Introductory Concepts

This course will explain data communications and computer networking concepts in terms of system programmers. It is not a complete course on data communications but will focus on real-world examples which are encountered every day. The world of data communications and software development is drastically changing, as well as the nature of business, because of open source software, and cloud computing.

You are probably saying to yourself: **"I'm a programmer, why do I need to know about data communications?**

The answer is simple to make you a more productive programmer. But before we explore the reasons for this answer, let's consider a little terminology to get started:

- A computer is a programmable machine that can perform various computations, store data, and create documents by following a set of prerecorded instructions called a program. The data generated or stored by the computer is a collection of zeros and ones.
- Data communications is the sending of signals that represent zeros and ones over a point-to-point circuit between two computers.
- Networking begins when point-to-point circuits are joined together into a collection of computers for the exchange information and the sharing of resources.

In other words, data communications is concerned with bits and bytes and networking is concerned with interoperability of information.

The above are traditional definitions of these 3 words have been used for the last 50 years. However, these definitions are changing and the dividing line between them today is very blurred.

The objective of this course is to make you a more productive programmer by understanding data communications because the nature of programming and software development is changing. There are 3 reasons by you need to understand data communications and networking.

1 **Programmers need to know what happens in the network cloud to write useful applications**.

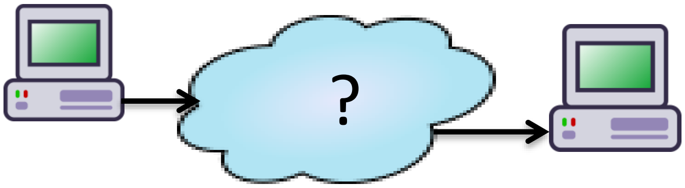


Figure 1: Networks Common Displayed as a Cloud

Have you ever wondered why networks are always displayed as "clouds"? The reason is simply that it doesn't matter how the information gets across the cloud; it only matters what goes into the cloud and what comes out.

When I began my career, programmers wrote programs that were installed on individual workstations or servers. Network support staff made sure that the network was up and operating within acceptable performance standards. Programmers never spoke to network support staff or vise versa – there was no need. Today, the nature of programming and network support is " converging" and it is essential that they communicate. Applications are not installed on just one machine, but rather on 3 or more machines. This means that the programmer must understand how the network works in order to best build the application. On the other hand, network support staff must know how the program works in order to correctly install the parts of the application and set up the ACLs (Access control lists) correctly so that application is secure and works as intended.

The enterprise computer is the smartphone. The iPhone 5 has 1GB of memory and suppose you are asked to write an application for credit card payment. You need to understand the limitations of RAM and network speed to properly build the application. The dropdown listing of countries in apps takes 30 KB of memory. jQuery, a popular API for writing JavaScript and manipulating HTML pages takes 90KB. Also, it makes no sense to write an application if the mobile network can't handle the throughput (actual speed to client host – note network speed is in bps not Bytes per second). The map below shows that there is presently a big different in network throughput among areas of Canada, ranging from 42 Mbps to 335 Mbps. Programmers must be able to estimate the total amount of data that needs to be transmitted within a particular time period (seconds, minutes, hours) to know how much throughput is needed. For example, suppose you wanted to transmit a 8 ½ X 11 inch page of data with a 1 inch margin all around (content area 6.5 X 9 inches). The average font has 10 characters per inch and there are 3 lines per inch.

How many characters are there? 10 characters X 6.5 inches X 27 lines = 1,755 characters

How many characters using UniCode encoding? 1755 characters X 16 bits = 28,080 bits per page

How long will it take to transmit?

Dialup @ 56000 bps = 28080 / 56000 =0.5 seconds approximately

4G @ 1.5Mbps = 28080 / 1500000 = 0.019 approximately

Download speed of .5 seconds sounds fast, but this speed will prevent VoIP of IoT applications from working. Throughput must be less than a 1/3 of a second for these applications to work. What will the speed be if we factor in congestion, bad weather and distance?

Network speed can change:

2 Weather – rain or snow can slow the network

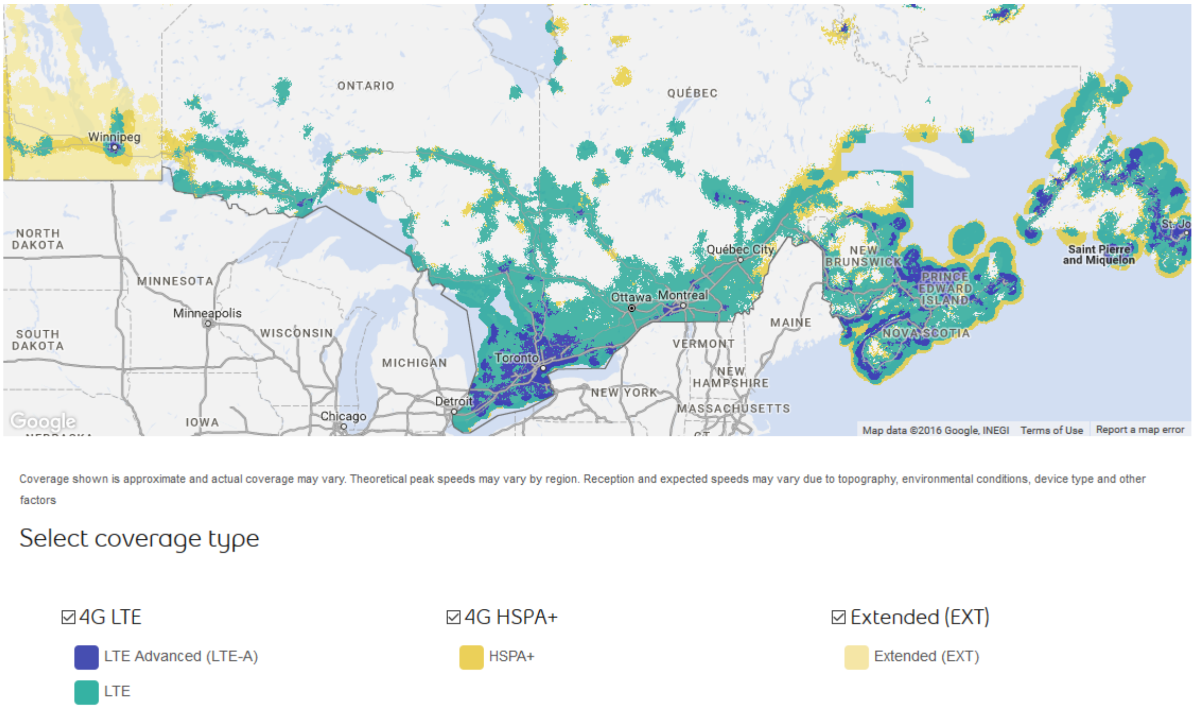
3 Bandwidth is share so the more users the slower the network

4 Distance from Cell tower – the further way the slower the network

5 Amount of Network Congestion – latency at towers and routers slows the network

By understanding the layers of abstraction below the network level you will be better able to write and to troubleshoot network application. **The goal is for you to be a more productive programmer.**

https://www.whistleout.ca/CellPhones/Guides/Coverage



Coverage shown is approximate and actual coverage may vary. Theoretical peak speeds may vary by region. Reception and expected speeds may vary due to topography, environmental conditions, device type and other factors

Select coverage type

☑4G LTE
- 🟦 LTE Advanced (LTE-A)
- 🟩 LTE

☑4G HSPA+
- 🟨 HSPA+

☑Extended (EXT)
- 🟨 Extended (EXT)

⬜	Dialup 56 kbps
🟨	4G – 1.5 Mbps
🟩	LTE – 2.5 Mbps
🟦	LTE-A – 5.5 Mbps

Table 1: Partial Map of Mobile Telephone Networks in Canada

Don't forget that this map was taken in 2015, and the majority of Canada, once you get 100 miles from the 49th parallel is dialup speed at 56K maximum. We often forget this when we live in the city. As a programmer, you need to design your application to work with the available speeds of the user's area.

For more detailed information about the service providers and their network coverage refer this article.

https://www.whistleout.ca/CellPhones/Guides/Coverage

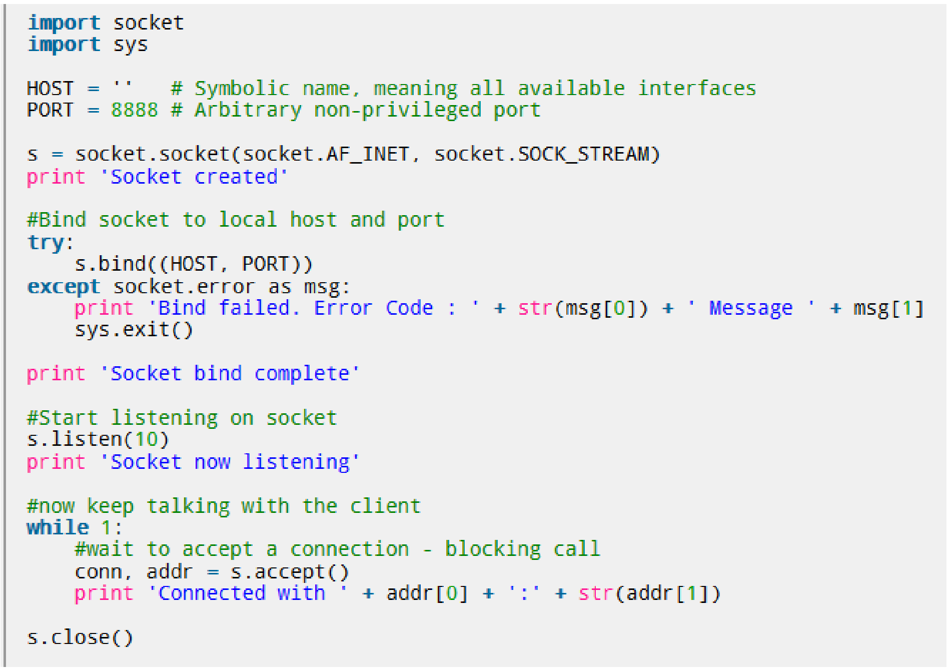
2 Programmers must also know how hackers can exploit applications

Programmers must also know how hackers can exploit

Figure 1: Simple Socket Server using Python

applications in order to build a secure application and if it fails, to fail as safely as possible. All applications today are being accessed over a network. Programmers need to understand how hackers will try and use the application in ways it was not intended. Everyone has heard the term “buffer overflow” or “SQL injection”. These exploits are examples of programming errors where the programmer was overly trusting about the applications input data. In terms of the Internet many developers have a basic knowledge, such as 192.168.0.1 is an IP address used by IPv4 to designate a host on a network. But as programmers, we need to understand the process of using sockets in applications and how application data can be changed by malicious individuals. There is no programming in this course. The simple socket server example below is used to illustrate the process of creating a socket and level of abstraction needed to build a mobile application. A good mobile application will have a “choke-point” where all input data will be checked to ensure the data is in the format and length expected by the application. And if not, how to sanitize the input using stored procedures or regular expressions.

A common attack on a LAN is MAC spoofing or ARP Cache poisoning where an attacker over-writes the data communication address used on LANs with his/her MAC address and thus places himself/herself in the middle to two authorized users who are communicating. This is called a Man-in-the-middle attack (MITM).

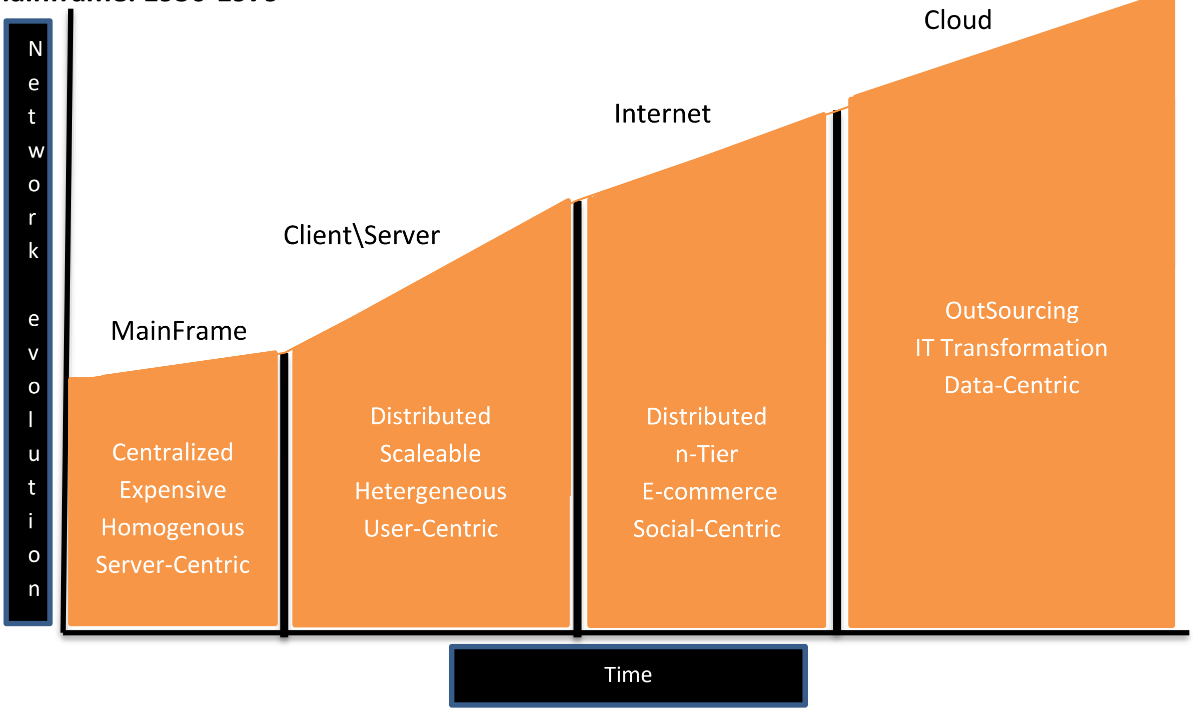


```
import socket
import sys

HOST = ''   # Symbolic name, meaning all available interfaces
PORT = 8888 # Arbitrary non-privileged port

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
print 'Socket created'

#Bind socket to local host and port
try:
    s.bind((HOST, PORT))
except socket.error as msg:
    print 'Bind failed. Error Code : ' + str(msg[0]) + ' Message ' + msg[1]
    sys.exit()

print 'Socket bind complete'

#Start listening on socket
s.listen(10)
print 'Socket now listening'

#now keep talking with the client
while 1:
    #wait to accept a connection - blocking call
    conn, addr = s.accept()
    print 'Connected with ' + addr[0] + ':' + str(addr[1])

s.close()
```

3 John Gage has described the IT environment of today as “The network is the computer.”

With the cloud becoming a development platform, our definition of a computer applies to a virtual computer in the cloud. The cloud can run programs and store data just like the computer, and through virtualization the cloud is a network of thousands of computers with unparalleled processing power. Let’s briefly give a historical framework of how cloud computing came about and its impact on business. See Figure 1: Network Evolution over Time.

Host to Mainframe: 1950-1979



Host to Mainframe: 1950-1979

One of the first networks was a host to mainframe network. A host is a computer connected to a network. These computers are not like the PCs of today, but are "dumb" computers, or terminals, lacking a hard drive and RAM processing power. A host to mainframe is a "server-centric", homogeneous system. All processing is centralized on the server and programs had to be written with data and executable code on the same machine. This architecture is also homogeneous, meaning the hardware and software were usually supplied by the same vendor. Mainframe systems are very powerful and reliable, but very expensive - only major corporations could afford this architecture.

Figure 2: Network Evolution over Time

.

Client-Server: 1980-1990

The personal computer was first marketed in the fall of 1979. The marketing of a low cost "smart" computer brought processing power to everyone. So, instead of using one big computer with centralized software and 'dumb' terminals, client-server networks gave each employee a personal computer and 'empowered' the employee to use whatever software was best for the job. This feature made client-server architecture very appealing because it was user-centric. Unlike mainframe, client-server architecture is heterogeneous meaning hardware and software can be purchased from different vendors. Processing power was distributed between the client and the server. The client accessed shared resources and the server shared resources. Distributed computing is easily scalable to the size of the business making it ideal to a growing business and more affordable than mainframes.

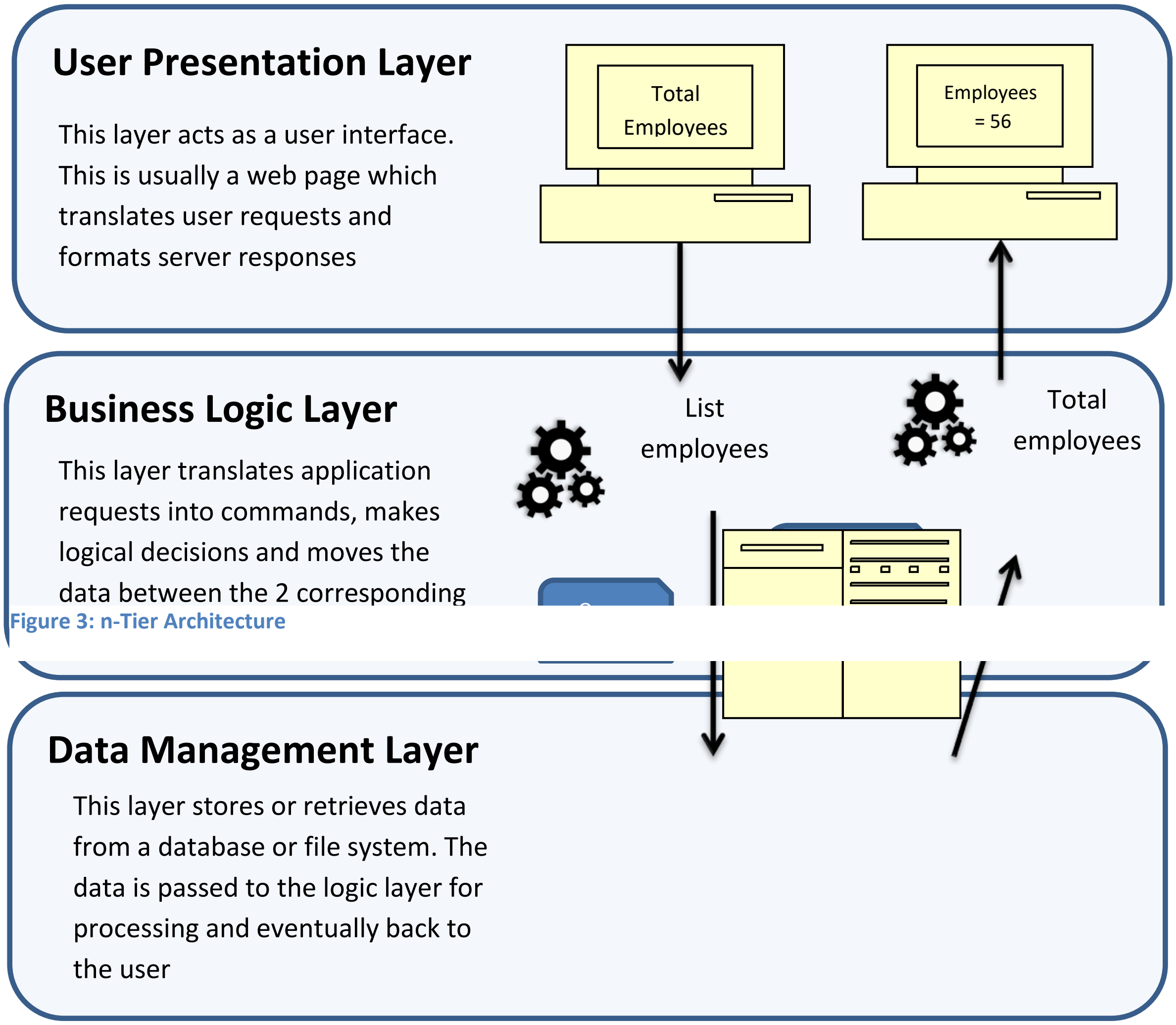
Programs were written to take advantage of the combined processing power by developing a client application to interface with the server. The client captures user input and forwards it to the server for processing as a request. The server application is usually written in such a way that it does not provide any method to interact with a user. Instead, it waits for client programs to connect. The server processes the data and sends it back to the client which formats the output. The client-server architecture continues today but has changed with the advent of the Internet.

Internet: 1991-2009

The World Wide Web developed in 1991 by Tim Berners Lee provided a simple way of accessing files and displaying text and images using HTML markup language. The HTML image tag allowed the placement of images in a web page, but it had an attribute called SRC (source) which allowed the image file to be on a different server than the web server processing the client page request. This concept of using multiple servers to process a web page led to a more distributed client-server system called **n-Tier or multi-Tier programming.** The most widespread use of n-Tier programming is a three-tier model separating the presentation, which acts as a user interface, from the business logic, which contains the executable code, and data management layers, which is responsible for data storage. This created a more heterogeneous system because with the aid of middleware, the layers can even be of different platforms. For example, you can have an Oracle database server on the backend with a Unix application server working with a Windows client.

The growth of the web moved businesses from displaying production inventory in static pages to the web becoming a point of sale through interactive, and collaborative processes called E-commerce.
Users can read reviews of the product, search for the best price among vendors and buy the product on line. Businesses often outsource E-commerce, due to its high cost, rather than setting up in-house web departments. This outsourcing trend is continued and enhanced with cloud computing.

This programming model is the business standard today splitting the application into several parts and each part controlled by a different machine. For example, Figure 4 illustrates the tiered approach if you wanted to retrieve the total number of employees. The user presentation layer which is the top-level of the application acts as a user interface. It captures the user's request to get the Total number of employees and passes this data to the Business Logic layer. This second layer provides a web service to the application layer and translates the user's request into commands, generating a query to retrieve a listing of employees. The command is sent to the third layer, Data Management layer. This layer runs the commands producing a listing of employees. The listing is sent back to the Business Logic layer which uses a web service to total the listing and pass the number back to the user presentation layer which displays the result.



User Presentation Layer

This layer acts as a user interface. This is usually a web page which translates user requests and formats server responses

Business Logic Layer

This layer translates application requests into commands, makes logical decisions and moves the data between the 2 corresponding

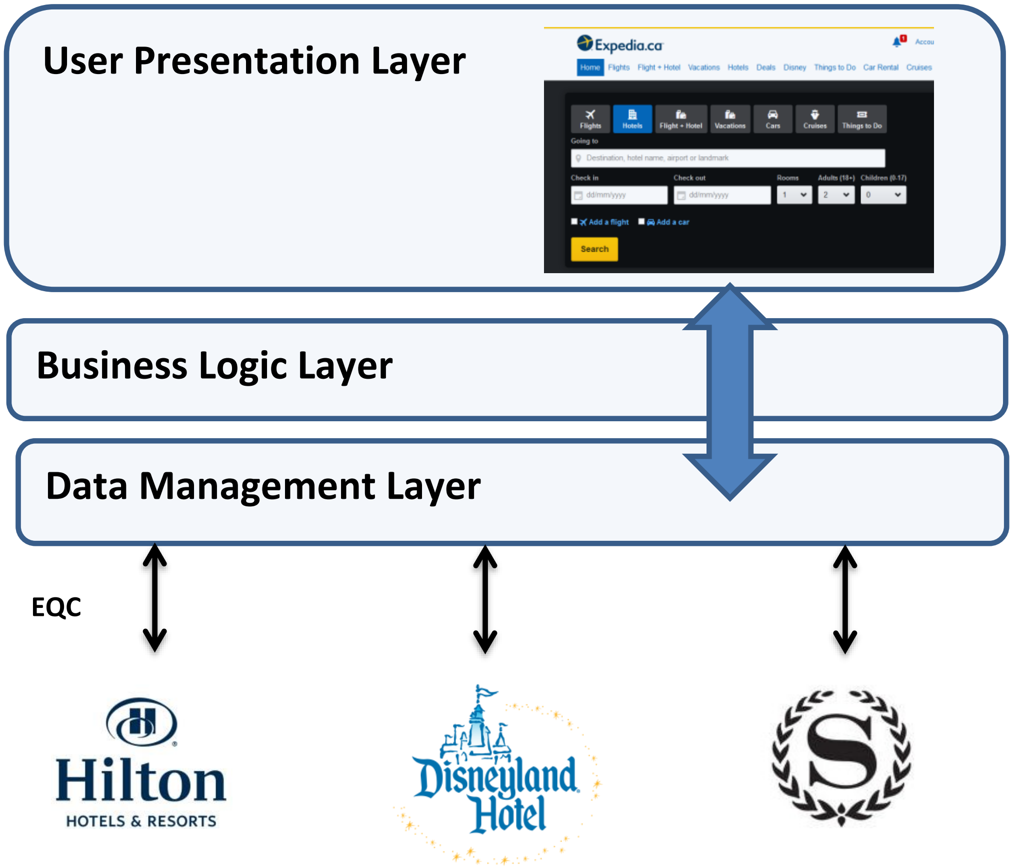
Data Management Layer

This layer stores or retrieves data from a database or file system. The data is passed to the logic layer for processing and eventually back to the user

Figure 3: n-Tier Architecture

The n-Tier model is not restricted to a single network, but can be used to integrate businesses into a common portal of information, such as Expedia.

For example, it is difficult for travelers to know all of the specials and room rates of hotels at a specific destination. Hotels, at that destination, have a time sensitive inventory of rooms; no booking, no revenue. To maintain an occupancy rate of 70%, hotels will sell a block of rooms with a room rate of $170 per night, to Expedia for $95. Expedia then resells the rooms to you on their web site for $120-$130. It’s a win-win combination; travelers are guaranteed the best rate and the hotel gains revenue and exposure to a wider market of travelers. Expedia has developed software called Expedia Quick Connect which links the hotel’s central reservation system directly into the Data Management Layer. Changes made to the central reservation system are automatically upgraded to Data Management Layer.



Cloud-Computing: 2010- present

In 1990, Bill Gates stated information will be at everyone's fingertips when 3 developments occur: a more "personal" personal computer, more powerful communications networks, and easy access to a broad range of information. It took 20 years of development to get here, but his vision is now reality. Smartphones and tablets are connected to fast broadband wireless networks which connect to a third-party cloud service, making it possible for you to connect to all the information you want for anywhere on any device. This combination of mobile and cloud technologies has created a new data-centric era. As more companies migrate their data and applications to the cloud, the IT landscape will change dramatically. Some of the changes are the following:

1. OutSourcing

Outsourcing is an established practice for companies looking to take the complexity out of managing their infrastructure. During the Internet period outsourcing was most popular with E-commerce - allowing them to hand over the problem to a service company for a fee.

In 2008, Netfllix had a server failure and their DVD mailing system was down for several days. Management realized that reliability would be essential for the new online streaming service it hoped to introduce in 2009. Instead of creating its own server farms and staff for the online system, Netflix outsourced the server operation to Amazon Web Services (AWS). This form of relation is called Infrastructure as a Service (IaaS). Amazon had leveraged its expertise in managing server E-commerce server farms into a cloud service that customers like Netflix could use without having to understand server operators. Also, cloud service is sold like electricity; you only pay for what you use. This is more efficient than developing your own farms and maintaining under-utilized servers during holiday periods. Though paying for a cloud computing service provider may cost money, most companies, like Netflix, will save money in the long run as a result of cloud computing.

When data is in the cloud, a breach of the cloud provider's security compromises the company's data, and a malfunction of cloud software can result in the loss of data that the company needs. Security is a key factor with cloud service providers and it is impossible to understand the provider's security practices (since you are trusting a 3rd party). To protect against these problems, companies often invest in preventative measures, such as VaultLogic.com which is online cloud backup service and Service License Agreements (SLAs) to protect against lose.

2. IT Transformation

As cloud computing is adopted fewer businesses will have a need for in-house IT departments. Many of the in-house departments will be moved to the cloud, but the impact of cloud computing is greatest in IT departments. In the beginning, the company will pay to migrate data to the cloud. However, after a cloud computing system is in place, the company won't require as many IT professionals, which saves money in wages. Personnel who used to work in IT will either be laid off or given new responsibilities such as inputting and managing company data on the cloud, or liaising with cloud service providers. Likewise, the company won't need to upgrade in-house software any more, nor will it need to purchase large amounts of hard drive storage to manage its own data.

3. Data-Centric

IaaS allows companies to outsource infrastructure into the cloud. SaaS allows companies to do the same with software. Companies need to support employees who today have multiple devices. A typical worker will have a smartphone, a laptop, and a desktop computer. Workers want to access files in a consistent manner across all devices (client hosts). The cloud provides this experience by creating a virtual client with virtual drive and software. When a worker turns on a laptop, for example, and logs into the virtual client, he or she has access to all of the virtual client's applications and data files. In addition, the virtual client remembers its configurations, so all of the worker's shortcuts and customization appear on devices. When the worker uses another device, he or she can continue working on the same document, with the same program, picking up exactly where he or she left off.

Since the cloud consists of virtual client's, sharing work can be done in ways that are impossible with a desktop computer. Specific folders or files can be shared and the worker can set access rights to the data. This allows multiple users to collaborate on a document.

The most important aspect of the cloud is its processing speed. The cloud with hundreds or even thousands of virtual clients can process data many times faster than in-house systems. This development speeds up all business-related operations, such as buying, selling, searching for information, and data management, making businesses more efficient overall.

Cloud and Programming

The cloud has become a new development platform for programmers. Microsoft Azure, Google's Cloud SKD allow programmers to write cloud-based applications. The 15th version of MSOffice is fully compatible with the cloud to allow users to use the cloud as their local PC. Many developers today, are converting existing programs to run on clouds, rather than creating new types of applications that exploit the power of the cloud. For example, an online music retailer could monitor popular social-media feeds; if a singer suddenly became a hot topic, advertising and special offers across the retailer's site could be instantly reconfigured to make the most of the spike in interest.

Joseph Hellerstein, of the University of California, Berkeley, is working on a new language called BLOOM which will help programmers write complex data-centric cloud applications. His big idea is to modify database programming languages so that programmer's just think about the results they want, rather than micromanaging data in the cloud. Let's watch a short video on the BLOOM project.

https://www.scnsoft.com/blog/cloud-development-languages

https://www.improgrammer.net/20-cloud-programming-languages/

http://www2.technologyreview.com/article/418545/tr10-cloud-programming/

Next week we will look at Network Standards-