**UNIT I**

**Introduction**

The increasing use of computers in the last few decades has seen the ascendance of information technology to a position of prevalence in human affairs. In our homes & offices today we have more computers than there are people who live & work. In this semester we shall look into how & what software contributed to this growth and as well its relationship to Open Standards.

We have already understood Open Standard and Open Source in the previous semester. To recap, open standards promote…..

* Interoperability
* Prevents vendor lock-in
* Flexibility
* Collaborative innovation
* Lower cost
* Freedom of action

Also, we have understood that going forward Open Source software is there to exist for long time, to promote…..

* Lower Cost of Ownership
* Quality
* Innovative reuse
* Technical competence

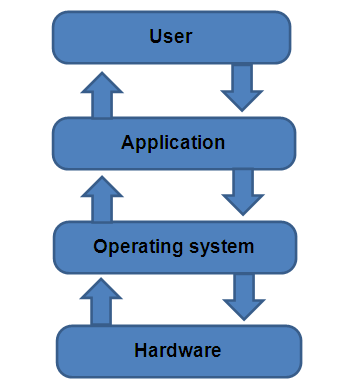
In this context, we have to remember that sharing of software for enhancements & reuse is not new, but it has been there from the days of sharing printed copies of programs along with the hardware. In the 1960’s IBM distributed the source code of mainframe operating system, Airline Control Program (ACP). All the software available was usually supplied by IBM without additional charge.

One more thing to remember is the start of software industry when hardware manufacturers unbundled software and services from hardware sales and end users had to pay for the software, which in turn brought in third party’s to develop software that works on a given piece of hardware.

In this semester, particularly we will discuss about programs (software) that make use of OS capabilities and instruct the computer, how to produce information.

Programs are the software. Software can be written in numerous programming languages. A programming language is used to express the instructions which tell the computer what to do in response to a certain action. Software could be broadly divided into 3 sections:

* **Operating systems (OS).** Communicates with the hardware. Acts as basis on which other software runs. The means by which other programs and the user interact. E.g DOS, Linux, Windows, UNIX and Macintosh.
* **Languages.** To write the software e.g.C, Smalltalk, C++, Java, JavaScript
* **Applications.** Enables the user to carry out various activities e.g. Word, Photoshop, …

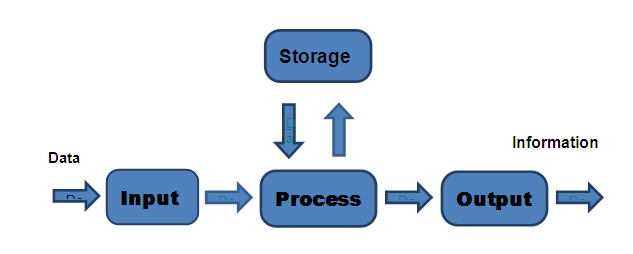


## Understanding Information Technology

### What is Information Technology (IT) ??

As defined in Information Technology Association of America (ITAA) “IT” is study, design, development, implementation, support or management of computer based information system particularly software’s applications & computer hardware.

It deals with treatment of information, and is one of the corner stones of our economy while forging a computer based society. (Others are Land, Labour, & Capital)



The term **“IT”** in its modern sense first appeared in a 1958 article published in the *Harvard Business Review*, in which authors Leavitt and Whisler commented that "the new technology does not yet have a single established name. We shall call it information technology (IT)."

Information technology (IT) has proven to be a key enabler of socioeconomic progress and development, enhancing productivity and therefore economic growth, reducing poverty and improving living standards in many ways. IT is increasingly revolutionizing production processes, access to markets, and information sources together with social interactions. IT also has an impact on government efficiency, fostering transparency and better communication and services with and to citizens.

Information technology (IT) is a defining force affecting all areas of society well into the next century, changing every institution, every business, and every individual in profound ways. Technology itself has changed dramatically in the past 15-20 years. Anticipate even more rapid change in the future, changes that impact organizations and society.

Within a short span of time we have seen better algorithms, many fold increase in computing power and hence more mobility of information.

## Shifts in computing paradigm

For some, computing today is radically different from early experiences. In fact, we have seen three major waves of computing… host-centric, client/server and network-centric.

Host-centric, or "tops-down" computing, dominated the environment for the past twenty years. The focus was on the physical enterprise, with a specific behavior pattern:

* Buy a computer system;
* Write applications;
* Define the reports needed;
* Develop fixed screens to display the results;
* Operate over a private, wire network.

We still see host-centric computing among administrative applications in some of the institutions and businesses; operate their institutional research or registration offices with a specific computer, using applications written in-house. Creation of reports is fixed by the application, a new program must be written to generate a new report.

Today a distributed client/server model is prevalent. The focus is the distributed enterprise with a different behavior pattern:

* Buy individual "client" units;
* Purchase applications;
* Use windowing to view information;
* Operate over private wire local area networks or mixed private and public switched networks.

The client/server architecture has enabled organizations to do a great deal of "mixing and matching" to suit individual needs. An example of this pattern is in purchasing client machines. People are buying CPUs separate from memory. Specific hard drive sizes are ordered. Memory modules may be different. Connectivity is a mix. Customization is the rule rather than the exception. Applications are no longer written in-house; they are purchased. Information is viewed through a windowing system, whether Windows, OS/2, Mac operating system, or something else. Users tile their windows to view what they need.

The next step is network-centric computing. The behavior pattern changes again:

* multi-source hardware platforms;
* subscriptions for software rather than purchase;
* human-centered, multi-form factors;
* public switched infrastructure, both wired and wireless.

Hardware comes from many sources. The network contains your applications as well as the data. The network is, in fact, your application. Software need no longer be purchased and installed on your computer. When you connect to the network, you access the latest version of the software for which you have a subscription.

The network’s technical characteristics will be of little concern to its users. Its presence will be assumed. The vendors and the technologies that enable service will be in the background. It will not matter what the topology is, where the server is located, or whether the connection is facilitated by wire or wireless technology. From your computer, you will have access to the resources you want and need.

In spite of its importance, the network is "dumb": It is unaware of individual computing needs and preferences. It is merely a transport vehicle. Yet, one individual’s computing needs are different than another person’s. The "personalization" of your computing interface will reside in the software.

Advances in computing technologies, such as high-resolution displays, 3-D graphics and animation, handwriting and speech input, and natural language understanding will be used to improve the end-user interface, to facilitate personal interaction and customization with computers. This will enable new interaction models, including:

* Intuitive, task-tailored interfaces;
* Virtual reality environments;
* Mobile, Hands-free, and Eyes-free use;
* Intelligent agents that will isolate users from the details of the infrastructure but will carry out tasks tailored by the user;
* and Easier searching techniques.

Personal computers are in the phase of general purpose use today. PCs are used for many purposes: word processing, electronic communications, spread sheets, graphics, multimedia, etc. The PC is highly adaptable because of the range of applications used to tailor its functionality.

Yet to obtain this functionality, users are required to purchase specific software packages as well as upgrade hardware and operating systems, and keep up with new versions of the applications. Over the next few years, the PC will move into another phase of evolution, that of being an "appliance."

The "appliance" phase is characterized by a "thin-client, fat-server" model. In this scenario, code does not permanently reside on the client, it resides on a server. When the user needs an application, it is accessed through the network and executed on the client machine.

## Application Software

ap·pli·ca·tion  
noun

1. the act of putting to a special use or purpose: the application of common sense to a problem.
2. the special use or purpose to which something is put: a technology having numerous applications never thought of by its inventors.
3. the quality of being usable for a particular purpose or in a special way; relevance: This has no application to the case.

Application software is a subclass of computer software that employs the capabilities of a computer directly and thoroughly to a task that the user wishes to perform.

This should be contrasted with system software which is involved in integrating a computer's various capabilities, but typically does not directly apply them in the performance of tasks that benefit the user.

In this context the term application refers to both the application software and its implementation. A simple, but may not be perfect analogy in the world of hardware would be the relationship of an electric light bulb (an application) to an electric power generation plant (a system).

The power plant merely generates electricity, not itself of any real use until harnessed to an application like the electric light that performs a service that benefits the user. Typical examples of software applications are word processors, spread sheets, and media players. Multiple applications bundled together as a package are sometimes referred to as an application suite. Some might bundle together a word processor, a spreadsheet, and several other discrete applications.

The separate applications in a suite usually have a user interface that has some commonality making it easier for the user to learn and use each application. And often they may have some capability to interact with each other in ways beneficial to the user.

For example, a spreadsheet might be able to be embedded in a word processor document even though it had been created in the separate spreadsheet application. User-written software tailors systems to meet the user's specific needs.

User-written software includes spreadsheet templates, word processor macros, scientific simulations, graphics and animation scripts.

Even email filters are a kind of user software. Users create this software themselves and often overlook how important it is.

In some types of embedded systems, the application software and the operating system software may be indistinguishable to the user, as in the case of software used to control a VCR, DVD player or Microwave Oven.

### List of common applications

#### Word processing

* + Microsoft Word
  + Lotus Word Pro

#### Spreadsheets

* + Microsoft Excel
  + Lotus 123

#### Databases

* Microsoft Access
  + Lotus Approach

#### Presentation

* + Microsoft PowerPoint

#### Accounts / Payroll

* + Sage

#### Web browsing

* + Microsoft Internet Explorer
  + Mozilla Firefox

#### Web authoring

* + Microsoft FrontPage

#### CAD & CAM

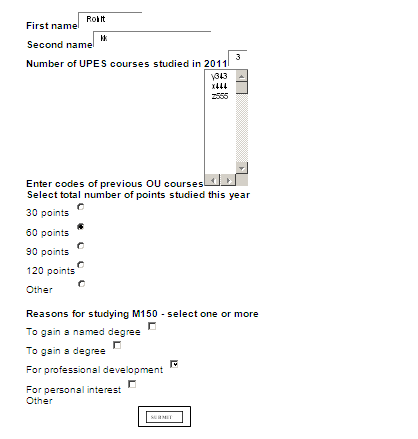
* + AutoCAD

#### System Management

* + HP Open View
  + IBM System Director

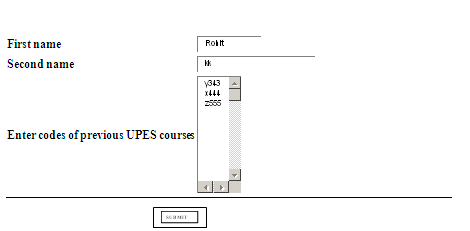
## Transforming Data into Information

Here is a form for student data.



Say, after clicking on submit button we get the following data….

**Student Data**   
First name: Rohit  
Second name: kk  
Number of courses studied in 2004: 3  
Previous courses: Y343 X444 Z555  
Number of credits studied in 2004: 60  
**Reasons for study are:** For professional development  
Here is another form ……



Think, after clicking on submit button you get this:

**Information about Rohit kk**

Rohit is a continuing student at UPES

In the case of first form we entered the data, “data collected” and when we pressed the submit button the data is “stored & operated" upon by a program; program could be a Java Script embedded in a webpage. The output simply reflected the values we entered.

With second form, the data was operated in such a way to display some **information** about a student, whether he/she is a new student at UPES. Bottom of Form

Here the “program” that is the JavaScript is called the “application software or IT application”.

### Example 2:

Another example we can look at is the smart card system where application software and hardware components require interoperability.

In today’s world, all of us probably use a credit card-sized plastic card for one reason or another almost daily. Among these are credit cards, debit cards or automatic teller machine (ATM) cards, in mass transit, for identification, not to mention cards to access buildings or specific rooms.

A smart card, typically a type of chip card, is a plastic card that contains an embedded computer chip–either a memory or microprocessor type–that stores and transacts data. This data is usually associated with either value, information, or both and is stored and processed within the card's chip. The card data is transacted via a reader that is part of a computing system. Systems that are enhanced with smart cards are in use today throughout several key applications, including healthcare, banking, entertainment, and transportation. All applications can benefit from the added features and security that smart cards provide.

The microprocessor on the smart card is there for security. The host computer and card reader actually "talk" to the microprocessor. The microprocessor enforces access to the data on the card. If the host computer can read and write the smart card's random access memory (RAM), it would be no different than a diskette.

Smarts cards may have up to 8 kilobytes of RAM, 346 kilobytes of ROM, 256 kilobytes of programmable ROM, and a 16-bit microprocessor. The smart card uses a serial interface and receives its power from external sources like a card reader. The processor uses a limited instruction set for applications such as cryptography.

So we need **applications** to transform the data residing on smart cards for appropriate usage like,

* Security solutions for electronic banking platform using 3DES and PKI cryptography, ensuring secured financial transactions.
* Security solutions for mobile payments using SIM/WIM cards.
* Campus card solutions used to store and manage identification, access control, attendance monitoring, marks card and e-cash based payment information.
* Security solutions using smart card as a copy protection device for software. Solutions for distributors and retailers of consumer goods.
* Solutions for vehicle dealers to keep track of all services and repairs done for a vehicle, along with loyalty points accrued

Readers and terminals operate with smart cards to obtain information that is contained in the card and perform a transaction.

Generally, a reader interfaces with a PC for the majority of its processing requirements. A terminal is a self-contained processing device. Both readers and terminals read and write to smart cards.

A separate and distinct software application is required, that interacts with the data and service from the card. The data is processed in varying environments where interoperability of application software as well as hardware components is a must; otherwise data will get blocked in the network and becomes useless.

In the upcoming chapters we will see how IT Applications are made interoperable in different industry verticals by adhering to open standards and also how open standards are governed by standard setting organization.