

269101

Introduction to Information Systems and Network Engineering

Dr. Kenneth Cosh

# Table of Contents

Syllabus.....	1
Pretest.....	4
Chapter 1 – Fundamental Concepts, Introduction to Information Systems.....	5
1.0 What are Information Systems?	5
1.1 What is Information?	5
1.2 What is a System?	6
1.3 What is Information Technology?	7
1.4 Information System Resources	8
1.5 The Importance of Information Systems	9
1.6 Media	10
1.7 Vocabulary	11
1.8 Key Questions	12
1.9 Assignment – Case Study	12
Chapter 2 – Fundamental Concepts, Digitisation.....	16
2.0 Introducing Binary	16
2.1 Introduction to Encoding	17
2.2 Encoding Objectives	18
2.3 Huffman Coding	18
2.4 Media	20
2.5 Vocabulary	20
2.6 Key Questions	21
Chapter 3 – Communication Networks.....	22
3.0 Quick Quiz	22
3.1 Network Simulation Exercise	22
3.2 Important Computing Laws	22
3.3 Assignment – “Metcalfe’s Law is Wrong”	22
3.4 The Evolution of Communication	23
3.5 Telecommunication Systems	25
3.6 Protocols	26
3.7 Cables	27
3.8 Network Topologies	27
3.9 Media	28
3.10 Vocabulary	29
3.11 Key Questions	31
Chapter 4 – Social Networks.....	32
4.0 Introducing Social Networks and Web 2.0	32
4.1 Information R/Evolution	32
4.2 The Machine is Us/ing Us	33
4.3 Assignment – Presentation	33
4.4 History of Social Networks	33
4.5 Going Viral	34

4.6 Vocabulary	34
4.7 Key Questions	35
Chapter 5 – Hardware.....	36
5.0 Hardware	36
5.1 Evolution of Input Devices	36
5.2 Evolution of Output Devices	37
5.3 Interfaces and Interaction Styles	38
5.4 The System Unit	39
5.5 Media	40
5.6 Vocabulary	40
5.7 Key Questions	40
Chapter 6 – The Future of ICT.....	41
6.0 Assignment – The Movie Project	41
6.1 Changing Technology	41
6.2 Ubiquitous Computing	42
6.3 Media	43
6.4 Key Questions	44
Chapter 7 – Software.....	45
7.0 What is Software?	45
7.1 Application Software	45
7.2 System Software	45
7.3 Programming Languages	45
7.4 Problem Solving	46
7.5 Exercise	48
7.6 Key Questions	48
Chapter 8 – Software Development Lifecycle (SDLC).....	49
8.0 The Importance of Software	49
8.1 Introducing Software Engineering	49
8.2 The Waterfall Model	50
8.3 Evolutionary Development	51
8.4 Incremental Development	52
8.5 Reuse Based Development	52
8.6 The Requirements Process	53
8.7 Software Development	56
8.8 Key Questions	57
Chapter 9 – Data.....	58
9.0 Characteristics of Good Data	58
9.1 File Processing Systems	58
9.2 Database Management Systems	58
9.3 Microsoft Access – Quick Tutorial	59
9.4 Exercise	61
9.5 Key Questions	62
Chapter 10 – People.....	63
10.0 Project Blast Off	63

10.1 Stakeholders	63
10.2 Assignment – Belbin	64
10.3 Key Questions	65
Chapter 11 – Dependability.....	66
11.0 Dependability	66
11.1 Reliability	66
11.2 Security	67
11.3 Key Questions	68
Chapter 12 – Ethics.....	69
12.0 Ethical Questions	69
12.1 Business Ethics Theory	69
12.2 Media	69

# SYLLABUS – 269101

## Semester 1 / 2014

### INTRODUCTION TO INFORMATION SYSTEMS AND NETWORK ENGINEERING

Instructor Dr. Kenneth Cosh  
Office Dept of Comp Eng, 30<sup>th</sup> year anniversary bld. E-mail drkencosh@gmail.com  
Lecture Hours Mon/Thurs – 9:30-11:00

---

#### Texts:

Gary B. Shelly, Thomas J. Cashman and Misty E. Vermaat, Discovering computers 2011

#### Description:

Introduction to information systems for information systems and network engineering, Introduction to computer hardware for information systems and network engineering, Introduction to computer software for information systems and network engineering, Introduction to computer networks for information systems and network engineering, Introduction to database systems for information systems and network engineering.

#### Evaluation:

Midterm examination	25 %
Final examination	25 %
Assignments	40 %
Quizzes	10 %

#### Course Schedule:

Week	Topic	
	<b>Fundamental Concepts</b>	Assignment – A tale of two airlines
<b>1</b>	<ul style="list-style-type: none"><li>- What are Information Systems?</li><li>- What is Information?</li><li>- What is a System?</li><li>- What is Information Technology?</li><li>- Information System Resources</li></ul>	

- The Importance of Information Systems

### **Digitisation**

- 2 - Introducing Binary
- Introduction to Encoding
- Encoding Objectives
- Huffman Coding

Assignment – What are the effects of Digitisation on Quality, Ownership, Value, Form

### **Communication Networks**

- 3 / 4 - Important Computing Laws
- The Evolution of Communication
- Telecommunication Systems
- Protocols
- Cables
- Network Topologies

Assignment - “Metcalfe’s Law is Wrong”

### **Social Networks**

- 5 - Introducing Social Networks & Web 2.0
- Information R/Evolution
- The Machine is Us/ing Us
- History of Social Networks
- Going Viral

Assignment - Rethinking Copyright, Authorship, Identity, Ethics, Aesthetics, Rhetorics, Governance, Privacy, Commerce, Love, Family, Ourselves

### **Hardware**

- 6 - Hardware
- Evolution of Input Devices
- Evolution of Output Devices
- Interfaces and Interaction Styles
- The System Unit

### **The Future of ICT**

- 7 - Changing Technology
- Ubiquitous Computing

Assignment - The Movie Project

### **Software**

- 8 - What is Software?
- Application Software
- System Software
- Programming Languages

- Problem Solving

### **Software Development Lifecycle (SDLC)**

9 /  
10

- The importance of Software
- Introducing Software Engineering
- The Waterfall Model
- Evolutionary Development
- Incremental Development
- Reuse Based Development
- The Requirements Process
- Software Development

### **Data**

Assignment - Microsoft Access

11

- Characteristics of Good Data
- File Processing Systems
- Database Management Systems

### **People**

Assignment - Belbin's Team Roles

12

- Project Blast Off
- Stakeholders

### **Dependability**

13 /  
14

- Dependability
- Reliability
- Security

### **Ethics**

15

- Ethical Questions
- Business Ethics Theory

### **Assignments:**

Students must submit all assignments by midnight on the due date to avoid penalty of 5% per day. No submission will be accepted after 2 weeks. Extensions on assignments will be allowed for appropriate reasons.

### **Plagiarism:**

Plagiarism is academic dishonesty and totally unacceptable. Using other people's ideas or words requires acknowledgement. No work will be graded if failing to do so. To avoid plagiarism, the source/author must be given credit where required.

## Pretest

- 1) What is I.S.?
- 2) What is I.T.?
- 3) What is the difference between I.S. and I.T.?
- 4) What is Data?
- 5) What is Information?
- 6) What is a System?
- 7) What is Technology?



## Chapter 1

### Fundamental Concepts – Introduction to Information Systems

#### 1.0 What are Information Systems?

These two quotes point out the essence of information systems.

*“Interrelated components working together to collect, process, store and disseminate information to support decision making, co-ordination, control, analysis and visualization in an organization” (Laudon)*

*“An information system can be any organized combination of people, hardware, software, communication networks and data resources that collects, transforms and disseminates information in an organization.” (O’Brien)*

Firstly, information systems (and systems in general) are composed of ‘interrelated components’ – i.e. stuff – and these components work together to do something. O’Brien extends the definition by highlighting the 5 key components of an information system: **People, Hardware, Software, Communication Networks** and **Data**. We will examine these elements in more detail through this course.

Secondly, information systems involve input, processing and output – Laudon uses the words “collect”, “process” and “disseminate”, while O’Brien says “collects”, “transforms” and “disseminates”, but these map neatly onto “**Input**”, “**Process**” and “**Output**”. It is worth pointing out that Laudon also includes “store”, which reflects that data (or information) will need to be kept during the running of an information system.

Thirdly, information systems have a purpose, normally within some kind of organization. The purpose could be for supporting decision making, co-ordination, control, analysis or visualization.

Let’s break things down even further, and consider ‘Information’ and ‘Systems’ separately.

#### 1.1 What is Information?

Information is data that has been processed so that it becomes useful. Data is the raw symbols, which on its own is quite meaningless – for instance ‘30’. On its own it doesn’t mean anything, until we process it into information – for instance ‘there are 30 students in the class’.

Ackoff presents data and information as being just the first two rungs on a ladder (or pyramid) to wisdom. While **data** are the raw metrics, **information** can then be used to answer questions such as “What?” – What time does class start? This information can be further processed to create **knowledge**. Knowledge helps us to answer questions such as “How?” – How do information systems work? A further stage of processing can transform knowledge into **understanding**. Understanding offers us an appreciation of “Why?” – Why do certain events happen? This can be further processed, or evaluated to

create **wisdom**. Wisdom, through experience over a period of time, affords us the ability to project understandings into the future and predict outcomes and consequences before they occur.

This progression is often presented as a ladder, and there are alternative perspectives, for instance 'understanding' is often removed as a separate level, with the suggestion that understanding is the process by which data is transformed into wisdom.



It is worth pointing out that 'computers' are very good at dealing with data, and processing the data to provide information.

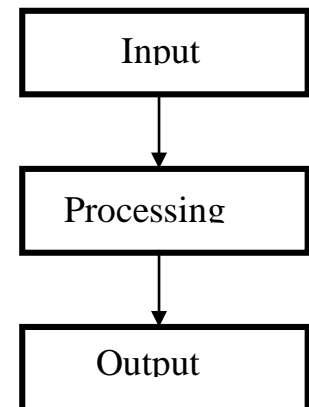
## 1.2 What is a System?

*"Groups of interrelated or interacting elements forming a unified whole." (O'Brien)*

There are many types of system, and the general definition is simply for when things work together. Examples of other systems include Physical Systems (such as weather systems, or the solar system), Biological Systems (such as the human body), Educational Systems (including schools and universities) and of course Information Systems.

These systems consist of 3 basic elements; **Input, Processing** and **Output**. Input are the things that enter into the system, for example foods may enter into a biological system like the human body, orbits and pressure systems are inputs to the solar system and weather systems. Processing is the transformation of inputs to outputs; this could be a chemical reaction, complex calculations, etc. Finally output is the result, where things leave the system moving on to other systems, for instance in the human body the food input is converted into energy and the output is movement.

*For an Information System, data is the input and information is the output.*

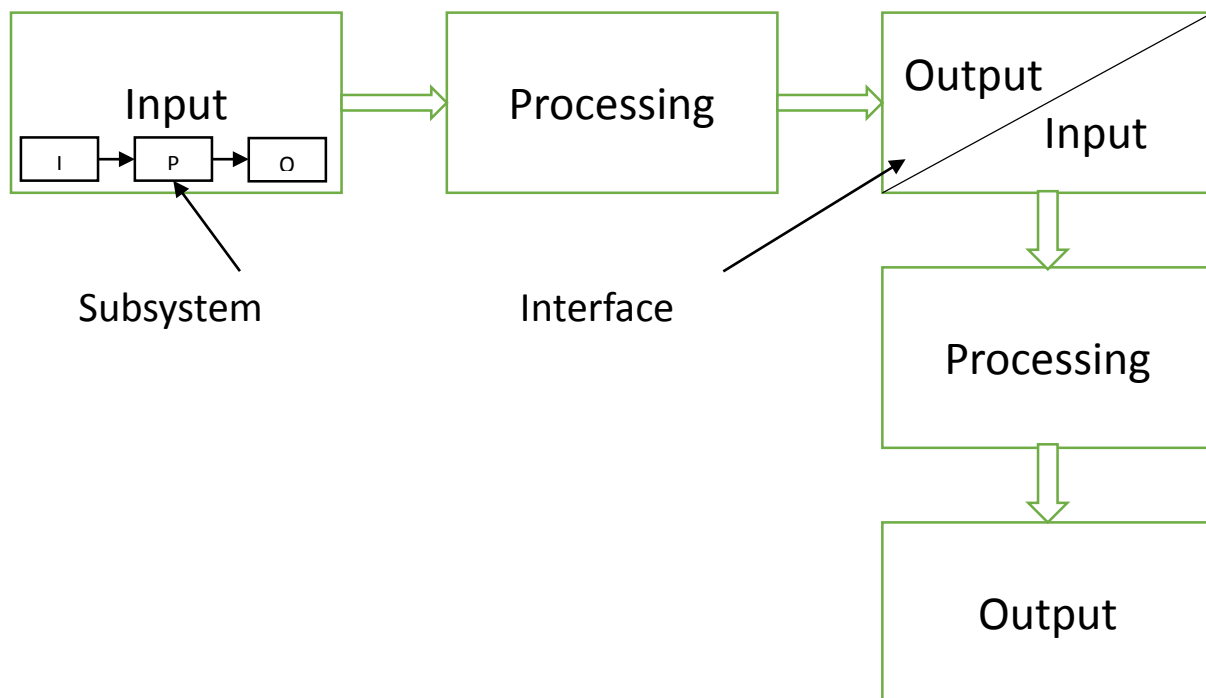


Systems become more useful if we add some 'managerial' functions, or a 'feedback and control' loop whereby data is collected about the performance of the system (feedback) and this feedback can

be monitored to determine if the system is moving appropriately towards its goal, with adjustments being made where necessary.

As an example, the thermostat on the air conditioner is a system, where the current room temperature is taken as an input via a sensor. Some processing occurs when it is compared with the desired temperature, and the output is either to increase the flow of cold air or not. In more complicated versions, calculations might be performed based on power saving settings. Finally there is the feedback and control portion whereby human users can adjust the temperature.

It is worth highlighting that systems do not exist in a vacuum, i.e. systems are generally part of an environment or greater system. When a system is part of a bigger system, it is known as a **sub-system**. As well as systems being part of a larger system, some systems are **Interfacing Systems** where the output of one system becomes the input to another system – for example the stock control figures are output from the store room to become inputs to the sales department. In these cases the systems interact with one another through a common, shared interface. **Adaptive Systems** are systems which have the ability to change itself, or its environment in order to survive or improve its performance.



### 1.3 What is Information Technology?

Often the term Information Technology (IT) is used instead of Information Systems (IS), and more recently Information and Communication Technology (ICT) is also used with some confusion. Technologies are the tools, machines and techniques that help to solve a problem, achieve a goal or perform some function. Information Technology are those tools which facilitate Information Systems, with Communication Systems being closely related and hence ICT.

Early technologies come from when humans first started writing (i.e. recording data in primitive information systems). Many technologies have been created to assist with information systems, including paper & pencils (hardware) and filing cabinets (data resource), although clearly the advances in computer based technology have had the biggest impact on Information Systems.

The computer (simply) consists of **Input Devices** (such as keyboards, mice and microphones), **Output Devices** (such as monitors, printers and speakers), a **Processor** (i.e. the CPU) and **Storage Capabilities** (both the main memory, or RAM, and secondary memory such as hard disks). Notice how closely this maps onto the “Input -> Process -> Output” model discussed for Information Systems. The storage capability is reflected in the original quote by Laudon.



## 1.4 Information System Resources

In this section we take a closer look at the five key resources in an information system, i.e. People, Hardware, Software, Data and Network (Communication).

### *People*

People are an important part of any information system, with most being ‘Socio-Technical Systems’ – i.e. involving Society and Technology. We could break people down into 2 key groups – the End Users, who use the system, and the IS Specialists, who design, develop, implement and maintain information systems.

### *Hardware*

Hardware are the physical / tangible devices and materials used in an information system. The important part being that you can touch it. We could break it down into 2 key groups – the Machines, such as PCs, monitors and printers, and the Media, such as printouts or disks.

### *Software*

Software is a set of processing instructions. Often software is broken down into System Software and Application Software – where the system software includes the operating system and important instructions to get the machine to work (including drivers etc.). The Application Software are the programs that we can use to perform the tasks we need in our information system, such as word or excel. Taking the definition of software as ‘a set of processing instructions’, there is a further set of software – as well as the instructions followed by the computer, there are instructions followed by people – i.e. the operating instructions such as data entry procedures. These are also an important part of an information system.

### *Data*

We have already discussed data, but in computer based systems data is stored either in the RAM (Random Access Memory) and/or in secondary memory. Data in primary memory is often stored as variables, although there are many exciting data structures to discover later. Data can also be stored in different ways on the hard disk, commonly in files or in a database.

### *Network*

The communication network is an increasingly important part of information systems as information needs to be shared. To facilitate the communication network we need communication media, such as cables, fiber optics, and wireless networks, as well as supporting hardware technology such as modems along with software technology such as internet browsers.

## **1.5 The Importance of Information Systems**

Over the past decades modern information systems have taken on a vital role for business and organisations. For instance;

- Supporting Business Processes & Operations
  - Assisting with automating routine processes
- Supporting Decision Making by employees and managers
  - By making the right information available to make the right decisions
- Supporting Strategies for Competitive Advantage
  - Today very often information systems are the cause or the driver of organization strategy

These three bullets represent how information systems perform a role at different levels – the day to day level, the tactical level and increasingly at the strategic level. At a day to day level, systems assist with making routine tasks automated helping workers perform their roles accurately and reliably. At the tactical level making better information available helps managers to make better decisions. At the strategic level, today information systems are driving corporate strategy, for example by opening up new markets or reducing costs through systems like Supply Chain Management systems (SCM).

While originally information systems simply provided automation functions, they have increasingly taken on a more significant (or strategic) role. This is due to both technology push and business pull effects – i.e. as new technologies are developed, they push their way into being used in organisations, meanwhile businesses are demanding more information systems and investing in their

development. The reasons for “Business Pull” effects couple be Political, Economic, Social or Technological (PEST).

New technologies have been developed allowing greater capacity, better processing and storage abilities and better connections, all of which with a comparative cost reduction. This is related to, or extrapolated from “**Moore’s Law**” as capabilities have improved while costs haven’t increased.

As Information Systems have taken on an increasingly important role in organisations, we can identify 4 key eras of technology:-

#### *Data Processing Era (DP)*

This era (in the 1950’s and 60’s) was concerned with automation – replacing routine manual tasks with computerized support. The objective being to decrease costs and maximize the use of the expensive equipment. The computer was generally a mainframe, locked away in the basement of an organization, under the control of the computing department. The biggest limitation on the system was the hardware, with batch processes often taking all night to run. The focus was to write more efficient programs.

#### *Management Information Systems Era (MIS)*

During the 1970’s and 1980’s the technology was noticed by managers who saw the advantage that could be gained from better information. As the comparative cost of PCs decreased below mainframes, processing became distributed through the company with each department owning its own machine. Rather than being controlled by the computing department each department took ownership with a focus on identifying the needs of different departments and users.

#### *Strategic Information Systems Era (SIS)*

As networking capabilities were developed the information from different departments could be combined and used by senior executives to help with strategic decisions. During the 1980’s and 90’s information systems and related technologies were transforming organisations with the only limitation being the vision of senior management.

#### *Internet Era*

The internet has further changed the use of information systems as networks spread outside of the organization. The reach and range of communications became global and information systems became the business with nearly everyone involved. The internet is still evolving with web 2.0 and social networks

### **1.6 Media**

This week’s videos can be found here:-

Did You Know? Shift Happens. We Live in Exponential Times.

<http://www.youtube.com/watch?v=pRVVZIGb7oc>

## 1.7 Vocabulary

One of the key objectives of this course is to develop English skills, including learning key vocabulary. The most effective way of learning vocabulary is using flash cards, and we will use the free to download Anki software, available from:- <http://ankisrs.net/>.

Each week I will give you a package of words for you to import. Spend a little time each day using the software and you will quickly learn the vocabulary. When each word appears, write it down in English, and then see if you can remember the Thai equivalent – then choose the appropriate interval before the words come back.

Here are the first set of words:-

- Interrelated - เกี่ยวพันกัน
- Era - ยุค
- Wisdom - เมธา
- Mainframe - เมนเฟรม
- Coordination - การประสาน
- Decision - การตัดสินใจ
- Vacuum - สูญญากาศ
- Database - ฐานข้อมูล
- Unified - ปึกแผ่น
- Interface - อินเตอร์เฟซ
- Appreciation - ความกตัญญู
- Vital - สำคัญยิ่ง
- Role - บทบาท
- Adaptive - ปรับได้
- Interaction - การมีปฏิสัมพันธ์
- Tangible - สัมผัสได้
- Combination - การรวมกัน
- Device - เครื่อง
- Evaluation - การประเมินผล
- Disseminate - เผยแพร่

## 1.8 Key Questions

- 1) What is an Information System?
- 2) What are the 5 key resources working in an information system?
- 3) What is the difference between Data and Information?
- 4) What is a sub-system?
- 5) What is an interface?
- 6) Discuss the 4 eras that information systems have been through.

## 1.9 Assignment – Case Study

Read the case study “A Tale of Two Airlines in the Network Age” (adapted from Corporate Information Strategy and Management, by Applegate, Austin and McFarlan, 2003) on the following 3 pages, and in groups answer the following questions;

For each of the three flights mentioned in the case study, consider what were the key resources involved in the information system.

- What hardware?
- What software?
- What people?
- What data?
- What communication network?
- Was the result a success or a failure - and why?



# Tale of Two Airlines in the Network Age

## Part 1

As Professor Roger McPherson's wait to go through the security process a second time dragged on into its third hour on this spring day in 2002 (all passengers had to be rescreened upon the discovery that one of the airport screening machines was unplugged), he was reminded of another delayed business trip and the role that information technology played in the story.

At 5:30 PM on February 15<sup>th</sup>, 1995, 200 feet off the ground, Professor Roger McPherson gazed anxiously through the fog as his airplane moved to touch down at Hartsfield Airport in Atlanta, more than 1 hour and 15 minutes late. He had 30 minutes to catch his 6:00 PM flight to London, where he would be meeting with the executive leadership of a major British power company to discuss their information strategy.

He felt fortunate, however, to be flying this carrier, which had a reputation for outstanding service. He was even more comfortable because he had a full-fare first class ticket and was a Gold Card member. Professor McPherson was always uneasy about the large premium charged for full, first-class tickets but knew that in a crunch it often meant the difference between making a connection and missing one. He well remembered a decade ago flying this airline from Milan to London to connect to a flight to New York. Bad weather then had also reduced his 1-hour-and-30-minute connect time to 10 minutes. A discussion of the problem with the first-class cabin attendant had resulted in a phone call from the pilot to London (the airline's hub city) and a car to whisk him and one other passenger to the New York flight, which took off only one minute late. That extraordinary service had made Professor McPherson a 10-year devotee of the airline.

NOTES :

---

---

---

---

---

---

# Tale of Two Airlines in the Network Age

## Part 2

In the information age, he knew it would be different, and he was secure. The airline flying to London would have identified him off their computer as a close-connecting passenger. It would have noted he checked no bags through, and it would be anxious to capture his \$2,500 fare – about 10 times that of the average passenger – on an only moderately loaded flight.

As his plane pulled into the gate at 5:40PM, he knew it would be tight but he would make it, particularly given the fact that all planes were coming in late. Moving his 57-year-old frame into a dim recollection of a high school 400-metre specialist, he set off. Two escalator rides and one train ride later, the gate hove into sight, and he braked to a halt at 5:53 PM. It was close, but he had done it.

Looking through the airport window, however, he was stunned to see the air bridge detached from the plane with splendid teutonic efficiency 7 minutes early. The door to the bridge was closed, no agent was in sight, and he was reduced to waving his bags through the window to the pilot 20 yards away (it had, after all, worked once in a similar situation on Continental Airlines). Alas, by 5:58 PM the plane was pushed back, and the agents emerged and quite cheerfully (and unregretfully they had no clue who he was) booked him onto another airline that would leave 1 hour and 45 minutes later. He would be 30 minutes late for his meeting in London, but the executives would understand. Distinctly irritated, he straggled off to the new airline's first-class lounge to begin a frantic series of phone calls and faxes to the United Kingdom. As he trudged through the airport, McPherson began to see the beginning of a lecture on service in the information economy and the fact that technology is only a small enabling piece of a total service concept.

NOTES :

---

---

---

---

---

---

---

---

# Tale of Two Airlines in the Network Age

## Part 3

At 7:50 PM, comfortably seated in the first-class cabin of his new carrier, McPherson jerked to attention as the captain came on to announce that because of a leak in the hydraulic system, there would be an aircraft change and a two-and-a-half-hour delay. Sprinting off the plane, McPherson realized that the meeting with the power company executives, planned three months ago, would be over before he got there. The following day he was due in Frankfurt to give the keynote address at a major information systems conference. Flying to the United Kingdom to connect to Frankfurt would be a hassle and unnecessary since the purpose of stopping in the United Kingdom was now totally negated. Glancing up at the departure board, McPherson was surprised to see a 7:55 PM boarding departure for a plane to Frankfurt, nine gates away. Pulling into the gate at 8:02 PM, he discovered several things:

- 1) The plane was at the gate, and with commendable dispatch the gate agent relieved him of his London boarding pass and his London-to-Frankfurt ticket and hustled him onto the plane minutes before the door closed.
- 2) The cabin attendant, giving him his favourite drink, explained that because of favourable tail winds across the Atlantic and the fact that eight passengers (plus now McPherson and one other) had very tight connections, they had decided to hold the plane for 15 minutes to get the extra passengers and still arrive on schedule. The note of pride in the cabin attendant's voice was evident.

One and a half hours later, appropriately wine and dined, McPherson drifted off to sleep, reflecting on what a remarkable case study had played out in front of him in the previous two hours. Information technology, operations strategy, management control, an empowered (and unempowered) workforce, and service management had been interwoven into a tableau. A revised format for his speech in Frankfurt began to emerge. Best of all, he would not have to go through a case release process because it had all happened to him.

NOTES :

---

---

---

---

## Chapter 2

### Fundamental Concepts – Digitisation

#### 2.0 Introducing Binary

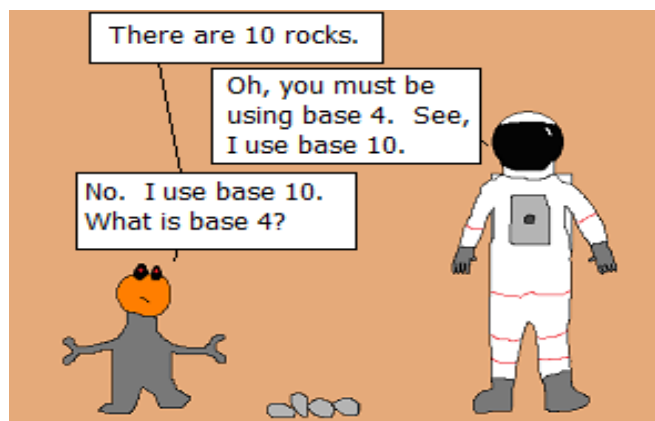
Let's learn to count! 1, 2, 3, 4... and what comes next? Generally we count in base 10, which means we have 10 symbols to represent the numbers – the first of these symbols being '0'. What happens when we get to 9? We run out of different symbols, so we add a number to the column to the left and then go back to our first symbol – i.e. 10. Similarly when we reach 19, we increase the previous column by 1 and return to the first symbol – i.e. 20. When we reach 99 we have no more symbols for either column so we add a new column and reset the other numbers to the first symbol – i.e. 100.

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11... ...19, 20, 21... ...98, 99, 100, 101...

This might seem trivial, but it becomes more relevant when we use a different base. Consider if we counted in base 16 (hexadecimal) – i.e. where we have 16 different symbols to use. For this we use the number symbols from 0-9 and then the letters from A to F. In this case we only run out of symbols when we reach the number 16 (or F), then we need to add a 2<sup>nd</sup> column.

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, 10, 11... ...19, 1A, 1B, 1C, 1D, 1E, 1F, 20, 21... ...FE, FF, 100, 101...

Note that here the symbols 10 represent the same as the decimal (base 10) number 16 and the symbols FF are the same as the decimal number 255. We can count using any base, depending on how many different symbols we have – our standard is 10, probably because we have 10 fingers.



Every base is base 10.

With computers we have 2 symbols, and so counting in binary (or base 2) is essential. The 2 symbols could be switches that are either On or Off, it could be the direction of magnetism (+ or -), it could be different voltages or different levels of light intensity depending on whether data is being stored or transferred using different media. To keep things simple, we will use the symbols '0' and '1' – each of which is a binary digit, or a **'bit'**. The same principle applies that when we run out of symbols we create a new column and return to the first symbol, only with binary this happens quicker.

0, 1, 10, 11, 100, 101, 110, 111, 1000, 1001... ...1111, 10000, 10001... ...11111110, 11111111

A ‘**nibble**’ is a sequence of 4 bits, such as 1001, 0011 or 1111. There are 16 different variations of this, which is of course the same as a single hexadecimal character. Each of these combinations could be used to represent the different things we need to store on the computer, but with only 16 variations we couldn’t even encode the letters of the alphabet, so generally we store things in bytes. A ‘**byte**’ is a sequence of 8 bits, such as 10010101 or 00000100. A byte gives a total of 256 different combinations, or FF in hexadecimal!

## 2.1 Introduction to Encoding

As mentioned, the different combinations can be used to represent different things, such as the letters in the alphabet, or colours, or sounds and so on. Before we consider how to encode things in binary, let’s consider another famous code – Morse Code. Morse code was developed in America as a way to communicate using electric current through telegraph machines. At first look the code appears to use dots and dashes to encode the letters of the alphabet – which would be similar to binary, however Morse Code actually uses base 5 as there are also the gaps between the dots and dashes. Firstly there is an intra-character gap, there is also a short gap between letters and a longer gap between each word.

The famous “SOS” call, or “...---...”, is an international distress signal. Note that it consists of all 5 different code symbols, including the gaps between dots and dashes. It is worth highlighting that “E” is encoded as “.” and “T” is encoded as “-“. As E is the most common character in the alphabet it is given the shortest possible code, followed by T. Clearly giving frequently occurring characters a shorter code will enable a message to be sent more quickly. The necessity to have different length gaps is further highlighted as without it the stream “-..-“ could represent the uncommon letter “X”, or the word “TEET”.

With binary the letters of the alphabet are encoded using just the two symbols (0 and 1). ASCII (American Standard Code for Information Interchange) is a standard way of encoding letters, while Unicode is an industry standard for representing most of the world’s writing systems. As well as representing the alphabet, binary is used for representing everything stored on a computer or shared between them. For example, colours are normally represented as 3 bytes – one for the amount of Red, one for the amount of Green and one for the amount of Blue (RGB). Once a colour has been encoded, it is straightforward to store a picture as a sequence of colours to represent each pixel. Once we have a picture we could store a moving picture as a sequence of pictures changing over time. Sounds are stored in a similar way.

Digitisation is the process of converting ‘stuff’ into binary, whether that ‘stuff’ is letters, colours, pictures or sounds. As things are converted to a digital format, they can then be easily stored on various

## International Morse Code

1. The length of a dot is one unit.
2. A dash is three units.
3. The space between parts of the same letter is one unit.
4. The space between letters is three units.
5. The space between words is seven units.

A	● —	U	● ● —
B	— ● ● ●	V	● ● ● —
C	— ● — ●	W	● — —
D	— ● ●	X	— ● ● —
E	●	Y	— ● — —
F	● ● — ●	Z	— — ● ●
G	— — ●		
H	● ● ● ●		
I	● ●		
J	● — — —		
K	— ● —	1	● — — — —
L	● — ● ●	2	● ● — — —
M	— —	3	● ● ● — —
N	— ●	4	● ● ● ● —
O	— — —	5	● ● ● ● ●
P	● — — ●	6	— ● ● ● ●
Q	— — ● —	7	— — ● ● ●
R	● — — ●	8	— — — ● ●
S	● ● ●	9	— — — — ●
T	—	0	— — — — —

media, and also easily shared and copied between machines. The ability to do this has a significant effect on a variety of digital products and has impacts on the quality, value and ownership of these products. For instance, consider how music has changed from vinyl records to cassette tapes to cds to mp3.

## 2.2 Encoding Objectives

Clearly there are multiple different ways of encoding things and different choices for how something can be represented by a code word, this is illustrated by the different encoding standards and the codecs used to code and decode them. A good encoding standard should follow certain encoding objectives.

*Each code word corresponds to exactly one symbol*

Clearly there would be a lot of confusion if a code word could be decoded into more than one symbol, or if a symbol could be encoded in different ways.

*Decoding should not require any look ahead*

This is known as the **prefix property** where there are no valid code words that are a prefix (or the start) of another valid code word. Suppose we had the symbols A, B and C and we represented them with the codes 0, 1 and 01 respectively. This would fail the prefix property as if the encoded message '01' was sent it wouldn't be clear whether the message is 'AB' or just 'C'. Extending this, as decoding the message should not require any look ahead, when we receive the first '0', we shouldn't need to wait for any further codes to be certain that the message begins with 'A'.

*The length of a code for one symbol should not exceed the length of a less likely symbol.*

As discussed previously with Morse code, there is a good reason that "E" has the shortest encoding – it is the commonest letter. Given the probabilities of each symbol, we can express this mathematically;

$$\text{If } P(m_i) < P(m_j) \text{ then } L(m_i) < L(m_j)$$

Where L represents the length of the code word.

*There should be no unused short codes*

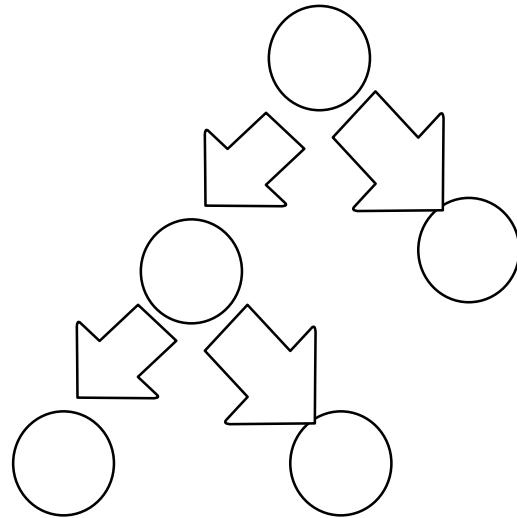
The objective is to the shortest possible encoding, so where possible the shorter codes should be chosen with no unused short codes either as standalone encodings or as prefixes for a longer used code.

## 2.3 Huffman Coding

Code which adheres to these objectives, particularly the prefix property, is often referred to as prefix code, and often as Huffman code, after the Huffman algorithm which is often used to generate such encodings. The method relies on constructing a **'heap'**, based on the frequencies of each symbol. The heap is a special kind of **'binary tree'**, which is a key data structure used to store data. A tree is a linked structure where parent nodes are connected to child nodes, with a visualization usually showing a

root node at the top of the tree, and leaf nodes at the very bottom. With a binary tree each node can have at most 2 children, as shown here;

In this example the top node is the root node, which has 2 children. The root's left child is also a parent node with 2 children, both of which are leaf nodes, while the right child of the root is also a leaf node. Note that this is a binary tree as each node has either 0, 1 or 2 children.

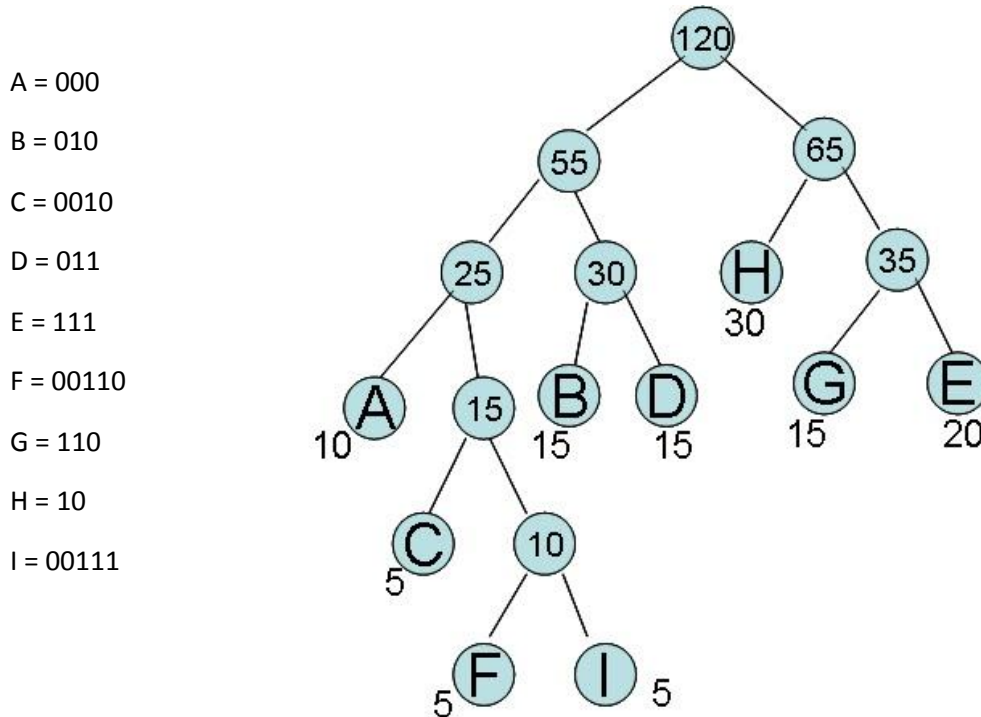


A heap is a binary tree where each node stores a value, and every parent's value is greater than any of their children, with the root node logically having the highest value. For the purpose of Huffman coding, the values stored in each node are the frequencies of each symbol that needs to be encoded, and each symbol is placed in a leaf node.

The process of constructing a heap begins with the frequency of each member of the set of elements that need to be encoded. For example with the following 9 letters and their frequencies in brackets:

*A (10), B (15), C (5), D (15), E (20), F (5), G(15), H (30), I (5)*

Then a tree is constructed where the lowest frequencies appear lower in the tree. The codes are then allocated by starting at the root and counting a 0 to move to the left and a 1 to move to the right, for example in the following tree H is reached by stepping right and then left and so is given the code 10.



## 2.4 Media

This week's videos can be found here:-

Microsoft's Vision of Future Productivity

<http://www.youtube.com/watch?v=t5X2PxtvMsU>

1001 Joke Reboot

<http://www.youtube.com/watch?v=xc8s3LPehWM>

## 2.5 Vocabulary

Here is the 2<sup>nd</sup> set of vocabulary.

- Fundamental - ซึ่งเป็นรากฐานสำคัญ
- Ownership - กรรมสิทธิ์
- Comparison - การเปรียบเทียบ
- Procedure - ขั้นตอนการ
- Element - ธาตุ
- Hexadecimal - เลขฐานสิบหก
- Magnetism - อำนาจแม่เหล็ก
- Stuff - สิ่ง
- Value - คุณค่า
- Digit - เลขโดด
- Sequence - ลำดับ
- Intensity - ความเข้มข้น
- Gaze - เพ่งมอง
- Strategy - กลยุทธ์
- Fortunate - โชคดี
- Decade - ทศวรรษ
- Extraordinary - วิสามัญ
- Escalator - บันไดเลื่อน
- Splendid - โอโถง
- Irritated - ที่ได้ทำให้โกรธ
- Trudge - การย่ำ
- Sprint - วิ่ง



- Hassle - การรบกวน
- Negate - ลบต่าง
- Glance - จำเลื่อง
- Interwoven - สาน
- Tableau - รูป

## 2.6 Key Questions

- 1) What is the difference between decimal, binary and hexadecimal?
- 2) What is the decimal number 255 in binary? In hexadecimal?
- 3) What is a bit? A nibble? A byte?
- 4) What is the prefix property?
- 5) What is a binary tree?
- 6) Explain the process of Digitisation.

# Chapter 3

## Communication Networks

### 3.0 Quick Quiz

0011101000110101101 001110100010011  
100010101111110000 0001100 1001000101010100 011101110110000100100?

A – 111	C – 1101	E – 01010	H – 01011	I – 000
J – 01100	K – 01101	M – 01110	N – 01111	O – 0010
Q – 0011	R – 0100	S – 1100	T – 10000	U – 101
W – 10001	Y – 10010	Z – 10011		

### 3.1 Network Simulation Exercise

Using just string, this exercise involves sending messages at 1 bit per second through a simulated network. The game illustrates the key elements of networks and the advantages and challenges of different network topologies.

### 3.2 Important Computing Laws

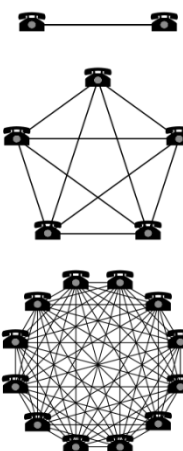
#### Moore's Law

Moore's Law states that the number of transistors on integrated circuits doubles approximately every two years. Gordon Moore recognized the trend in 1965 and the trend has remained relatively consistent ever since, often being used to predict the future and to set targets for the hardware industry. The law is often reflected on as performance improvement doubling every 18 months to 2 years, while costs remain the same – the effect noticed by general users.

#### Metcalf's Law

Metcalf's Law states that the usefulness of a network increases by the square of the number of users connected in the system ( $n^2$ ). The law reflects that as more users are connected to a network, it becomes more valuable – while one telephone isn't very useful, two telephones can connect with each other, as you add more telephones, more connections can be made.

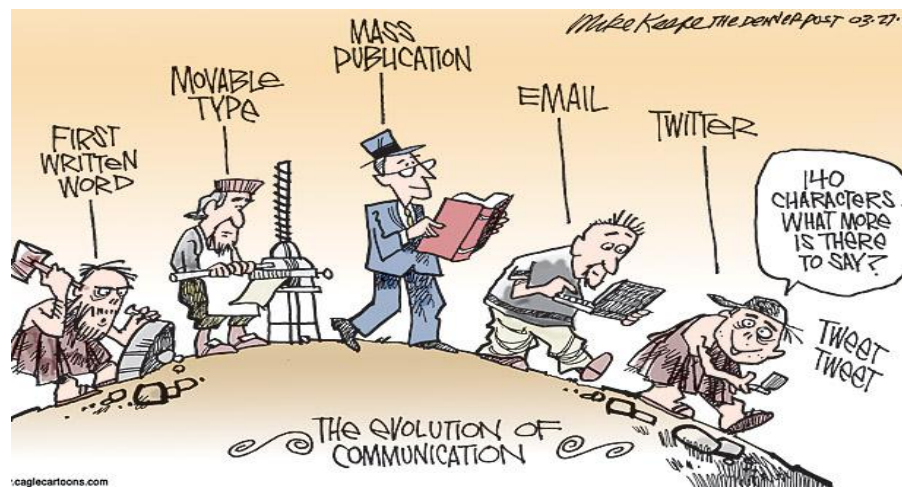
The law has been used to predict growth in web based business, for instance in social networks, however there are some reservations. Firstly there are difficulties in assessing the 'value' of a network, but assuming we could, would all connections be equally valuable?



### 3.3 Assignment

Read the article "Metcalf's Law is Wrong" by Bob Briscoe, Andrew Odlyzko and Benjamin Tilly, IEEE Spectrum, July 2006 and write a summary of its contents - (around 500 words) - Do you agree with the authors?

### 3.4 The Evolution of Communication



#### *Gesture*

The earliest communication between humans was using body language, such as by making gestures – consider the early cavemen grunting and waving at each other. Gesture based communication has remained a natural interface and interestingly today gesture is increasingly returning as the natural human interface with technology – consider swiping an e-book reader to turn a page, or the interface with the iPad.

#### *Speech*

It is believed that a mutation of the FOXP2 gene enabled humans to speak somewhere between 1.5 million and 200,000 years ago. Speech dramatically sets humans apart from animals and crucially it facilitates the transfer of knowledge from one generation to another. Speech also forms the basis of later written languages.

#### *Writing*

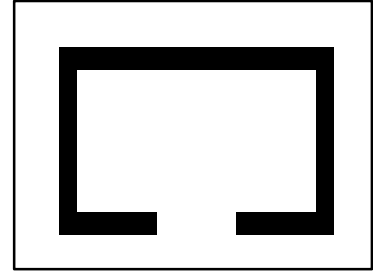
There was a gradual progression towards what we now consider as writing. This begins with the earliest cave paintings, dating from around 30,000BC. Around 10,000BC petroglyphs started to emerge where the picture is carved out of the rock surface, later pictograms were used as a sequence of drawings which could tell a story, and later still ideograms were used to represent concepts such as emotions as well as just objects.

It still took many millennia before writing was developed with technological advances and inventions taking a (comparatively) long time. Around the 3<sup>rd</sup> millennium BC the Sumerians developed the first writing system (although writing was also independently developed in China around the same time). The Sumerian system evolved from pictograms and developed into a language called Cuneiform. This was originally around 1,000 characters but refined to around 400 characters in Hittite Cuneiform.

From this the Egyptian hieroglyphs were created – a complex system where writing was figurative, symbolic and phonetic. By pressing into clay, or carving into rocks knowledge could be preserved and passed on with some examples still remaining today. The Egyptian hieroglyphs sometimes represented words and sometimes represented sounds, and it was the phonetic elements which

became key to developing an alphabet. The Egyptians used 22-24 phonetic hieroglyphs for recording foreign names, i.e. when traders visited and no hieroglyphs existed to write their names, a name was created out of characters that sounded the same.

One example is the Egyptian glyph “Pr”, which meant house (or floor plan). As workers migrated some took parts of the writing system, notably the Canaanites, where the symbol took on the Canaanite word for house – which was “bayt”. “Bayt” evolved to “bet” and then “beta” and became the modern letter “B”.



Trade routes expanded through the Mediterranean connecting the Egyptians with the Phoenicians and the Greeks, each with their own languages with its own features. In Greek vowels were essential, while in other languages they became awkward to write. Therefore as the Greek alphabet was developed vowels were given equal status. The Latins, or Romans, were similar to the Greeks, and eventually a set of 26 symbols were settled upon. Obviously other tribes developed different alphabets, and of course the Thai alphabet has a different history, but many stem from the same origins.

### *The Printing Press*

One of the most important inventions in all human history was the development of the printing press – by Gutenberg in around 1440AD. It is worth noting that the Chinese had invented paper in 105AD, and sometime not long after 1000AD they had a system of wood block printing, but with the thousands of characters in the Chinese alphabet it was when Gutenberg adapted a wine press. Gutenberg is credited with the invention and he was named #1 person of the millennium by A&E Network and by Time Life, coming ahead of people like Christopher Columbus, Freud, Galileo Galilei, Einstein, William Shakespeare, Abraham Lincoln, Isaac Newton, Charles Darwin and Leonardo Da Vinci!

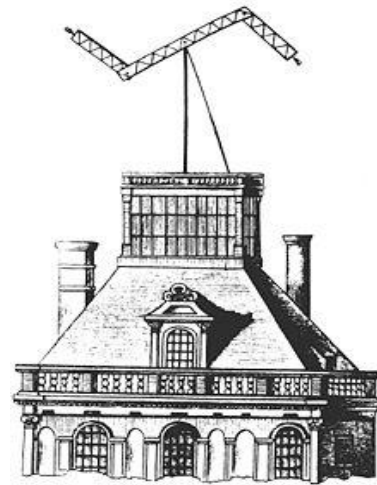
The effect of the printing press was massive – while previously it would take a monk around 20 years to hand write a copy of the bible, with the printing press books could be mass printed. Having more books opened the door to sharing knowledge, exchanging ideas and creating new knowledge.

### *Telecommunication*

Telecommunication is the transmission of signals over long distances – for communication. Primitive versions of this were fire beacons – i.e. visual cues – although drums and horns have been used also – i.e. audio. One limitation of using fire as a signal is that it only has two states – i.e. no fire means there is no problem, while a fire means there is a problem. This limits the number of messages that can be sent.

Hydraulic Telegraph dates from around 400BC, where two identical clay pots were filled with water and placed at a distance, but in line of sight. Each pot had a rod floating in it and at a given notice a plug was pulled simultaneously from both pots. The water drains out at the same rate from both pots with the rod sinking lower. Along the rod different messages are marked and at a signal both pots are plugged allowing the message recipient to read off a message based on how much water had drained out. Still a primitive solution, but it is technology allowing different messages to be sent.

In the late 18<sup>th</sup> century semaphore was developed, with different countries using slight different methods of encoding messages. One example from France involved placing on the roof of a high building a structure with 1 cross bar (that had 4 different positions) and 2 arms extending from the end of the cross bar (these arms had 7 positions each. The structure was around 8 metres in size, allowing it to be seen over long distances. With  $4 * 7 * 7 = 196$  different states, plenty of messages could be sent and the message would arrive much quicker than a person could travel. Issues such as security, or broken connections, were as much an issue in those days as with today's communication networks.



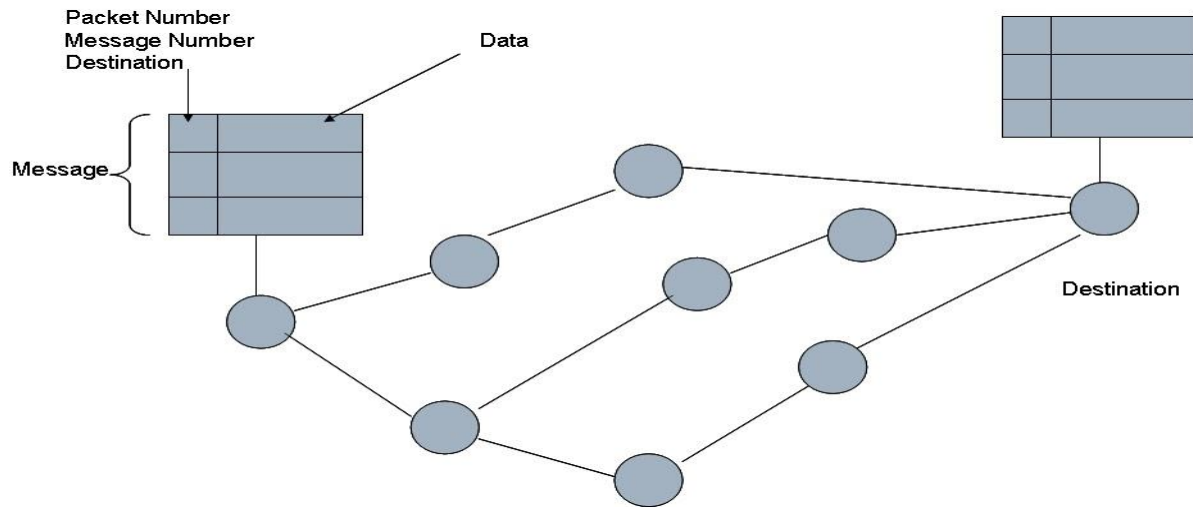
We have previously mentioned Morse, and the Electrical Telegraph massively sped up communications by using the properties of electricity to send electrical signals very quickly along wire cables. On some occasions, such as during war time, the wires became problematic and various researchers investigated alternatives with electromagnetic waves allowing the wireless radio and later television signals.

### 3.5 Telecommunication Systems

Telecommunication Systems are a combination of hardware and software that are arranged to transmit data from one location to another. Telecommunication systems perform a variety of tasks to ensure the message is successfully received, largely hiding the process of transmitting data from the users of the system. Some of the tasks include;

- Establishing an interface between sender and receiver.
- Routing messages along the most efficient paths
- Some basic information processing to make sure the right message is received by the right recipient.
- Converting message speeds, as messages move between faster and slower cables.
- Controlling information flow through the network.

One important concept is the idea of 'packet transfer', whereby in order to improve the efficiency of a network by breaking the message down into packets, or smaller bundles of data. The details of packet size and format vary depending on the system and protocol being used. Essentially the data is broken down into packets, with each packet containing a header of information needed to route the message and then rebuild it on arrival. Each packet can then take different routes through the network.



### 3.6 Protocols

A protocol is a set of rules that govern a communication system to ensure the communication is effective. Both the sender and the receiver of a communication act need to agree on the protocol in order for it to be effective. For instance, during class there is a protocol that English language will be used, and generally the teacher does the talking, unless a question is asked. Were a class to break the protocol, for instance by the teacher speaking in German, or the students talking amongst themselves, then the communication act wouldn't be as effective. There are several important protocols governing how computers communicate with each other, for example;

- TCP / IP = Transmission Control Protocol / Internet Protocol
- FTP = File Transfer Protocol
- WAP = Wireless Application Protocol
- HTTP = HyperText Transfer Protocol

TCP/IP is clearly an important protocol, with the Internet Protocol being a suite of other protocols that work together to enable the internet to work. It is a layered protocol, consisting of 4 layers;

- *Link Layer* – responsible for actual data exchange, error checking, bit rate etc.
- *Internet Layer* – responsible for things like addressing and routing.
- *Transport Layer* – responsible for forming the data packets, adding header information, ensuring the right packets arrive etc.
- *Application Layer* – responsible for providing user services such as FTP, SMTP (Simple Mail Transfer Protocol) and BitTorrent.

One of the features of IP is the IP address, where each machine is given a unique IP address, for example 172.17.28.143. This IP address means the machine can be located and messages routed towards it. Notice that the IP address is actually a 32 bit word, which is converted into a decimal-dot notation, with 4 numbers separated by a dot. The 4 numbers are each a byte in size allowing a number between 0 and 255. The result is around 4.3 billion different IP addresses ( $256 \times 256 \times 256 \times 256$ ).

Fortunately clever ways have been developed so that there are enough IP addresses, such as by offering Dynamic IP addresses, where devices only use an IP address when they are connected to the network. Fortunately also we don't need to remember the IP address of the machine we are trying to communicate with, as the DNS (Domain Name Service) translates IP addresses into human readable, Natural Language, such as a URL.

### 3.7 Cables

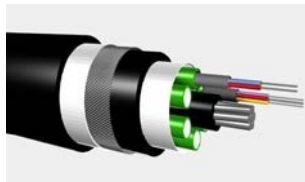
There are different types of cables used in modern computer networks, each with different capacities, speeds and costs.



*Twisted Pair Wire* consists of pairs of wires are twisted next to each other to cancel out electromagnetic interference (EMI). The wires are insulated and create a common, cheap option. The cables are also thin and flexible, which is useful for wiring into a building.



*Coaxial Cable* consists of a solid conductor, normally copper wire, which is then surrounded by insulation and then shielded to prevent the signal from leaking.

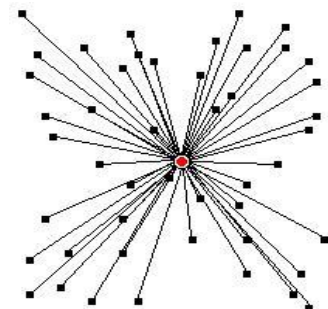


*Fiber Optic Cables* use the medium of light, which makes them comparatively more expensive, but very fast.

### 3.8 Network Topologies

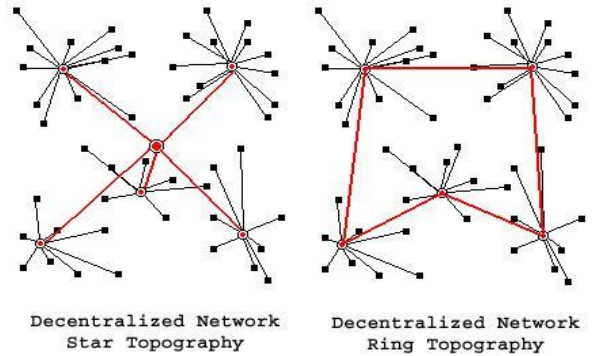
There are several different ways in which nodes can be connected, and each has its own features.

*Centralised Network* – In a centralized network all nodes are connected via a single central node. All communication passes through that central node. This allows close control of the network, however it is limited by the capacity of the central node. It is comparatively inefficient and has the risk of a single point of failure – i.e. if the central node fails, all communication fails.

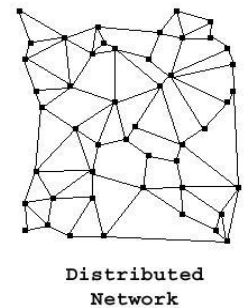


Centralized  
Network

*Decentralised Network* - In a decentralized network there are more than one central node, and each of these nodes manages their mini network and also communication with other networks. You could imagine each 'central' node as a different country, or a different city. This network adds greater efficiency and robustness as if one node fails, others can continue to operate. However, the administrative burden increases and it is harder to control the network. In the info graphics there are 2 types of decentralised networks demonstrated.



*Distributed Network* – In a distributed network, or P2P network, any node can conceivably connect with any other node. This offers shortest route efficiencies, multiple route efficiencies and is a robust, infinitely scalable topology. There are of course issues, not least the administrative difficulties which is managed through standards, policies and protocols. There is also an issue when certain routes (the backbone) become congested, as this limits the speed and efficiency of the network.



One key to improving the efficiency of the network is to focus on the backbone – the key routes, for example the transatlantic connections between Europe and the US. Other ways to improve a network's capability is through routing and caching. Routing algorithms ensure that each packet takes the best route through the network with each router deciding where to send the next packet. Caching improves a network's efficiency by reducing the amount of data that needs to be sent. A cache is a closer copy of the data that can be shared between multiple users to save downloading it more than once.

This intention to improve the efficiency, speed and reliability is likely to continue into the future, with more devices, and more interfaces allowing cheaper and more reliable connectivity.

### 3.9 Media

This week's videos can be found here:-

Where the Hell is Matt? 2006

[http://www.youtube.com/watch?v=bNF\\_P281Uu4](http://www.youtube.com/watch?v=bNF_P281Uu4)

Lord of the Rings, Lighting the Beacons

<http://www.youtube.com/watch?v=i6LGJ7evrAg>



### 3.10 Vocabulary

Here's the 3<sup>rd</sup> set of vocabulary:-

• Frequency	-	ความถี่
• Representation	-	รูปแทน
• Simplify	-	ลดความซับซ้อน
• Corresponding	-	ตรงกัน
• Exactly	-	เพง
• Optimisation	-	การเพิ่มประสิทธิภาพ
• Length	-	ความยาว
• Encoding	-	การเปลี่ยนเป็นรหัส
• Exceed	-	เกินกว่า
• Protocol	-	โปรโตคอล
• Probability	-	ความน่าจะเป็น
• Particular	-	โดยเฉพาะ
• Node	-	โหนด
• Facilitate	-	อำนวยความสะดวก
• Estimate	-	ประมาณ
• Hieroglyph	-	อักษรอียิปต์โบราณ
• Phonetics	-	สัทศาสตร์
• Floor plan	-	แผนชั้น
• Square	-	สี่เหลี่ยม
• Usefulness	-	ความมีประโยชน์
• Generations	-	รุ่น
• Speech	-	การพูด
• Carving	-	การแกะสลัก
• Ability	-	ความสามารถ
• Particularly	-	โดยเฉพาะ
• Demonstrate	-	สาธิต
• Hindrance	-	อุปสรรค
• Consonants	-	พยัญชนะ
• Semaphore	-	สัญญาณ

• Hence	-	ด้วยเหตุนี้
• Movable	-	เคลื่อนย้ายได้
• Mass	-	มวล
• Longevity	-	ช่วงชีวิต
• Incision	-	รอยตัด
• Pictogram	-	รูปสัญลักษณ์
• Narrate	-	เล่าเรื่อง
• Ideogram	-	ตัวเขียนแสดงความคิดเห็น
• Crucial	-	สำคัญมาก
• Transmission	-	การส่งผ่าน
• Beacon	-	กระโจมไฟ
• Efficiency	-	ประสิทธิภาพ
• Domain	-	โดเมน
• Congestion	-	ความแออัด
• Reliance	-	การพึ่งพา
• Packets	-	แพ็คเกจ
• Layer	-	ชั้น
• Editorial Tasks	-	งานบรรณาธิการ
• Destination	-	ปลายทาง
• Comparatively	-	เปรียบเทียบ
• Scalable	-	ปรับขนาดได้
• Deal	-	จัดการ
• Actual	-	ที่จริง
• Notation	-	หมายเหตุ
• Various	-	ต่างๆ
• Robust	-	แข็งแรง
• Infinitely	-	เพียบ
• Inefficient	-	ไม่มีประสิทธิภาพ
• Stream	-	ไหลหลั่ง
• Twisted	-	บิด

### 3.11 Key Questions

- 1) What is Moore's Law?
- 2) What is Metcalfe's Law?
- 3) What is the role of a telecommunication system?
- 4) What is TCP/IP?
- 5) What is the purpose of a protocol?
- 6) Briefly introduce and compare the following network topologies:-
  - a. Centralised
  - b. Decentralised
  - c. Distributed

## Chapter 4

### Social Networks

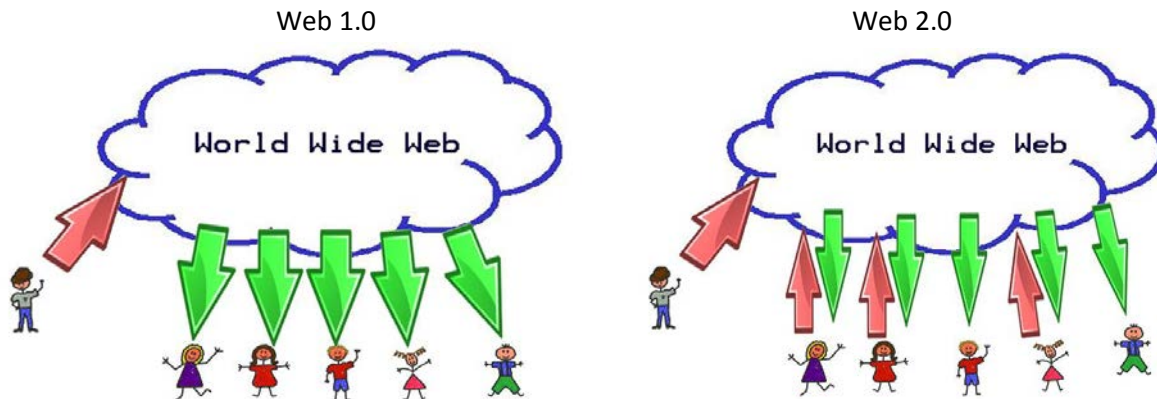
#### 4.0 Introducing Social Networks and Web 2.0

"Family, and friends and their families, that together create an interconnected system through which alliances are formed, help is obtained, information is transmitted, and strings are pulled. In an organizational setting, it usually constitutes the group of one's peers, seniors, and subordinates who provide privileged information on how to get things done, how the power structure operates, and who holds the strings at present."

BusinessDictionary.com

Obviously today we think of social networks as web based social networks, such as Facebook etc. Here users can connect with friends, employers, employees and groups through public or private messages, sharing information through their profiles.

Social networks has developed the creation of social media – a shift from read-only media towards collaborative creation on new media. Web 2.0 is a new version of the web where users contribute to the content of the web, through blogs, wikis and simply social bookmarking, tagging and liking.



Whereas the traditional (Web 1.0) version of the web involved a webmaster creating the content for users of the web to read, under the current view (Web 2.0) of the web, all users participate in the creating content for the web and this has led to a new information revolution.

#### 4.1 Information R/Evolution

An excellent video by Michael Lee Wesch, a cultural anthropologist from Kansas State University can be found here:-

<http://www.youtube.com/watch?v=-4CV05HyAbM>

Digital Information is different, changing our assumptions about information. Traditionally the characteristics of information are that it is a 'thing', with a 'logical place', where it can be found – on a shelf, in a file system, in a category. This comes from the age when information is printed in books, taking on a physical form which then needs to be stored on a shelf in a library. Digital information is different, with no single physical form, and not necessarily a single category or location where it can be found.

Traditionally managing information involved managing categories, and needed expert librarians to decide where the information belonged (or would be easiest to find). Again, digital information is different, as in Web 2.0 information is managed by all members of the community. Today the responsibility for creating, critiquing, harnessing, organizing and understand digital information is on all of us. Are we ready?

## 4.2 The Machine is Us/ing Us

Another excellent video by Michael Lee Wesch can be found here:-  
[http://www.youtube.com/watch?v=NLIgopyXT\\_g](http://www.youtube.com/watch?v=NLIgopyXT_g)

This video examines how modern web technologies such as Ajax, XML have led to the changes in information. A key change was the separation of **Form** and **Content**. Content being the information, and form being how it is presented. That separation meant that users no longer needed to understand 'complicated' code in order to upload content to the web. This shift has led to the machine becoming 'us' – with 'us' responsible for deciding which information is correct, which information is worth spreading, and where different information belongs.

“When we post and then tag pictures, we are teaching the Machine... Each time we forge a link, we teach it an idea... Each of the billions of times that a user clicks on a link each day, it's teaching the Machine. The web is no longer just linking information, Web 2.0 is linking people – people sharing, trading and collaborating.”

This means we will need to rethink a few things;

- Copyright
- Authorship
- Identity
- Ethics
- Aesthetics
- Rhetorics
- Governance
- Privacy
- Commerce
- Love
- Family
- & Ourselves.

## 4.3 Presentation

Given the Michael Wesch video watched in class, in groups prepare to present how we will need to rethink these things.

## 4.4 History of Social Networks

While modern social networks are hugely popular and successful, the first social networks came in the form of BBS, or bulletin board systems, developed in the late 1970s. These were popular among

nerds, who largely hid in basements. Interactions were simply text based, via dial up connections connecting to a central node among geographically close groups of users. As the Internet kicked in several sites created early social networks, AOL, Classmates, Sixdegrees, Friendster, LinkedIn, MySpace, but the phenomena really took off with Facebook – before that invites were often considered spammy.

Other social networks have taken off in other languages, and in other countries, such as Qzone in China and Odnoklassniki in Russia. More recently Twitter through microblogging is affecting how real time search is conducted in a community fashion.

#### 4.5 Going Viral

One of the knock on effects of Social Networks is when something ‘goes viral’. Like a virus spreading from person to person, or computer to computer, sometimes news articles, videos, photos or ideas go viral as members of social networks share, like and comment on it. But why? What drives us to decide to share certain things, and not others? Remember the “Where the hell is Matt?” video – why would you share it? Consider the following movies and discuss why you might be driven to help it ‘go viral’.

The German Coastguard

<http://www.youtube.com/watch?v=gh5xu35bAxA>

Chinese Backstreet Boys

<http://www.youtube.com/watch?v=N2rZxCrb7iU>

Bear Fights Man for Fish

<http://www.youtube.com/watch?v=cplj6zyehOU>

Bouncing Balls

<http://www.youtube.com/watch?v=KMI5l6mOySU>

#### 4.6 Vocabulary

Here are the latest set of words:-

- |               |   |                 |
|---------------|---|-----------------|
| • Cache       | - | จุมทรัพย์       |
| • Alliance    | - | การเกี้ยวพาราสี |
| • Obtain      | - | ได้รับ          |
| • Participate | - | มีส่วนร่วม      |
| • Participant | - | ผู้มีส่วนร่วม   |
| • Critique    | - | วิพากษ์         |
| • Utilizes    | - | ใช้             |
| • Recap       | - | ปะยางรถ         |
| • Peer        | - | ผู้เท่าเทียม    |

- Privileged - ซึ่งมีสิทธิพิเศษ
- Subordinate - ผู้ใต้บังคับบัญชา
- Characteristics - ลักษณะ
- Interconnected - ที่เชื่อมต่อกัน
- Antenna - เสาอากาศ
- Capacity - ความจุ
- Syndication - การแจกจ่าย

#### 4.7 Key Questions

- 1) Briefly discuss the evolution from Web 1.0 to Web 2.0 and how it has affected your life.
- 2) How can companies leverage the power of viral videos to spread their advertising message?

# Chapter 5

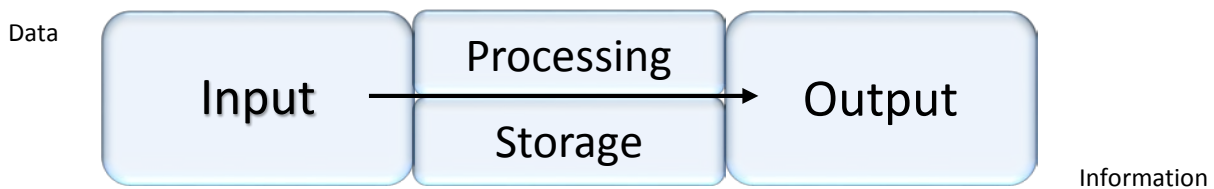
## Hardware

### 5.0 Hardware

Hardware components are the physical objects that you can touch – the tangible devices.

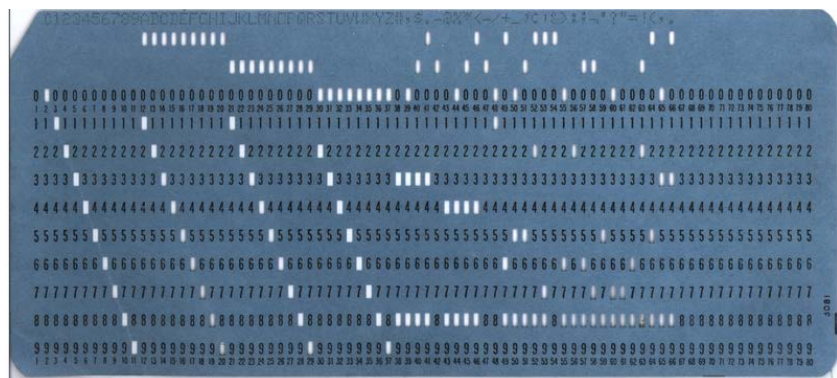
- Monitor
- Keyboard
- Data Storage
- Hard Disk
- Mouse
- CPU
- Motherboard
- Graphics Cards
- Etc...

If we consider the original definition of Information Systems, as Input -> Processing -> Output, with storage capability as well. There are hardware devices in each of these categories; input devices, output devices, storage devices and the processor.



### 5.1 Evolution of Input Devices

The earliest input devices were **punched cards**, essentially a card that could have holes punched in it, the locations of the holes could be used to store data, processing instructions; essentially any digital information represented by the presence or absence of holes.



Arguably the most common input device is the **keyboard**. The keyboard is a legacy from the typewriter, the mechanical device. The “QWERTY” layout that is now still found on most keyboards is due to the mechanical nature of former typewriters, and a design that aimed to stop bars from clashing with each other. Even though it isn’t the most efficient typing layout, users are unwilling to relearn how to type.



The **mouse** is another input device that has evolved from a mechanical contraption with wheels, through a mouse ball, until the optical mice of today. The mouse finally afforded direct interaction with the system and opened up interesting new interfaces – the GUI or Graphical User Interface. The mouse is so important that it is a part of the WIMP acronym from Human Computer Interaction (HCI). WIMP stands for Windows, Icons, Menus and Pointer – with Pointer referring to the mouse.

**Touch based input** is another important input technology. With touch based input again the user can interact directly with the machine, but the mouse a later invention became more popular, and more reliable. With recent tablets, touch based inputs have become more popular again. Multi touch is a recent development, enabling richer interactions.

**Mobile input** refers to inputs using a mobile device like a telephone. As texting became popular efficient ways of enabling textual input on a small device with a limited interface were needed. The standard interface was limited to the 10 number buttons and a couple of extra buttons. To facilitate all the letters of the alphabet the initial approach was multiple key presses, leading to T9 input. Other approaches include miniature full sized keyboards, and later predictive text combined with auto correction. The awkwardness of mobile input is also responsible for the development of netspeak such as “LOL”.

More recent developments include direct interaction through gesture based interactions and **haptic** devices. Haptic refers to the sense of touch, and is an area of continuing research, although examples include vibration feedback on joysticks. The Wii uses sensors to detect gesture inputs.

What is the future? Perhaps thought based interactions?

## 5.2 Evolution of Output Devices

Early output devices were CRT technology or Cathode Ray Tube monitors. That evolved to LCD (Liquid Crystal Display) and later plasma. The effects of this evolution has been increased desk space as the footprint of the monitor decreases. Improvements have also include better resolutions and refresh rates as well as a more modern style. Moving forward, are screens going to be necessary?

One of the challenges is finding ways to display on different size screens – we could call it the “Inch”, the “Foot” and the “Yard”, in other words finding ways to display on a portable mobile device, on a standard computer monitor and projected onto a large screen. Mobile projection is one option to enable users to share screens, while glass is an approach for small screen, personal screens.

Research is also ongoing into 3D displays, whether through holograms or perhaps through retinal implants. Implants are likely to continue to change the human computer interface.

*“Having tasted the benefit of interconnection, we will continue the process until virtual reality is “real reality”. That is, our senses will no longer be limited to the here and now. The ultimate user interface will be a direct neural stimulation and output. Our eyes will become cameras and our ears, microphones. We will touch, taste, and smell remotely. We will share direct brain-brain links, achieving “mind-meld” with others.”*

Steve Czarnecki (Lockheed Martin)

Output is not only for visual output, but output to the other senses are not as effective. Sound output has developed through higher quality speakers, but there are still challenges involved with effectively distributing sounds, or *not* distributing sounds. Touch output, as previously mentioned, is being addressed with haptic devices, while less progress has been made with digitising smell or taste.

### 5.3 Interfaces and Interaction Styles

The current interface between the human and the computer is largely explicit – i.e. fully and clearly expressed with nothing left implied. To make a more natural interface inputs, and outputs, should be more implicit. *Context Aware Computing* is when the system is able to sense context from the environment and determine the correct response. There is no need for me to authenticate myself to each group of students when I walk into class, so why do I need to do so when I enter the computing environment?

From a HCI perspective, let's briefly consider 4 different styles of interaction.

*Direct Manipulation* - interfaces where the user receives immediate feedback from their inputs, for example when dragging an icon from one folder to another, where the user can see precisely where and how they are moving the icon. There are advantages to this interaction as users feel in control of the computer and aren't likely to be intimidated. Learning time is also relatively short, due partly at least to the way users can quickly detect mistakes and correct them. However, the style isn't suitable for all scenarios as sometimes the information / space model is complicated, and if the space is too large, how can it be navigated – imagine trying to navigate the web through direct manipulation. A further challenge comes from the heavy computational demands on the computer system.

*Menu Systems* – with a menu system users can choose from a limited list of possibilities, often by selecting their option using a mouse or a touch screen. The benefit is that users don't need to remember commands, typing effort is minimal, and errors are trapped by the interface as users aren't presented with illegal options. Context sensitive help can also be given as it is easy for the system to track where the user is in it. On the downside, conjunction (AND) and disjunction (OR) actions are harder to represent. The style works best when there are a limited number of clear choices, otherwise structuring the choices can become complicated – choosing a destination on the BTS in Bangkok is straightforward through a menu, but choosing the route to get to the destination wouldn't be as easily structured. Experienced users also find menu systems slower than using short cuts or command language.

*Command Interfaces* – With a command interface users type instructions, such as with a unix command line. It is easy to implement on an inexpensive terminal. Commands of arbitrary complexity can be constructed using a combination of commands. The difficulty of using command interfaces comes from the need to learn commands – while it is quick and easy for experienced users, it is impossible for occasional users. Some way of allowing error detection and recovery is essential.

*Natural Language Interfaces* – With the command interface the vocabulary is limited, whereas the goal with Natural Language (NL) Interfaces is to allow more versatile language. The vocabulary is still limited today due to the ambiguity involved. However NL processing is to a level that allows casual users to make effective use of a system, although experienced users often find it involves too much typing. Perhaps the future of Natural Language Interfaces is voice recognition technology, with advances such as Siri.

## 5.4 The System Unit

The system unit is the case which contains the electronic components of the computer used to process data. While the interface devices (input and output) interact with the user, the system unit is where the data happens. Inside the box are several important components, not least of course the CPU or Central Processing Unit;

- Drive Bays
- Power Supply
- Sound Card
- Video Card
- Memory
- The Processor

The motherboard is the main circuit board of the system unit, through which other components connect. On the motherboard the CPU is a crucial unit which performs 2 key functions – a control unit and an arithmetic logic unit. Today most modern computers have a multi-core processor.

*Control Unit* – the component of the processor that directs and co-ordinates most of the operations in the computer

*Arithmetic Logic Unit (ALU)* – responsible for performing arithmetic, comparison and other operations.

Each instruction that the CPU processes consists of 4 basic operations, otherwise known as the machine cycle.

Step 1) *Fetch* - the control unit fetches the math problem instructions and data from memory.

Step 2) *Decode* - the control unit decodes the math problem's instructions and send the instructions and data to the ALU.

Step 3) *Execute* - the ALU performs calculations on the data.

Step 4) *Store* - the results of the math problem are stored in the computer's memory.

Most current PCs support pipelining, where the processor can begin fetching a second instruction before it completes the machine cycle for the first instruction. The processor also contains some registers which can hold data and instructions temporarily during processing, and a system clock which controls the timing of all computer operations.

The processor can get very hot during processing, so the heat sink and fans work to move heat away from the processor.

The other elements of the system unit are the RAM and the hard disks. The RAM (Random Access Memory) stores data temporarily, while the hard disk stores data even when the system unit is switched off. The most noticeable trend in ram and hard disks that capacity is increasing, while prices are declining.

## 5.5 Media

Find the videos from this class here:-

Minority Report Trailer

<http://www.youtube.com/watch?v=QH-6UImAP7c>

TED Talk – Pattie Maes

<http://www.youtube.com/watch?v=EMmhfbM9JAs>

Nokia vision of the future

<http://www.youtube.com/watch?v=L3wdG-wYN3s>

Scottish Voice Recognition

[http://www.youtube.com/watch?v=sAz\\_UvnUeuU](http://www.youtube.com/watch?v=sAz_UvnUeuU)

## 5.6 Vocabulary

Here are the latest set of words:-

- Paradigm - ตัวอย่าง
- Metaphor - คำอุปมา
- Implicit - โดยปริยาย
- Explicit - ชัดเจน
- Integration - บูรณาการรวมหน่วย
- Manipulation - การจัดการ
- Logical - ตรรกะ
- Conjunction - ร่วม
- Disjunction - ความร้าวฉาน
- Awkward - อึดอัด
- Volatile - ระเหย

## 5.7 Key Questions

- 1) Briefly describe the evolution of input devices
- 2) Briefly describe the evolution of output devices
- 3) Discuss the following interaction styles;
  - a) Direct Manipulation
  - b) Menu Systems
  - c) Command Interfaces
  - d) Natural Language Interfaces
- 4) What are the 2 key functions of the CPU?
- 5) What is the role of the heat sink?

## Chapter 6

### The Future of ICT

#### 6.0 Assignment

Within the groups assigned to you, choose a futuristic movie. In a similar way to the Minority Report movie in the previous chapter, identify some technologies which don't exist today. Prepare a presentation where you need;

- Introduce the Movie and technology
- Is it physically possible?
- How might it work?
- What effect would it have on our lives if / when it is developed?

#### 6.1 Changing Technology

*"As the century closed, the world became smaller. The public rapidly gained access to new and dramatically faster communication technologies. Entrepreneurs, able to draw on unprecedented scale economies, built vast empires. Great fortunes were made. The government demanded that these powerful new monopolists be held accountable under antitrust law. Every day brought forth new technological advances to which the old business models seemed no longer to apply. Yet, somehow, the basic laws of economics asserted themselves. Those who mastered these laws survived in the new environment. Those who did not, failed."*  
(Anon)

This quote is used at the start of Shapiro and Varian's book "Information Rules – a Strategic Guide to the Network Economy". At first read it appears to discuss the effects of the internet (dramatically faster communication technology) and how some people have leveraged it to earn great fortunes, while others have had troubles with anti-trust type laws. A key principle being that the basic laws of economics remain true even with new technology and the key to success with new technologies is understanding these laws.

However, the most interesting thing about this quote is that it actually refers to the telephone, a dramatically faster communication technology that became available at the end of the 19<sup>th</sup> century. The vast empires refer to the industrialists that transformed the US economy using new infrastructure (electricity and telecommunication). Yet over a century later the quote is still relevant, indicating that as new technology emerges it is the way in which we make use of it that affects how successful we are, and understanding core economic principles is essential.

*'If you asked a farmer 100 years ago what kind of advance he'd like to see in farming in the next century, he'd be likely to say something along the lines "a horse twice as strong that eats half as much oats".'*  
(Anon)

This quote draws reference to Moore's Law, as capacity and capability increases while costs decrease. Again as new technologies are developed (even as simple as 'animal husbandry') there is a continuing demand for greater cost efficiencies.

The traditional view of the computer is the Personal Computer (PC), i.e. as a box, with a keyboard, mouse and monitor connected to it, but this view is changing. Of course as laptops were developed, the first stages of mobility emerged. However, an important part of this model is that, when we want to do some work, we find the computer, i.e. we need to enter the world of the computer, rather than the computer being part of our human world. One thing that is changing is that computing services are becoming increasingly distributed throughout our physical world, and integrated into it. For example, there are now web enabled fridges, complete with a screen that can tell you the fridge's inventory or allow you to surf the web. Our personal computing devices are getting smaller through *miniaturization*, an effect of Moore's Law, and these devices are coming with us in the form of smart phones. This is part of the trend towards *Ubiquitous Computing*.



## 6.2 Ubiquitous Computing

*"Machines that fit the human environment instead of forcing humans to enter theirs will make using a computer as refreshing as a walk in the woods."*

*(Weiser 1991)*

The desktop paradigm (or desktop metaphor) comes from the goal of replicating a traditional physical desktop into a digital desktop which provides us a logical workspace. One of the goals of ubiquitous computing is to break away from this paradigm, and rather than forcing the user to search for the interface, allow the computer to find and then serve the user.

*"Activate the world. Provide hundreds of wireless computing devices per person per office, of all scales (from 1" displays to wall sized)... This is different from PDAs, dynabooks, or information at your fingertips. It is invisible, everywhere computing that does not live on a personal device of any sort, but is invisible in the woodwork."*

*(Weiser 1994)*

Weiser talks about moving away from the Personal Computer space where we have our own devices, to a world where the devices are embedded all around us, invisible in the walls, usable by all. Cloud computing today locates our data 'in the cloud', with the goal of making it available anywhere, anytime. Clearly not all would agree, with security and privacy concerns often being cited.

*"We wanted to put computing back in its place, to reposition it into the environmental background, to concentrate on human-to-human interfaces, and less on human-to-computer ones"*

*(Weiser 1999)*

Here Weiser talks of the important interfaces, between humans, while today much effort is put into developing the graphical user interfaces allowing convenient access to interface with the machine. Social networks, as previously discussed, have had a big effect on our human to human interfaces.

It is worth noting that these three quotes from Weiser that discuss the ideas of Ubiquitous (or Pervasive) Computing come from the 1990s, and that work is ongoing to realize the vision. There remain some challenges to defining appropriate interactions.

### *Implicit Input*

Typing on a keyboard requires ‘Explicit Input’, while with implicit input, our natural interactions within the physical environment are enough to provide the inputs – in other words, we can imply our inputs rather than needing to spell them out. For example, walking into a space is enough to announce one’s presence and identity – there is no need for a teacher to authenticate themselves with username and password when they walk into the classroom. Handwriting is a more natural input than typing – after all, handwritten books pre-date the printing press, so handwriting recognition allows us to step back to that means of input. However, speech pre-dates handwriting, so speech recognition would allow an even more natural input. Although, as previously introduced, the oldest form of communication is through body language and gesture, so gesture based inputs would form our most natural interactions.

Progress is being made with technologies such as accelerometers, tilt sensors, infrared range finders or table top computing.

### *Scalable and Distributed Output*

There is a need to display output on different screens, with different resolutions, depending on whether that display is for personal or shared consumption. These different outputs present different challenges. Coupled with this is the challenge of ‘moving stuff’. With thumb drives and cloud computing we are able to move data around in different ways to get it displayed in different places, yet still there is something unsatisfying about it. Several of the future prediction videos highlight the ability to move data through gestures.

### *Context Aware Computing*

Context aware computing refers to when the machine can sense from the environment to decide on the most appropriate response. If I can walk into a room and the technology there senses my presence, then it can already answer questions like ‘where’ and ‘who’, so how about sensing what I am doing? Perhaps the machine could determine why I am doing it, and be able to better predict when I will do other things that it could assist with, like a personal agent.

### *Augmented Reality*

Another feature that has appeared regularly in ‘future tech’ videos is augmented worlds, and virtual worlds. While a virtual reality (VR) world completely immerses the user in a digitally created space, and augmented world simply adds useful information to our world – such as directions to your next gate at an airport.

## **6.3 Media**

Some of the videos from this class can be found here:-

Virtual Sightseeing Scenic Viewer at Lisbon’s National Pantheon

<http://www.youtube.com/watch?v=GCB1OeYwMKc>

MOCOM 2020 – The Future of Mobile Media and Communication

<http://www.youtube.com/watch?v=FSyddkTMITc>

Soldiers Get Virtual Reality Therapy for Burn Pain

<http://www.youtube.com/watch?v=jNlqyyypoig>

## 6.4 Key Questions

- 1) What is Ubiquitous Computing?
- 2) How could Augmented Reality help your life?



## Chapter 7

### Software

#### 7.0 What is Software?

While hardware are the physical devices that we can touch, software are the processing instructions. Computer software is a computer program in the form of machine-readable instructions that directs a computer to perform a particular sequence of operations. Software is generally divided into application software and system software.

#### 7.1 Application Software

Application software consists of programs designed to make users more productive and / or to assist with personal tasks, for example;

- To make business activities more efficient
- To assist with graphics and multimedia projects
- To support home, personal and educational tasks
- To facilitate communications

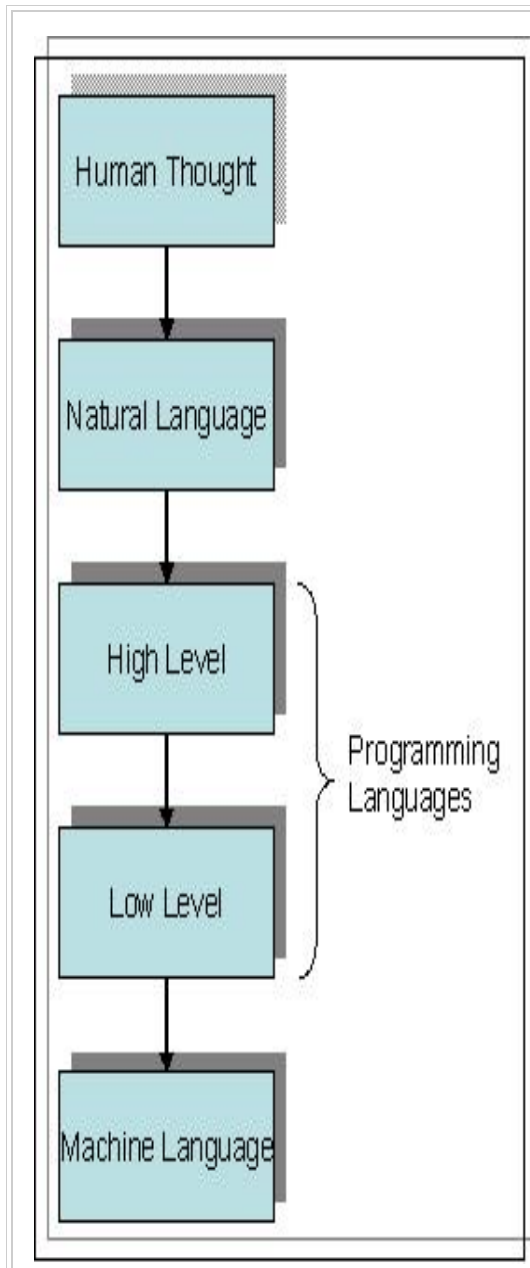
There are different types of application software including packaged software, web applications, open source software, shareware or freeware.

#### 7.2 System Software

System software acts as an interface between the user, the application software and the hardware. It includes the operating system and utility programs such as drivers that enable different pieces of hardware to function.

#### 7.3 Programming Languages

Computers work purely using binary – everything is converted into binary code for a computer to read and follow instructions. So, if we want a computer to follow a simple instruction  $x + y$  then this needs to be converted into binary. If 'x' was represented in binary by **0011** and 'y' by **0100**, the '+' might be represented by **1111** the instruction might be converted into **machine language** looking like **001111110100**. Fortunately we don't need to write programs in machine language in order to instruct a computer as this would be very error prone!



As humans, we use Natural Language to express our thoughts. Unfortunately it isn't possible to instruct a computer using Natural Language, as it is prone to ambiguities in meaning (*Semantics*) which computers can't identify – consider how you could define the word 'Spring'.

Natural Language can also have problems with grammar (*Syntax*) - it wouldn't be correct to say in English "Eat I now food" although we would probably understand. Therefore, **Programming Languages** have been developed in between **Natural Language** and **Machine Language** as neither are suitable. They are designed to have a more precise (unambiguous) syntax and semantics, to ensure the computer responds as intended, without using the error prone binary code.

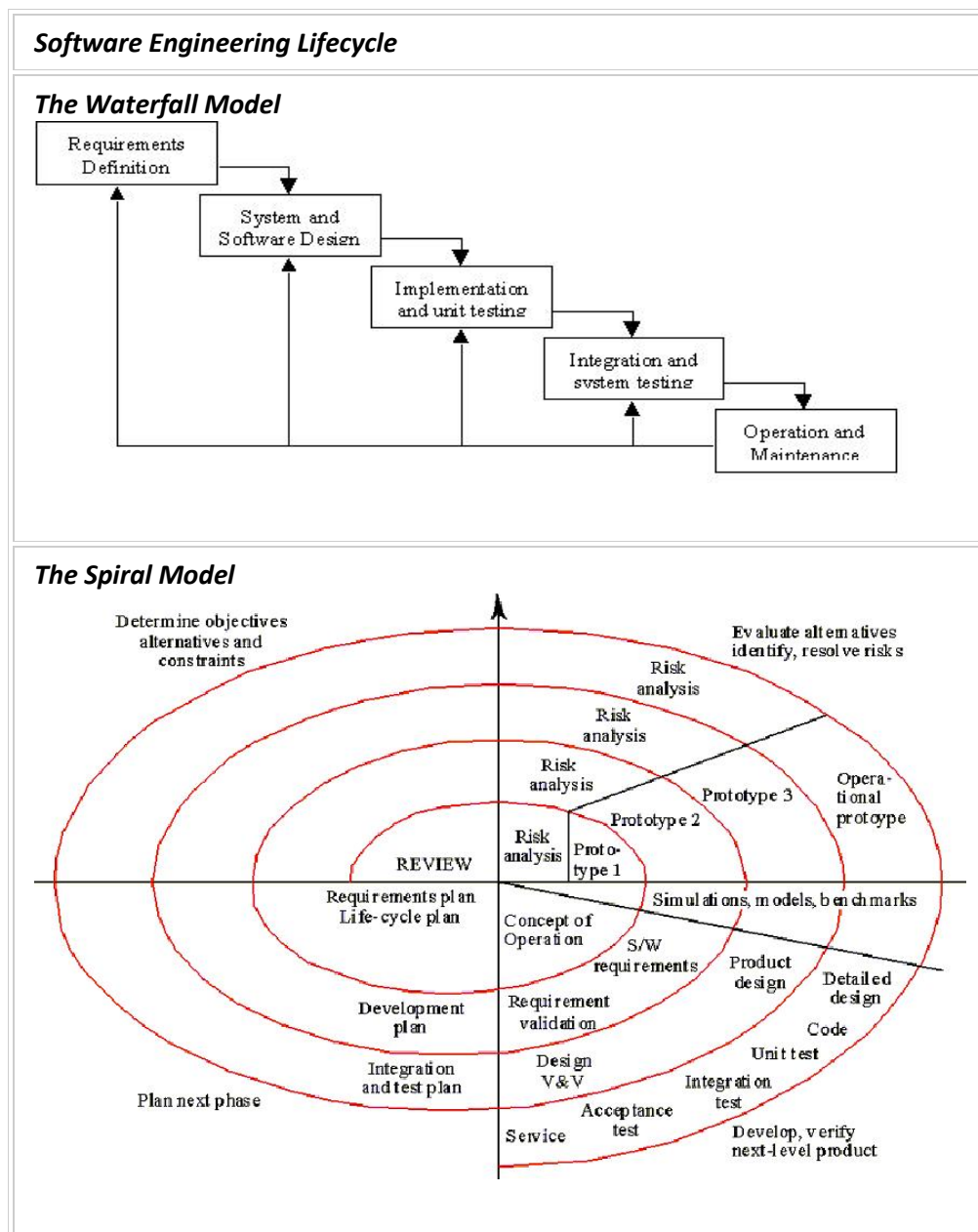
Early programming languages were '**low level**' or assembly language – an example could be;  
 ADD X Y Z

In Natural Language this could be "add the value in memory location x to the value in memory location y and store the result in memory location z". This would then be converted by the computer into a binary instruction similar to the one we encountered before. '**High level**' languages (such as C++) use much more sophisticated instructions, and then use a '**compiler**' to translate the instructions into machine language.

## 7.4 Problem Solving

A computer programmer is responsible for writing computer programs that is a series of instructions written in a programming language, which a computer can follow to perform a task. When learning their first programming language, many students believe that learning how to translate their ideas into the language a computer can understand is the difficult part of problem solving... **this is NOT true!** Learning to use a programming language, is **not** like learning to use a different natural language, so you shouldn't approach it in the same way that you did when learning English, (or Thai, Japanese, etc.) A good programmer is one who understands how to solve a problem - one who has a *method of solution*.

If you understand how to solve a problem, it is then relatively straight forward to translate your solution into a series of instructions, in C++, or any other programming language.



There are more than one approach to developing software, as indicated in the software development lifecycle models above. We will return to discuss the waterfall model in more detail, but for now, the first step, in any project, when considering developing a computer program is to ensure that you understand the nature of the problem (*Requirements Engineering*). This may sound obvious, but many software projects have failed due to incomplete problem understanding. Once you understand what your program needs to do you can consider how you could solve the problem(s) you are faced with.

### **BREAK IT DOWN!!!**



“Break your problem down into smaller more manageable chunks!”

Often problem solving involves developing an **algorithm**, or sequence of precise instructions, which can be followed to produce the intended result. This is breaking the problem down – taking the whole problem, and breaking it down into various parts which can be more easily managed.

### **7.5 Exercise**



Consider the problem;  
“I need to go to Bangkok, today, but currently I am in my chair in the classroom – how can I get there?”

Let’s break this down and develop an algorithm to solve this problem.

### **7.6 Key Questions**

- 1) What is the difference between system software and application software?
- 2) What is a compiler?
- 3) What is the difference between a high-level and a low-level programming language?

## Chapter 8

### Software Development Lifecycle (SDLC)

#### 8.0 The importance of Software

Software is one of the components in information systems, so it is very important to understand what makes good software, and the approaches to developing good software. The economies of all developed nations are dependent on software as more and more systems are now software controlled. A significant portion of GDP is spent on software production and it is the software cost that often dominates the system costs, more than the hardware costs. It is worth noting that software often needs to have a long lifetime, requiring maintenance throughout its life. The costs during maintenance may be significantly more than the initial development costs.

The attributes of good software include the following;

- *Maintainability* – It is important for developers that the software is maintainable. Software is often intended to be used for a long time, during which the environment that it works in is likely to evolve, and the software will need to react to those changes. Software that is well designed and structured well is more likely to be easier to maintain and have a longer life time.
- *Dependability* – Good software can be trusted by its users to perform the way it is intended to. There are several dimensions to dependability including reliability, availability, security and safety.
- *Efficiency* – Even with increasing memory capacity and improving processing capability, it is essential to build software that makes effective use of system resources. Time is included in system resources, and efficient algorithms are needed to perform quickly.
- *Usability* – Software is used by a variety of different users, each with their own skills, experience and abilities. Usability measures whether the software is appropriately usable by the intended users.

#### 8.1 Introducing Software Engineering

Software Engineering is a discipline that is concerned with effective software production. The field investigates systematic / organized approaches to developing software, using appropriate tools, techniques and methods. The key objectives for software developers are to create software that does the right thing, within the development constraints and allocated resources, both in terms of time and money.

A *Software Process* is a set of activities involved in the development (or evolution) of software. There are 4 generic activities involved in all software processes;

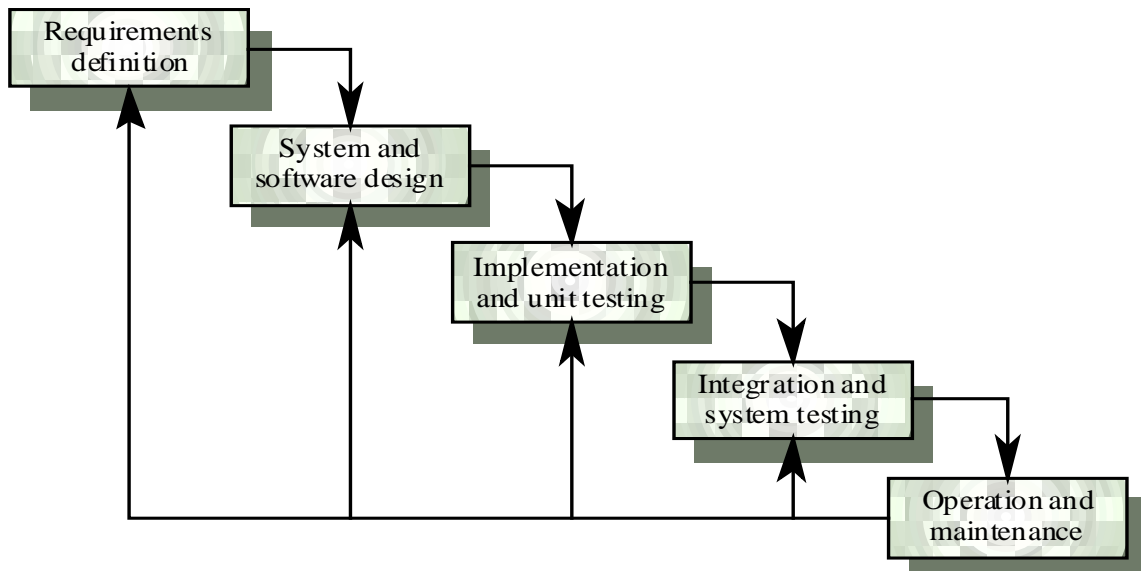
- *Specification* – clearly defining what the system should do, and what constraints there are on its development
- *Development* – the phase where the software is designed and implemented

- Validation – a phase of confirming whether the software meets what the customer wanted, i.e. testing
- Evolution – as the demands change over time, the software needs to change also

*Software Process Models* depict the way these four activities interrelate for a particular project. There are various generic software processes, and many projects are a variation of a generic approach. In the following sections we'll take a closer look at some of the more common approaches.

## 8.2 The Waterfall Model

The waterfall separates each of the phases (particularly specification and development) into distinct phases.



The 5 phases in the waterfall model are;

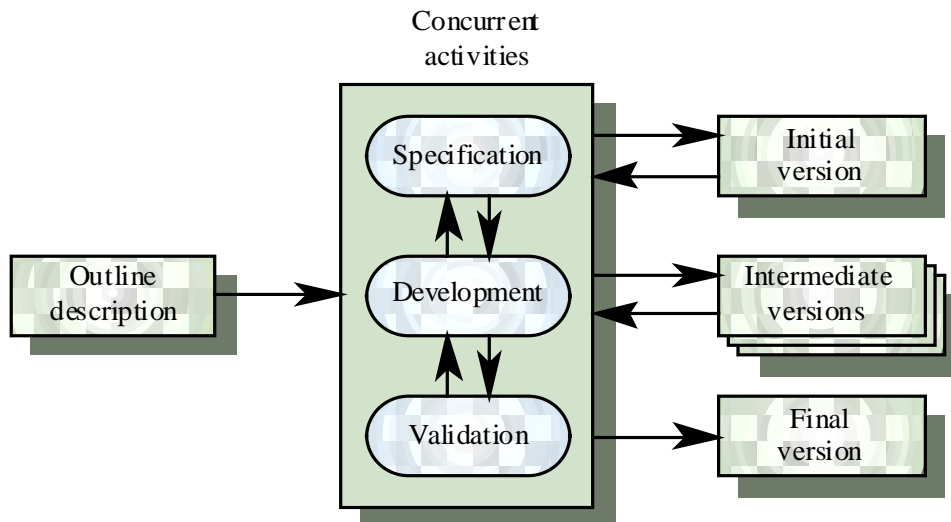
- *Requirements Definition* – the specification phase where the needs of the customer are analysed and defined, being then written (or specified) clearly and unambiguously.
- *System and Software Design* – in this phase the system is divided into hardware and software components, and then the software is designed, beginning with the system architecture and proceeding on to designing data components and software algorithms.
- *Implementation and Unit Testing* – the implementation phase is where the software coders take the design and realize it in code. Since the coding process generally involves finding and resolving bugs, this phase also includes the testing of each piece of software.
- *Integration and System Testing* – once all system units have been developed and tested a further phase of testing is needed to test how the units work when integrated together and testing the full system within its intended environment.
- *Operation and Maintenance* – once the system is ready users can be trained and it can go into operation. As mentioned previously it is likely to undergo a long period of maintenance, including corrective, perfective and adaptive maintenance.

The waterfall model is a popular approach, particularly with large projects. Its origins come from other rigorous engineering disciplines and is similar to the approach to building a house or a bridge. The distinct phases ensure that the problem is fully understood before time is spent, and potentially wasted on development. However, software is very different from a building and so software engineering is a different discipline to other engineering disciplines, so over time drawbacks to the waterfall model have been identified, primarily the difficulty in accommodating change once the process is underway. In other words, once we have moved on from one phase, it is difficult to revisit it – once the requirements have been specified and ‘frozen’, it is hard to allow changes to those requirements. If the software development lifecycle was to take many months, or even years, it would be difficult to react to an environment which had undoubtedly changed.

Therefore the waterfall model is only really appropriate in environments where the requirements are clear and well understood. Nonetheless, the model remains popular, often because project managers, and customers, are comfortable and familiar with having clearly defined milestones and deliverables.

### 8.3 Evolutionary Development

The goal of evolutionary development is to adapt a system until it meets the specification, in acknowledgement that the environment of a system is continually changing and that it is often difficult to understand the needs of customers, particularly not without some element of prototyping.



In evolutionary development the specification, development and validation phases are seen as concurrent, i.e. happening at the same time. It is important however to begin with a clear outline description, from which a general architecture can be decided upon. Then, as more components are specified, developed and validated, different versions of the software can be delivered until a final version is complete.

We can identify two different types of evolutionary development; Exploratory Development and Throw Away Prototyping.

In *Exploratory Development* the goal is to begin with an initial outline and partial solution, gradually adding elements, working alongside the customer, until the needs are met. In this case the best understood requirements are chosen first, these can then be satisfied while other needs are still being explored and specified.

In *Throw Away Prototyping* the objective is not to immediately start creating the final solution, the goal is simply to better understand the problem or system requirements. In this case the poorly understood requirements should be investigated first with the goal of discovering whether or not the analyst's understanding was correct.

As with the waterfall approach, evolutionary development is not a panacea, and there are problems with it. These include the lack of process visibility, which makes it harder to project manage and harder to know when the project will be completed. Another issue is that systems often end up being poorly structured – without knowing exactly what the project is trying to achieve, early decisions about system architecture could be flawed. Software which is constantly maintained, or has parts added and removed is inevitably harder to maintain in the future. Special skills and experience may be needed for this approach, for example using rapid prototyping languages.

Therefore evolutionary development is best suited for small, or medium sized interactive systems – for larger systems the waterfall approach may be better suited. It is very difficult to define a user interface without some element of prototyping, therefore the evolutionary approach is more suitable for interactive systems where the user interface is key, and is often used for just the user interface part of larger systems. Evolutionary development is also useful in short-lifetime systems, where the software may not need to be maintained long into the future, and a working version of the software is needed quickly.

## **8.4 Incremental Development**

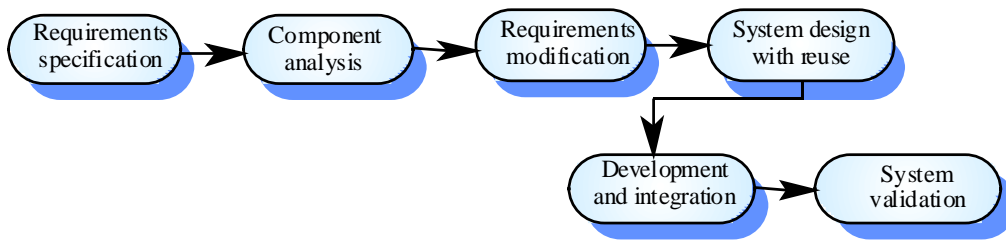
*Incremental Development* is when software is delivered in phases, or increments. This can be applied to either of the approaches discussed so far and has some advantages, including reducing the risk of overall project failure. Some system functionality is delivered early and can be used and tested in operation while other parts of the system is being developed. This means core parts of the system are tested more thoroughly, analysts can confirm they understood the requirements correctly, while the early version can also be used to elicit further requirements.

Modern approaches include *Extreme Programming* where development involves fast cycles delivering very small increments of functionality.

## **8.5 Reuse Based Development**

Reuse based development is where systems are built largely from integrating components that have already been developed. These components could be small bits of functionality, or could be whole sub-systems – for example COTS (Commercial Off The Shelf) solutions.

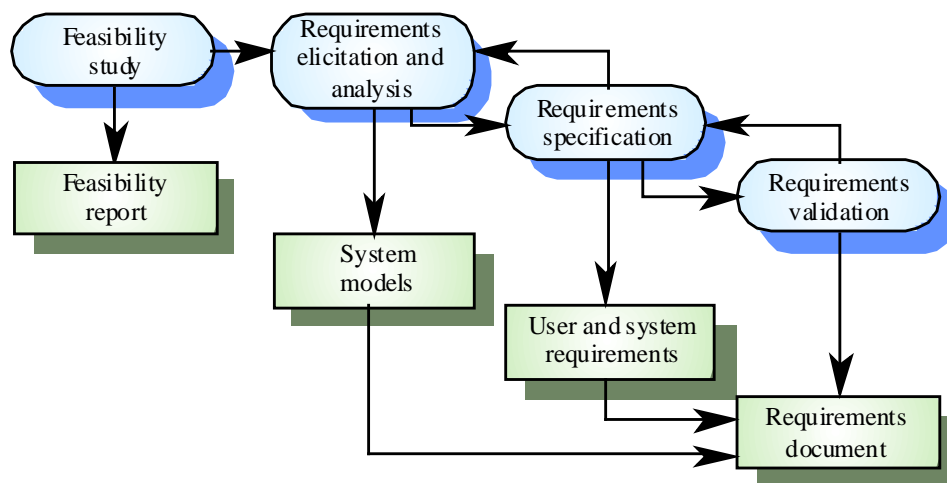




The Reuse based development process involves the phases of component analysis and requirements modification after the specification stage. Once we understand the problem, existing solutions can be looked for and assessed for their suitability. If suitable components can be found, the requirements can be adapted (or modified) to incorporate the existing components. Often customers would be happy with a solution that meets most of their needs and is available immediately, rather than a solution that they will have to wait to be developed. A further benefit of reusing components is that they have generally already been tested in a working environment.

### 8.6 The Requirements Process

The first phase in any software process model is the requirements process, often referred to as requirements specification. The requirements process can be broken down and illustrated as in the following process model involving 4 phases (Feasibility Study, Requirements Elicitation & Analysis, Requirements Specification and Requirements Validation) and 4 deliverables (Feasibility Report, System Models, User & System Requirements and finally the Requirements Document).



#### Feasibility Study

Feasibility is related to possibility – a test as to whether something is possible or not, but a better definition would include whether we are capable of doing it or not. A feasibility test however goes beyond whether something is possible or whether we are capable of it to test whether it is worth our while doing it given our resources and the risks of doing it. A feasibility study as the first step in the software engineering process is a good idea, to decide whether to go ahead with the project or not, before investing too many resources in it. There are quite a few considerations that go into making the ‘go or no go’ decision, and it is difficult to answer it without understanding at least some of the problem. Some of the things that would need to go into a feasibility report include;

- *Scope* – How big is the project? Which systems / products will the project replace? Which other systems will the project interrelate with?
- *Stakeholders* – Stakeholders (simply) are holders of stake! In other words they are anyone who has an interest in the success of a project. They could be managers, employees, customers, the client, users...
- *Cost / Benefit Analysis* – To decide if a project is worthwhile we need to know whether the benefits will outweigh the costs. However, it is difficult to predict both costs and benefits. While some costs are obvious, some are hidden – for example, the cost of ‘wasted’ manager time spent in meetings about the project. Some costs and benefits are unquantifiable – such as the improved motivation and happiness of employees. Also many analysts are too inexperienced to assess costs and benefits – many projects end up over time or over budget because of failures at this initial phase.
- *Risk Assessment* – The risks of doing the project need to be assessed. This includes identifying risks, and then assessing how likely they are to occur and how damaging they would be if they did occur. A strategy then needs to be formed for each risk to either reduce the probability or the effects of the risk occurring.

### *Requirements Elicitation and Analysis*

Once the project is given the go ahead, the next phase is to elicit the requirements. *Elicit* means “to evoke or draw out through questioning”, and is carefully chosen to describe the process of ‘getting’ requirements. Unfortunately it isn’t possible to just go and pick up the needs of the customer, so requirements need to be gathered, collected, extracted and trawled for by a variety of different means. Requirements could come from the stakeholders (already identified), existing systems, legal documents or competing systems.

- *Interviews* – A logical place to start would be by asking people what they need in an interview. However, often people don’t know what they want, or are unable to explain what they need. Also, different stakeholders will have a different viewpoint of a system, and the different viewpoints may well conflict. A further issue is that people are often busy, particularly if a systems analyst wants to meet with them, so finding a time to conduct interviews may be problematic.
- *Facilitated Meetings* – If people are likely to have conflicting needs, a good approach is to get the different people together for a meeting to discuss and resolve these conflicts. It is the analyst’s job to facilitate the meeting, as many of the same challenges exist as with interviews. Here also it is likely that nobody wants to argue with the senior figure and will simply go along with their opinion whether or not it is correct.
- *Ethnography* – Otherwise known as *observation*, ethnography involves watching people while they work, after all if stakeholders have difficulty in expressing their work, it might be easier to watch them doing it. Challenges here are that sociologists are better at ethnography than computer scientists, but they may not be as skilled at writing concise, unambiguous summaries. A further issue is that people tend to do things differently when they know they are being watched.
- *Scenarios* – A solid approach in all of these cases is to propose scenarios to the stakeholder asking them how the system should react in different situations.

However it is difficult to be comprehensive and discuss every possibility, and also difficult to assess which are the important scenarios.

- *Prototypes* – As has been discussed already, another good way of drawing out requirements is to build a prototype and check whether the understanding is clear. As prototypes are particularly useful for developing the user interface, there is a tendency for stakeholders to be carried away by the interface, i.e. how the system looks rather than what it does.

There are many other ways of eliciting requirements, the key is to use a variety of approaches, and then build some models to act as a visual guide for system processes.

### *Requirements Specification*

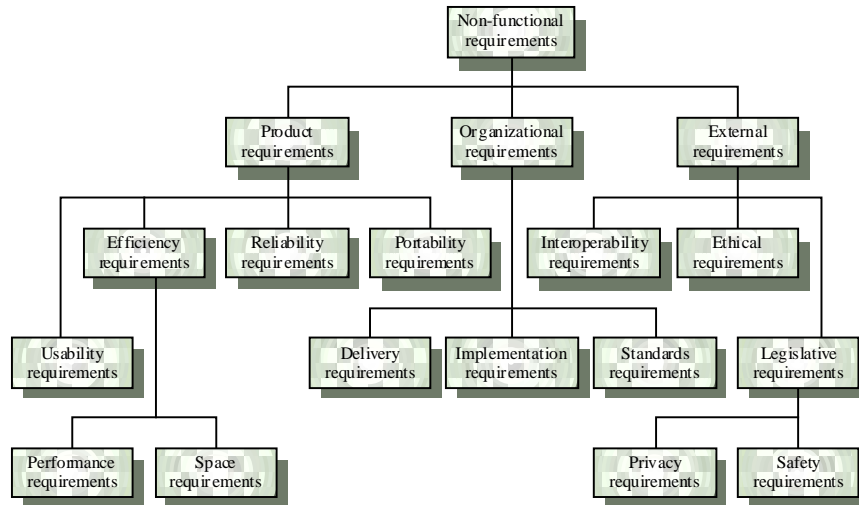
Once requirements have been suitably collected, they need to be appropriately specified, or documented. Writing requirements is also not trivial. A good requirements specification will be;

- *Complete* – not missing any requirements
- *Unambiguous* – so that all readers will share exactly the same understanding
- *Measurable* – so that we can know when the requirement has been satisfied
- *Consistent* – so there are no conflicts within the document
- *Precise...*

Often a requirements specification is divided into sections, and while there are different choices for how to structure the document, it should include both functional and non-functional requirements.

- *Functional Requirements* – these describe the things that the product must DO, i.e. functions that the system is performing. For example: The system should produce a daily sales report.
- *Non-Functional Requirements* – these are properties or qualities of the system, i.e. not things it should do, but how well it should do it. For example: Registered users should be authenticated within 0.25 seconds.

There are several classes of non-functional requirements, and it is worth highlighting that these are just as important as the functional requirements. Users will reject a system even if it performs all the tasks that it needs to, if it doesn't perform them quickly enough, or accurately enough. Here are some of the classes of Non-Functional requirement.

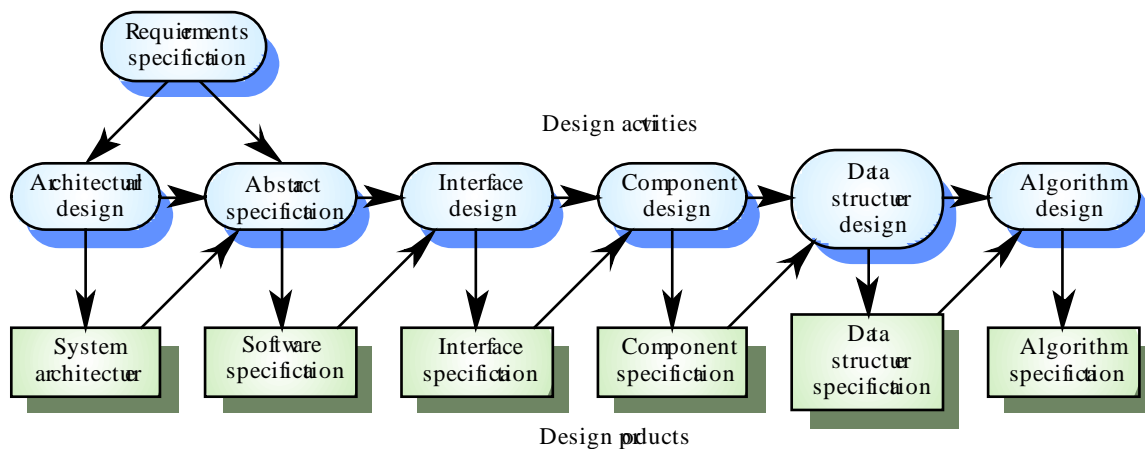


### Requirements Validation

Before we can go ahead to the design and development phases, particularly if following the waterfall model, we need to validate the requirements. Do we have all the requirements? Are they all testable? What appropriate acceptance tests could we use?

## 8.7 Software Development

The process of converting a comprehensive specification into an executable system involves the design and implementation phases. While there may be overlaps between these phases, it is worth making some effort to ensure a good design that will realize the specification. This phase will involve several design activities, beginning with the Architectural Design. Once a logical initial architecture has been selected, the problem can be broken down into appropriate components, with the interfaces between these components being clearly specified. Within each component, appropriate data structures and algorithms are also designed. Often the design phase involves creating models to depict how the software should work.



## 8.8 Key Questions

- 1) What are the attributes of good software?
- 2) What are the key generic software processes?
- 3) Compare the waterfall model with evolutionary development.
- 4) What are the differences between throwaway prototyping and exploratory development?
- 5) What benefits can be gained from reuse based development?
- 6) What are Functional and Non-Functional Requirements?
- 7) How could you go about eliciting requirements?
- 8) What are the key phases in designing a new system?

## Chapter 9

### Data

#### 9.0 Characteristics of Good Data

There is a famous adage that states “Garbage In – Garbage Out”, which means if you input poor data then you will get poor information out. Therefore the characteristics of good data include;

- Accurate
- Verifiable
- Timely
- Organised
- Accessible
- Useful
- Cost-Effective

#### 9.1 File Processing Systems

One option for storing data is in a file processing system, where data is stored in files. There are some challenges to storing data in files;

- *Data Redundancy* – Storing data in files leads to each department having their own files meaning the same data may exist in more than one place. Duplicating data can be a benefit, as it can serve as a back-up in case that data is lost, however it can also be a hindrance as it wastes space and creates problems when trying to maintain consistency across all copies – for example when someone changes their name they need to update their name in multiple locations.
- *Isolated Data* – Storing data in files means that data isn’t always relatable in convenient ways and accessing some information becomes more complicated. For instance if we store all student information, such as name, email address, etc. in one file, and in another a list of students registered for a particular course, in this case getting a list of email addresses for that particular course could be tricky – unless we duplicate the data.

Many systems today use relational databases to store data and use a DBMS (Database Management System) to help manage the data.

#### 9.2 Database Management Systems

Database Management Systems are an effective way of managing the data within a system, or collection of systems. There are several different DBMS’s all with the objective of providing convenient ways to maintain and retrieve data from a database – normally through a query language. SQL or Structured Query Language is a common method for accessing the data, i.e. to INSERT, UPDATE, SELECT or DELETE data from the database. The DBMS also handles security to ensure that only authorize users can access data at permitted times, as well as performing routine tasks such as back-ups and recovery.

Data in a database is stored in a series of layers, we could call the hierarchy of data;

- *Characters* – singles bytes of data, such as a single letter or number.
- *Fields* – a combination of related characters, such as someone’s name. Fields are given a name, a type and a size, in much the same way as variables.
- *Records* – a combination of related fields, such as all the data about a person. Records have a primary key which can uniquely identify each record.
- *File* – a collection of related records, such as all the data about all the people in an organization.

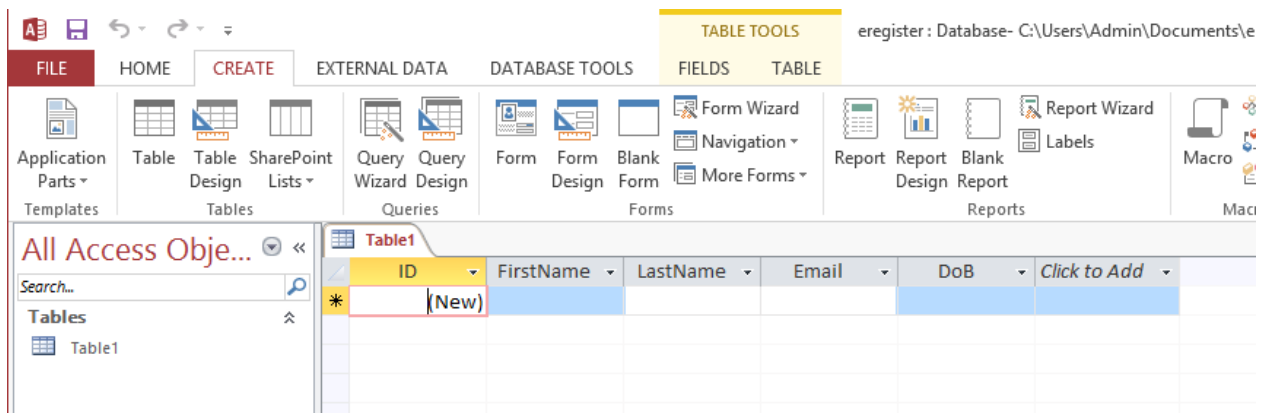
Relational databases offer a convenient way to manage relationships between data, i.e. the data about a student is related to the data about the courses they are studying and their advisor’s data, while the teacher’s data is related to the courses they are teaching. This simplifies the way the querying language can make more sophisticated queries, and also removes the need for duplicate data.

In a relational database, data is stored in tables, where each column represents a field (or a particular piece of data), and each row stores the data about a particular entity (or record) – much like in an excel spreadsheet. Each table will have a primary key, normally an ID, which uniquely identifies each record. This primary key can then be used as a foreign key in a different table to relate the data. For instance, in the table storing data about the grades for a course, the id for each student can be used as a reference.

### 9.3 Microsoft Access – Quick Tutorial

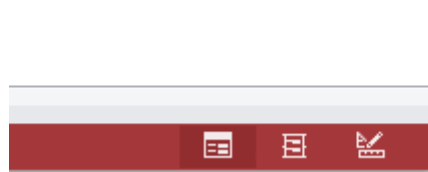
Access is part of the office suite, and gives us a way of creating databases on the desktop – we’ll use it to create a simple database to record student grades in various courses. For this we will need 3 tables – a table to store courses, a table to store student information and a table to store the grades.

Let’s start with the student table – what information do you need to store about each student? FirstName? LastName? Email? DoB? Open up Microsoft Access, and create a new blank desktop database. You will then be able to create the student table. The ID field is created by default, so by clicking on the next columns you can add the other fields you need, also selecting the type of data to be stored.



Save this table (as Student). The next step will be to create a form to allow a user to input the student’s details. Select ‘Create’ and ‘Form’, and a default form will be created for you. By using the

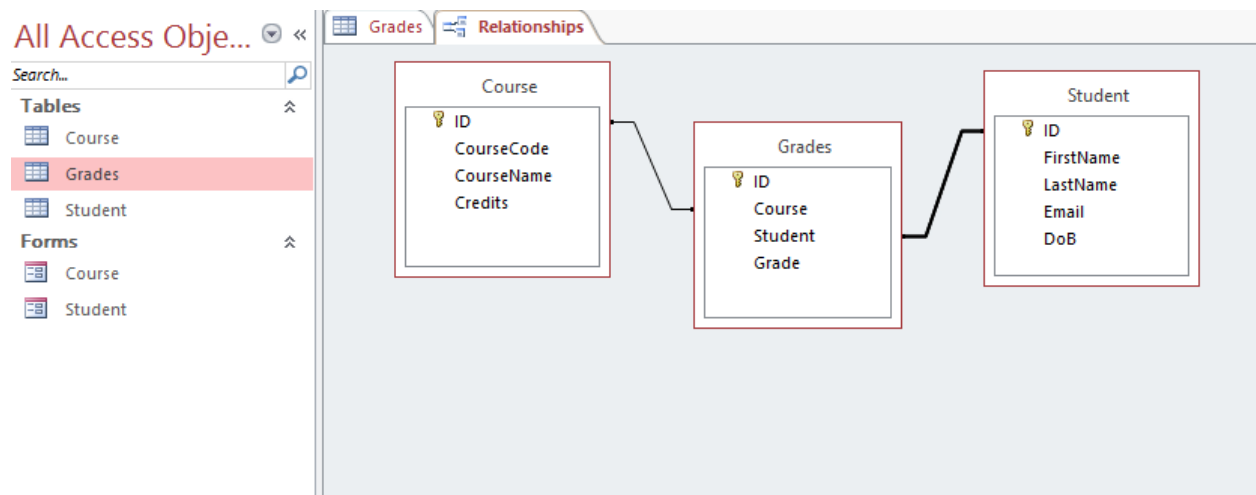
buttons in the bottom right corner, you can switch between different views of the form, and use the 'Design View' to make your form look nicer! You will probably want to remove the 'ID' field!



Save the form, and then you can create some sample data using the 'Form View'. You can check that the data is now in the table by going back to the table and selecting 'Refresh All' from the Home Menu.

Great! You should now have a form for storing student information, and some sample data. Go through the same process to create a new table for Course data – what fields are needed? CourseCode? CourseName? Credits? Create some sample data again.

As mentioned before, where databases get more useful is by being able to link data - here we want to link students with courses, and store their grades. Create the third table with the columns; Course, Student and Grade. This time when you choose the type for the fields 'Course' and 'Student', select 'Lookup & Relationship', and work through the simple lookup wizard to select how the user will choose the course / student information. Grade should be a simple number type. You can check that you have successfully added the "Foreign Keys" by choosing 'Table Tools' and 'Relationships' and you should get a graphic similar to the one below.



By now you should have 3 tables and 2 forms, so next up we need to create the form for the Grades table – the same as we did above, but notice this time you get a drop down box allowing you to choose which course and which student. Create some dummy data again.

### Queries & Reports

Having successfully got data into the database, the next step is to build queries and get data back out of the database – the right data! The first step here is to build a query, so let's suppose we want to generate a report of the grades for a particular course – choose 'Create' and 'Query Design', and when prompted add all 3 tables – you will again be shown the relationships between the tables as



above. Select which fields you want involved in the query, by double clicking on them – they will then appear in the table below. For this we will need the course code from the Course Table, the FirstName and LastName from the Student table and the Grade from the Grades table.

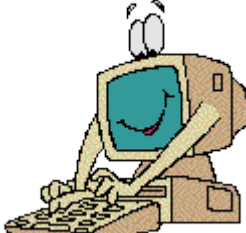
For this query we only want to get results for a particular course, so add the course code you are interested in to the 'Criteria' Field – as shown in the screen dump below.

Field:	CourseCode	FirstName	LastName	Grade
Table:	Course	Student	Student	Grades
Sort:				
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Criteria:	123456			
or:				

Now you have your query you can save it, and run it – you should be presented with the results you want!

Next up, let's build a report of the results generated from the query. Use the Report Wizard to help you choose the query information – and be sure to add summary data!

### 9.4 Exercise



Extend your E-Registration program to create a report for each student's grades.

## 9.5 Key Questions

- 1) What are the characteristics of good data?
- 2) Explain the problems of data redundancy and data isolations.
- 3) What is SQL?
- 4) What is the hierarchy of data?
- 5) What are the benefits of using a relational database?

## Chapter 10

### People

#### 10.0 Project Blast Off

Previously we've discussed the feasibility study as the start of a project. As a project gets approval and is feasible, it gets launched, kicked off or initiated – i.e. the project gets started. The start of a project normally involves a blast off meeting where the key stakeholders are brought together to discuss and define the project.

*“The loftier the building, the deeper the foundation must be laid” (Kempis)*  
(in other words, the bigger the project, the more important preparation is)

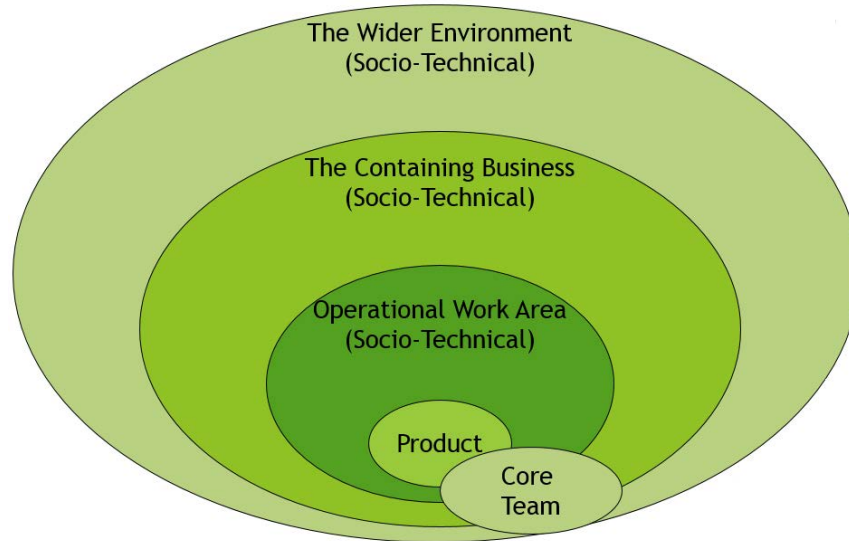
The deliverables from Project Blast Off include the Goals, Scope, Stakeholders, Constraints, Cost Estimations and Risk Analysis. There is a causal relationship between the Goals, Scope and Stakeholders.

- *Goals* – a short statement explaining, from a high level, what the product is intended to do and what advantages it will bring to the organization; i.e. the justification for doing the project.
- *Scope* – Once the goals are decided, the scope can be outlined. Scope is the business area affected by the product, and what won't be affected. The system will work alongside other systems, so which other systems will it interrelate with. The scope defines what work will be studied, and what won't be.
- *Stakeholders* – Once the scope of the project is set, then stakeholders can be identified. Stakeholders are any people who have an interest in the success of the project, or have an effect on the product.

Once the Stakeholders have been identified they may well bring new goals, and with new goals the scope of the project may change. If the scope of the project changes, then new stakeholders may be identified, with new goals. This cycle can be a useful part of identifying the bounds of a worthwhile project to proceed with, and useful for identifying the key stakeholders.

#### 10.1 Stakeholders

The important stakeholders need to be identified and invited to be part of the project team. The *Onion Diagram* can be helpful in identifying different kinds of stakeholders. The diagram shows a series of rings surrounding the product at the core. First there is the operational work area, a socio technical system which will include the people who will actually use the system in their daily lives. Next is the containing business, which will include people who also work for the same organization, but may not need to use the system themselves, perhaps a department manager. Outside of this is the wider environment, which might include any other people with an interest in the project, perhaps community groups. A socio-technical system is one which includes both social aspects and technical aspects. The core project team may well include stakeholders from all the rings of the onion.



The most important stakeholders include;

- *The Client* – the client is the person who pays for the project, while it is being developed. While people are working on a project, someone needs to pay for their time – therefore it is important to keep the client happy!
- *The Customer* – the customer buys a product once it is complete, i.e. they will go to Pantip and walk out with your system. Therefore we may not know who the customers are, but it is important to find representatives of potential customers to be part of the project.
- *The Users* – the users are those who will operate the system. There are many usability issues to consider, for example the experience level of the users, their intellectual capabilities, linguistic skills and perhaps disabilities.

There may well be overlaps between these three stakeholders, but all should be considered as separate roles. Other stakeholders include domain experts, negative stakeholders, cultural experts, technical experts and adjacent systems. It is up to the project manager to put together a team of stakeholders, analysts, designers and developers to successfully complete the project.

## 10.2 Assignment – Belbin

Belbin devised a way of identifying different team roles that people perform when working as a team. Complete the in class exercise to find out which team role you are most likely to be. Belbin's team roles are as follows;

- *Implementer (IM)* – Strong sense of organizational duty and efforts.
- *Co-ordinator (CO)* – Calm, strong and realistic
- *Plant (PL)* – Clever and innovative
- *Resource Investigator (RI)* – Ideas gathering and energetic
- *Shaper (SH)* – Leader, challenger with high need for achievement
- *Monitor-Evaluator (ME)* – Serious, proud, critical thinker
- *Team Worker (TW)* – Social, trusting and sensitive

- *Completer-Finisher (CF)* – Finisher, perfectionist
- *Specialist (SP)* – Individual, specialized in certain subjects

This is an individual assignment. Write a self-reflection paper to answer the following questions. While it is an individual assignment, you are encouraged to discuss the answers to part B with the other members of your team.

- Part A – Consider the team roles that you scored most highly on.
  - Which of Belbin’s Team Roles did you score most highly on?
  - Do you feel that the ‘Team Role Description’ for your role appropriately describes the kind of roles that you perform when working in a team?
    - In which ways does it describe you best?
    - In which ways does it not reflect on your approach?
    - How could being aware about your team role affect the way you work in a team in the future?
- Part B - Consider the team activity in class, and the other members of your team.
  - Do the team role descriptions appropriately describe the roles that the members of your team played?
  - How well did you work together as a team?
  - What would YOU do differently in the future to help your team be more successful?

### 10.3 Key Questions

- 1) What is the relationship between Goals, Scope and Stakeholders?
- 2) Who is the Client?
- 3) Who is the Customer?

# Chapter 11

## Dependability

### 11.0 Dependability

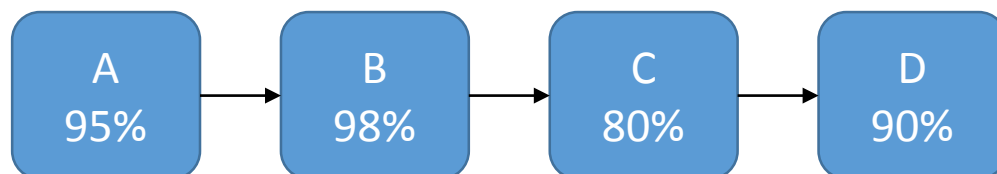
A key challenge that is continually affecting information systems is how to make them dependable. Dependability reflects the degree of trust users have that the system will perform the way it is intended to. There are 4 key dimensions of dependability;

- Availability
- Reliability
- Security
- Safety

Availability is often subsumed under reliability, as clearly if a system is unavailable, then it is unreliable. While there are system that can be unreliable so long as they are available, the issues concerning availability are largely similar to those concerning reliability. Safety critical systems are systems that could kill or seriously hurt people if they have some problem, for example the systems in an aircraft.

### 11.1 Reliability

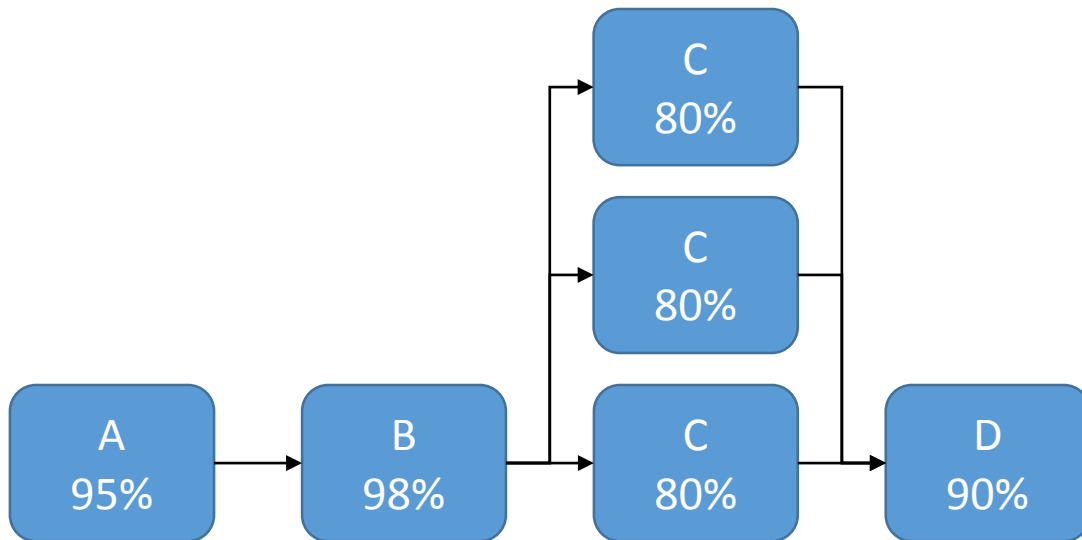
Reliability is the probability that a system will operate as it was intended, reflecting a user's trust in the system working. The key to improving reliability is redundancy, i.e. having spare components that are running in parallel. If we consider a system with 4 components working in series (perhaps a PC, a modem, a router and a webserver). Suppose each as the reliability indicated below;



In this case the reliability of the whole system working correctly, relies on all of the components working correctly. Therefore we can calculate the reliability of the whole system as;

$$0.95 * 0.98 * 0.8 * 0.9 = 0.67 \text{ (or 67\%)}$$

To improve the reliability of the whole system we need back up or spare components, i.e. if my computer was to break, I would simply use an old computer or move to the lab, therefore the reliability of the system is really dependent on one of the computers working. Suppose I have 3 computers, working in parallel, and each has reliability of 80%, then the probability of failure for each is 20%, and the probability of all 3 failing is  $0.2 * 0.2 * 0.2 = 0.008$  (or 0.8%). Therefore the reliability of the 3 components is 99.2%.



The reliability of the whole system can then be recalculated as;

$$0.95 * 0.98 * 0.992 * 0.9 = 0.83 \text{ (or 83\%)}$$

By implementing the 2 back up components the reliability of the system has increased by 16%. To continue improving the reliability of the system further back-ups can be implemented, each time selecting the least reliable component (now D), and adding a redundant back-up. Further examples of redundancy include having a redundant power supply – such as a UPS (Uninterruptable Power Supply), or having a redundant internet connection by connecting to multiple ISPs.

Having 3 components running in parallel is an example of “Triple Modular Redundancy”, which has some further benefits. An old adage suggests that when going to see either take 1 watch or 3 watches, but not 2 watches. The reason is because if you only have 2 watches and one fails, you still wouldn’t know what the time was, as you wouldn’t know which watch had failed. Having a 3<sup>rd</sup> component allows you to compare the results and determine which component is faulty. This works particularly well with hardware components where faults are likely to be random, unlike software where copying software will copy any faults in it.

## 11.2 Security

As more important data is stored digitally, and transmitted through public networks, there are increasing security threats, including;

- Unauthorised Access
- Viruses / Worms / Spyware
- Tapping
- Spoofing / Sniffing
- Identity Theft / Phishing
- Theft / Fraud
- Hacking
- Vandalism

- DDoS Attacks

Fortunately as new security threats are identified, new strategies are developed to counter the threats, including;

- Tight Security Policies
- Access Control Lists
- Authentication
  - Passwords
  - Biometrics
- Firewalls
- Anti-Virus
- Encryption

### **11.3 Key Questions**

- 1) What are the 4 dimensions of dependability?
- 2) Explain how you would go about improving the reliability of an information system.
- 3) What is Triple Modular Redundancy?
- 4) Identify 5 threats to an information system and explain how you could counter these threats.



## Chapter 12

### Ethics

#### 12.0 Ethical Questions

As information systems have taken increasingly important roles in our lives, it has created a number of ethical questions about how to 'do the right thing'. For example, computerizing some business processes have led to people losing their jobs as their roles are replaced through automation. Finding fair ways to reduce a workforce is challenging, yet sometimes the boring, mundane tasks are replaced with more creative tasks which improves the quality of life for those who keep their roles. Similarly products may then be created with higher quality and lower cost, providing better choices for everyone in society. When managing the use of technology within an organization, managers will need to answer questions such as;

- Should you monitor your employee's emails?
- Should employees be allowed to use work computers for private purposes?
- Should they be allowed to take copies of software home?
- Should you keep electronic copies of employee's personal records?
- Should you sell your customer's information?

While the answer to some these questions is a legal issue, for some it is an ethical dilemma. How can we weigh up the right to privacy of our employees, against the desire to monitor them for effective management? Is it possible to treat people as individuals or should all employees be given the same restrictions?

#### 12.1 Business Ethics Theory

There are several ways to look at these ethical questions, and when making the decision we could look at it from different perspectives.

- *Stockholder Theory* – Managers are agents of the stockholders, with the ethical responsibility to them to increase profits without breaking the law.
- *Social Contract Theory* – Companies have an ethical responsibility to all members of society.
- *Stakeholder Theory* – Managers should manage for the benefit of all stakeholders, shareholders, customers, suppliers, local communities, employees, etc.

Clearly Stockholder theory is at one end of a spectrum, where the only consideration should be maximizing profits, while at the other end of the spectrum is Social Contract theory where everyone becomes a consideration. As with most ethical questions, there isn't necessarily a correct way of finding a solution.

#### 12.2 Media

This week's video can be found here:-

Does what happens on the facebook stay on the facebook.  
<http://www.youtube.com/watch?v=wogtTQs8Kzw>