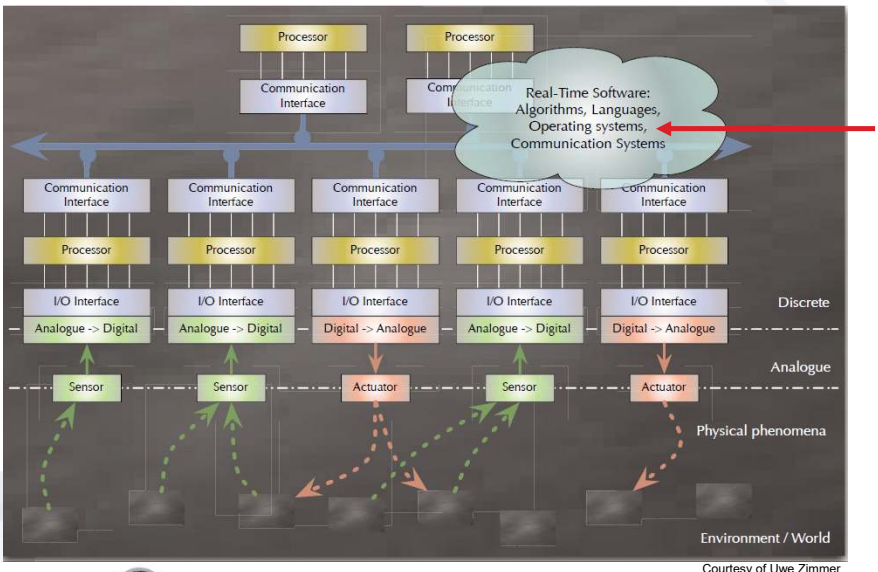


Real-time response

- Embedded systems react to input from the environment
  - E.g. from *sensors*
- These inputs (*events*) may cause changes in the state and outputs of the system (*system response*)
- The outputs often may control parts of the environment
  - Via *actuators*
- In many applications these responses need to happen within a specified time
  - Otherwise system 'fails'
    - E.g. aircraft control system, power grid monitoring, pacemakers
- These systems are called *real-time embedded systems*



Real-Time Embedded System



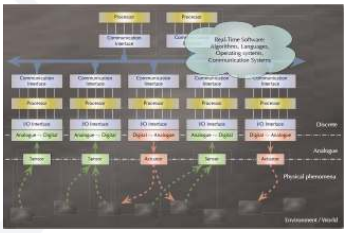
Courtesy of Uwe Zimmer

Real-Time Operating System (RTOS)

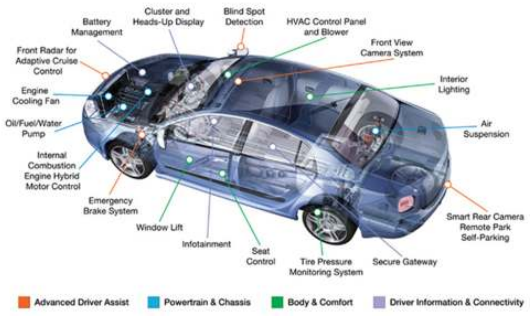
- An RTOS is specially designed to run applications with very precise timing and a high degree of reliability
  - Especially important in control and automation systems
    - Downtime is costly
    - A program delay could cause a safety hazard
- To be considered *"real-time",* OS must have a known maximum time for each of the operations that it performs
  - or at least be able to guarantee that maximum most of the time
- Some of these operations include OS calls and interrupt handling

Communication channels

- Many types of local communication channels
  - To connect sensors and actuators to controllers
  - To interconnect embedded subsystems within a larger system
    - E.g. in embedded subsystems in an automobile
- Wired channels
  - E.g. RS-232 serial, I^2C, USB, Ethernet
- Wireless channels
  - Infrared, Bluetooth, WiFi
- Increasingly embedded systems communicate over *Internet*



Courtesy of Uwe Zimmer



Internet of Things (IoT) *(recap Module 1)*

- A system of computing devices connected together via the Internet
- Consists of sensors, actuators and various other systems
- Enables us to do various things such as:
  - Control appliances such as air-conditioners and ovens using our mobile phones
  - Monitor security systems remotely
  - 'Smart' buildings can adjust lighting, heating and cooling based on internal conditions
  - Find the best route based on current traffic conditions

Some Impacts of IoT

- Enabled easier machine-to-machine communication
- Increased efficiency through faster response to environment
  - E.g. smart buildings, industrial control
- Need for lot more IP addresses
  - Important factor in development of IPv6
- Privacy / Security concerns
  - Data from devices can be used to track movement and other behaviour
  - Most devices are lack proper security
    - E.g. cases of cameras on devices remotely hacked
- Huge amounts of data generated

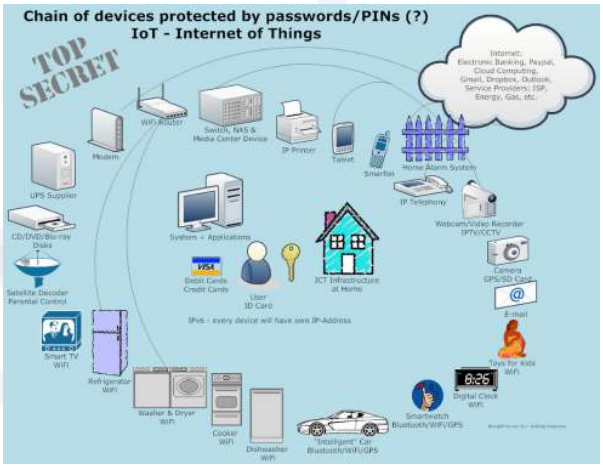


Image: Andrzej Kasprzyk, 2015 https://commons.wikimedia.org/wiki/File:Chain_of_home_devices_(including_IoT)_with_passwords_or_pin.png

Using data from IoT

- IoT and the data generated from it can be used in many useful ways
  - Regarding current situation of environment
    - E.g. traffic monitoring and route prediction, disaster detection and warning
  - For prediction / forecasting
    - E.g. weather forecasting, predictive maintenance
- To make full use of it requires analysis of data
  - Raw data generally needs to be *aggregated*
    - Summarised for analysis
  - Analysis then carried out by applications designed for that purpose
    - Generally done on servers with sufficient computing power (and storage)

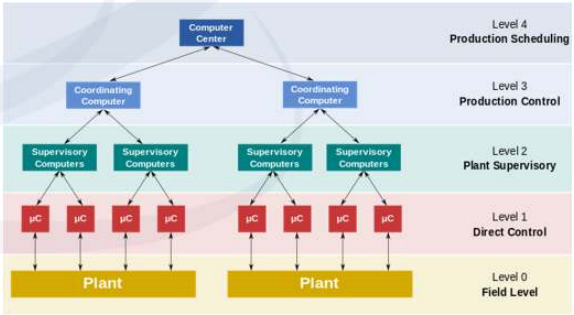


Image: https://en.wikipedia.org/wiki/Operational_historian

Computer networks and devices *(recap Module 10)*

- Server
  - Device or application that provides a service
    - Some form of data or shared resource access
    - 'Serves' client requests
- Client
  - Device or application that uses a service

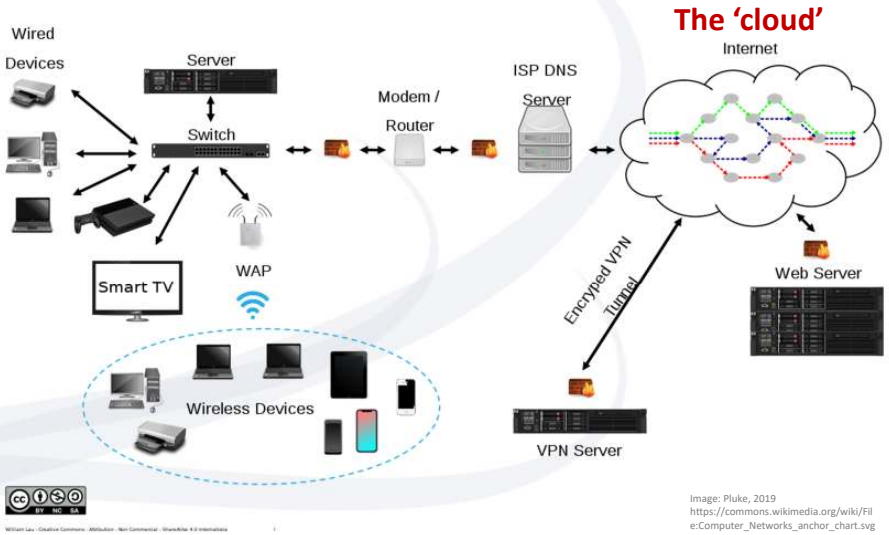


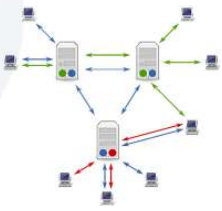
Image: Pluke, 2019 https://commons.wikimedia.org/wiki/File:Computer_Networks_anchor_chart.svg

Factors that affect server performance

- The main factors that limit server performance are:
  - *Processing power*
    - Particularly for computing intensive applications like analysis or video processing
    - Or when servers running many processes simultaneously (heavily loaded)
  - *Memory*
    - Amount of primary memory can be a bottleneck
    - Reading from / swapping out to disk slows down processing
  - *Internet bandwidth*
    - Congestion can severely limit efficiency of server
    - Particularly if high volumes of data being served out
      - E.g. streaming services

Overcoming limitations

- Work to be done distributed to a number of servers / nodes to overcome computing and memory constraints
  - These may be physical or virtual servers
- Having distributed servers
  - Helps distribute network traffic
  - Sometimes in geographically separate locations
    - Useful for content distribution
      - E.g. streaming services, download sites
- Use data centres that have high capacity Internet connections
  - Also provide other services / capabilities



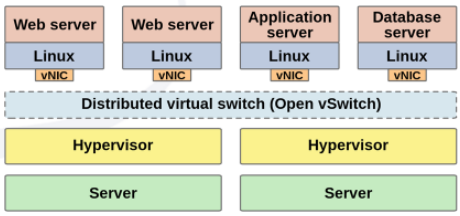


Image: Goran tek-en, 2014
https://commons.wikimedia.org/wiki/File:Distributed_Open_vSwitch_instance.svg

Virtual Machines (VM) *(recap Module 9)*

- VM software creates multiple virtual machines on the one physical machine
  - Examples of VM software are VirtualBox and VMware
- Each VM can have its own OS as well as virtual hardware
- A *hypervisor* (also known as *virtual machine monitor- VMM*) ensures that each VM has access to the resources that it requires
- VMs are often used for:
  - Trying new OS
  - Testing apps in different environments
  - Running old apps that need specific settings or environments

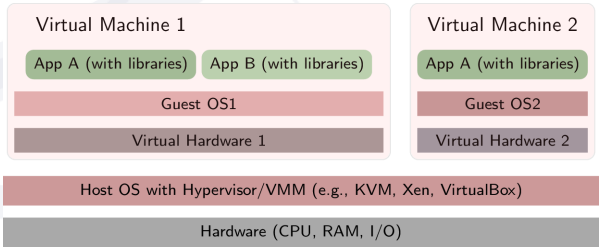


Image: Dr. Jens Lechtenbörger, DBIS Group, 2018
https://oer.gitlab.io/oer-on-oer-infrastructure/Docker.html#/sec-title-slide

Data Centre

- A data centre is a physical facility (space / building) that is used to house critical computing equipment and data.
- Data centres usually have the following facilities:
  - Space for the servers, storage units, routers, etc.
  - High-speed network and Internet connections
  - Stable (regulated) power supply with backup power supplies
  - Temperature controlled environment
    - Air-conditioning, computer cooling systems if required
  - Physical security for the equipment
  - Virtual security for data
  - Other safety / backup measures
    - Fire retardation systems, backup equipment, etc

Image: Earthranger, 2011
https://commons.wikimedia.org/wiki/File:Data_centre.jpg

Servers in data centres

- Servers in data centres are usually mounted in racks
  - Do not have individual keyboards and screen
  - Virtual or shared keyboard, screen, mouse

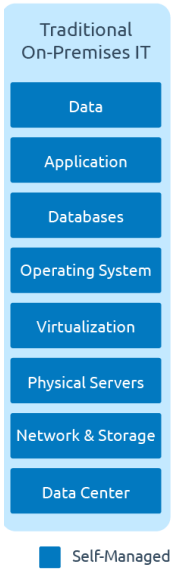
Image: Amber Case, 2010
https://www.flickr.com/photos
/caseorganic/4529327351

- Blade servers
  - modular single-board server that that contains just processor(s), main memory, network controllers and I/O interfaces
  - Some also have HDD/SDD secondary storage
  - Connected using normal Ethernet connections or other high-speed interconnect
  - Many are designed to be *hot-swappable*
    - Can be removed / put in while whole system is running

Image: Michael248, 2007
https://www.flickr.com/photos
/do-it_do-it-now/534918098

Traditional IT Infrastructure

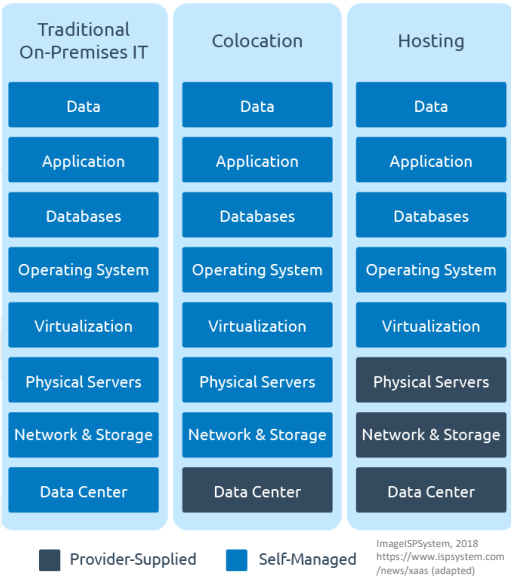
- Traditionally, large organisations provided all the infrastructure to deploy an application to users
  - The data centre
    - Includes cost of setting up centre + ongoing costs (staff, utilities, etc)
  - Network connections and storage
  - Physical servers
  - Virtualisation & OS
  - Databases and other resources required by application
  - The application itself
  - The actual data
- Very expensive to setup and to maintain



ImageISPSystem, 2018
https://www.ispsystem.com
/news/xaas (adapted)

Colocation and hosting

- Two methods for reducing this cost
- **Colocation**
  - Organisation sets up server and the other components
  - Server is located at data centre run by a 3rd party
    - Organisation is charged a fee for providing the data centre services
- **Hosting**
  - 3rd party provider supplies the physical servers and all infrastructure below it
  - Normally also provides the other technologies to run the hosted app (e.g. web hosting)
  - Hosting 'package' is static – fixed for a time period
    - Need to plan ahead and have required resources in place



ImageISPSystem, 2018
https://www.ispsystem.com/news/xaas (adapted)

Cloud Computing

- ***On-demand*** delivery of computing resources
  - Over the Internet
  - Pay-as-you-use model
  - Similar to traditional utility providers
    - electricity, water, etc.
- 3 parts
  - Software as a Service (SaaS)
  - Platform as a Service (Paas)
  - Infrastructure as a Service (Iaas)

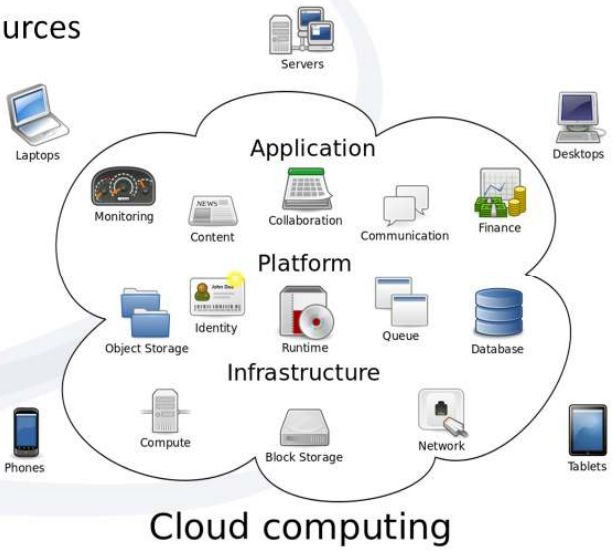


Image: Sam Johnston, 2009
https://en.wikipedia.org/wiki/File:Cloud_computing.svg

Infrastructure as a Service (IaaS)

- Virtualised computing resources over the net
  - Servers, storage, networking, other data centre services
    - E.g. Amazon Web Services (AWS) EC2, Rackspace.com
  - Service providers have sufficient resources to share between their clients on demand
    - Cost of setting up and maintaining infrastructure borne by the service provider
    - User of service can request resources as per their needs
    - Users pay based on what they use
  - Enables infrastructure to be scaled according to demand

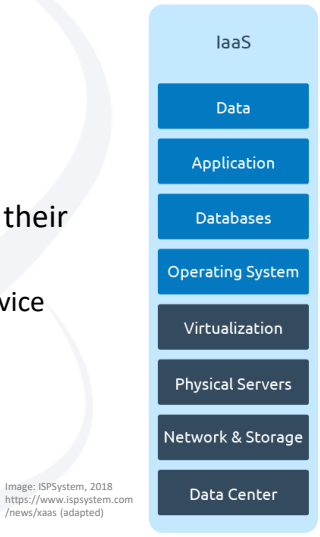


Image: ISPSystem, 2018 https://www.ispsystem.com/news/xaas (adapted)

Provider-Supplied Self-Managed

Platform as a Service (PaaS)

- Tools and software for to develop apps
  - Middleware, database management, OS
    - E.g. Microsoft Azure, Google App Engine
  - Users get to use an integrated development environment
    - Can store app on application hosting that provides all the necessary services
    - Provider takes care of upgrades of tools, patching and other routine tasks to keep system working optimally
    - User of service can request resources as per their needs
    - Also based on pay-per use model
  - Sits on top of IaaS layer

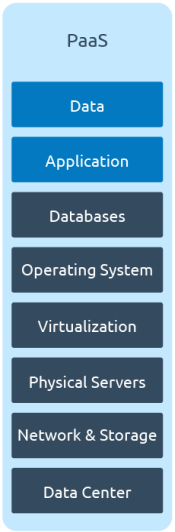


Image: ISPSystem, 2018 https://www.ispsystem.com/news/xaas (adapted)

Provider-Supplied Self-Managed

Software as a Service (SaaS)

- Delivery of applications-as-a-service
  - Accessed via a browser or app
    - E.g. Office365, Google Suite, Salesforce
  - Users get to use an application easily
    - Users do not need to install application on a local systems
      - Significantly reduces deployment and upgrade issues
    - Users not tied to a particular device
    - Upgrades and patches are done by service provider automatically
    - Scalable – no of users of an application can be changed easily
    - Various pay-per-use / subscription models
    - **Cons:**
      - May not be able to use software without Internet connection
      - Data privacy concerns

Image: ISPSystem, 2018
https://www.ispsystem.com
/news/xaas (adapted)

XaaS (Anything as a Service)

- 3 main components
  - SaaS (Software as a Service)
  - PaaS (Platform as a Service)
  - IaaS (Infrastructure as a Service)
- Other components of XaaS
  - Database as a Service (DBaaS)
  - Storage as a Service
  - Desktop as a Service (DaaS)
  - Communications as a Service (CaaS)
  - Disaster Recovery as a Service (DRaaS)
  - Malware as a Service (MaaS)
    - Helps protect against attacks such as ransomware and distributed denial of service (DDoS)

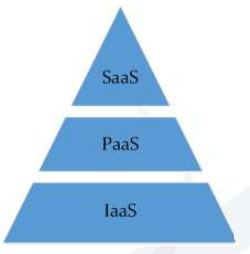


Image: BuRnZ, 2014
https://commons.wikimedia.org/wiki/File:XaaS.png

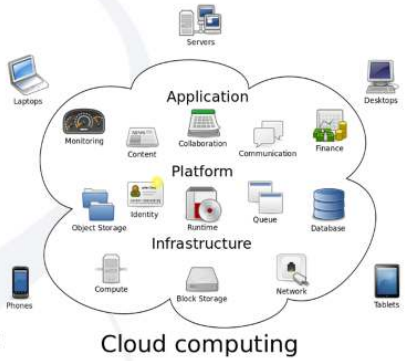


Image: Sam Johnston, 2009
https://en.wikipedia.org/wiki/File:Cloud_computing.svg

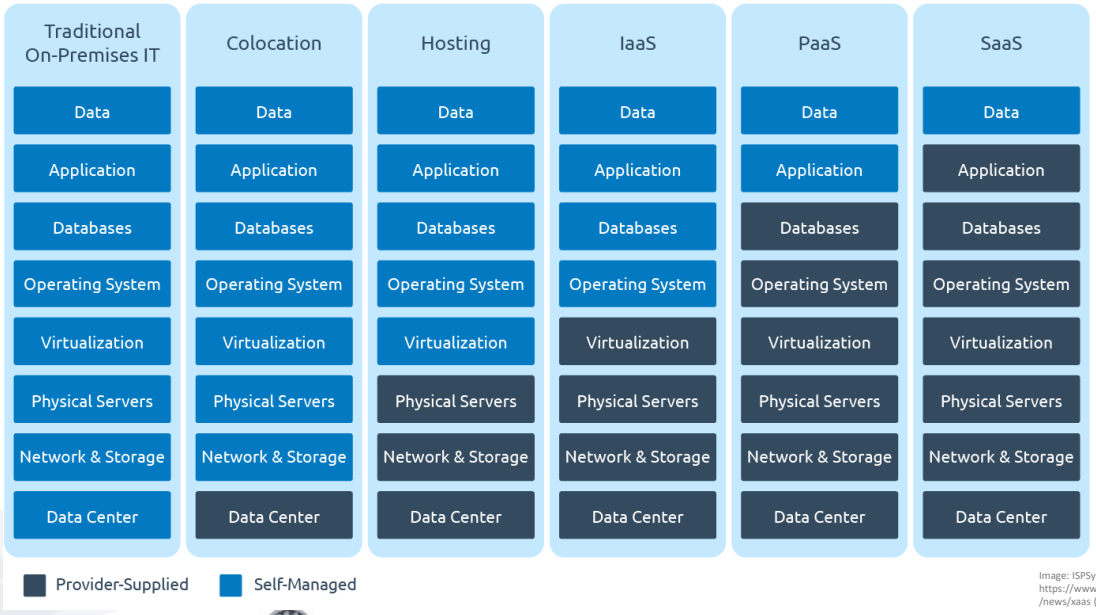


Image: ISPSystem, 2018
https://www.ispsystem.com
/news/xaas (adapted)

Pros and Cons of Cloud Computing

Image: Li, Shi, Shin and Qu, 2018
https://u.osu.edu/cloudcomputing/pros-and-cons/

Summary

- Topics covered:
  - Embedded systems
    - Processor characteristics, system behaviour, state machines
    - Transactional vs Reactive systems
    - Types of systems
      - Real-time embedded systems and RTOS
  - Internet of Things
  - Trends in systems that use the Internet
    - Data centres, colocation, hosting
    - Cloud computing
      - IaaS, PaaS, Saas, XaaS