**Into the industry competitors**

***(Comparing and contrasting the existing products) “Who what where”***

Autodata:

Vivid/Haynes:

**Evolution of Mobile Technologies**

Handheld portable devices have been in existent longer than you might think, for a lot of people, it was only yesterday that the iPhone emerged on the scene, and everyone wanted one. And then Apple announced the iPad, basically a big version of the iPhone with a bigger screen without the traditional phone features, which smartly bridged the gap between the laptop and the smartphone.

This brings us down to two complementary mobile devices, the smartphone -something that fits into your pocket - and the tablet, something still light and portable, which has the screen size and peripherals to do a large po work.

The Advent of Android

Nice as the iPhone and the iPad were technologically, they were insistently notopen source. In fact, many people had concerns with Apple’s limitations on what could and could not be put in your smartphone. The smartphone has been arriving for quite some time, with at least three well-established mobile operating systems in the market, so the iPhone was soon to have some rivalry.

Android Inc. was founded in 2003, and acquired by Google in 2005. The Open Handset Alliance - a consortium of companies including Google, Motorola, HTC, and a bunch of other handset manufacturers and carriers - was unveiled in November 2007 (only a mere two months after the launch of the iPhone). That same day, the Android platform was revealed.

Android was built on a Linux kernel, and is essentially open source. Android apps are mainly written in Java. The Android version of Java runs on what is called the Dalvik virtual machine, which is enhanced exactly for handheld devices.

Fast-forward a few years to 2011 there have been a few Android tablets released, but these are still basically enlarged mobile phones as far as their operating system is concerned. It’s only with the release of Android 3.0 (Honeycomb) that we have a version of the android operating systems that takes full advantage of the extended capabilities of a full-sized tablet.

To understand what makes Android so fascinating, we must examine how mobile development has evolved and how Android differs from rival platforms.

Remember when a phone was just a phone? When we relied on fixed land - lines? When we ran for the phone instead of pulling it out of our pocket? When we lost our friends at crowded places and waited around for hours hoping to reunite? When we forgot the grocery list and had to find a payphone or drive back home again?

Those days are long gone. Today, problems such as these are easily solved with a mobile phone.

Our mobile phones keep us safe and connected. Now we roam around freely, relying on our phones not only to keep in touch with friends, family, and co -workers, but also to tell us where to go, what to do, and how to do it. Even the most domestic of events seem to revolve around my mobile phone.

Mobile phones can solve just about anything - and we rely on them for everythingthese days. Before Android, mobile developers faced many obstructions when it came to writing applications. Building the better application, the unique application, and incorporating many common tasks such as messaging and calling in a familiar way were often unrealistic goals.

To understand why, let’s take a brief look at the history of mobile software development.

The Motorola DynaTAC 8000X was the first commercially available cell phone. First sold in 1983, it was 13 × 1.75 × 3.5 inches in dimension, weighed about 2.5 pounds, and allowed you to talk for just over a half an hour. It retailed for $3,995, plus hefty monthly service fees and per-minute charges.

It was referred to as “The Brick,” and the nickname stuck with many of those early mobile phones, which we either loved or hated. About the size of a brick, with a battery power just long enough for half a conversation, these early mobile handsets were mostly seen in the hands of traveling business execs, security personnel, and the wealthy. First-generation mobile phones were just too expensive. The service charges alone would bankrupt the average person, especially when roaming.

Early mobile phones were not full of features as we now expect as standard. These early phones did little more than make and receive calls and, if you were lucky, there was a simple contacts application that wasn’t impossible to use.

The first-generation mobile phones were designed and developed by the handset manufacturers. Competition was fierce and trade secrets were closely guarded. Manufacturers didn’t want to expose the internal workings of their handsets, so they usually developed the phone software in-house. As a developer, if you weren’t part of this inner circle, you had no opportunity to write applications for the phones.

It was during this period that we saw the first “time-waster” games begin to appear. Nokia was famous for putting the 1970s video game Snake on some of its earliest monochrome phones. Other manufacturers followed suit, adding games such as Pong, Tetris, and Tic-Tac-Toe.

These early phones were flawed, but they did something important - they changed the way people thought about communication. As mobile phone prices dropped, battery life improved, and reception areas grew, more and more people began carrying these handy devices. Soon mobile phones were more than just a novelty.

Customers began pushing for more features and more games. But there was a problem. The handset manufacturers didn’t have the motivation or the resources to build every application users wanted. They needed some way to provide a gateway for entertainment and information services without allowing direct access to the handset.

What better way to provide these services than the Internet?

The solution was the emergence of a new technology Wireless Application Protocol (WAP)

As it turned out, allowing direct phone access to the Internet didn’t scale well on mobile devices. By this time, professional websites were full color and wedged full of text, images, and other sorts of media. These sites relied on JavaScript, Flash, and other technologies to enhance the user experience, and they were often designed with a target resolution of 800x600 pixels and higher.

When the first clamshell phone, the Motorola StarTAC, was released in 1996, it merely had an LCD 10-digit segmented display. (Later models would add a dot-matrix type display.) Meanwhile, Nokia released one of the first slider phones, the 8110 - often referred to as “The Matrix Phone” because the phone was heavily used in films. The 8110 could display four lines of text with 13 characters per line.

With there stamp sized low-resolution screens and limited storage and processing power, these phones couldn’t handle the data-intensive operations required by traditional web browsers. The bandwidth requirements for data transmission were also costly to the user.

The Wireless Application Protocol (WAP) standard emerged to answer these concerns. WAP was basically a stripped-down version of HTTP, which is the backbone protocol of the Internet. Unlike traditional web browsers, WAP browsers were designed to run within the memory and bandwidth constraints of the phone. Third-party WAP sites served up pages written in a markup language called Wireless Markup Language (WML). These pages were then displayed on the phone’s WAP browser. Users navigated as they would on the Web, but the pages were much simpler in design.

The WAP solution was great for handset manufacturers. The pressure was off they could write one WAP browser to ship with the handset and rely on developers to come up with the content users wanted.

The WAP solution was great for mobile operators. They could provide a custom WAP gateway, directing their users to the content they wanted to provide, and rake in the data charges associated with browsing, which were often high.

Developers and content providers didn’t deliver. For the first time, developers had a chance to develop content for phone users; some did so but with limited success.

Most of the early WAP sites were extensions of popular branded websites, such as CNN.com and ESPN.com, which were looking for new ways to extend their readership and viewership. Suddenly phone users had the news, stock market quotes, and sports scores available at their fingertips on their phones.

Commercializing WAP applications was difficult, and there was no built-in billing mechanism. Some of the most popular commercial WAP applications that emerged during this time were simple wallpaper and ringtone directories that enabled users to customize their phones for the first time. For example, a user browsed a WAP site and requested a specific item. He filled out a simple order form with his phone number and his handset model. It was up to the content provider to deliver an image or audio file compatible with the given phone. Payment and verification were handled through various premium priced delivery mechanisms such as Short Message Service (SMS), Enhanced Messaging Service (EMS), Multimedia Messaging Service (MMS), and WAP Push.

WAP browsers, especially in the early days, were slow and frustrating. Typing long URLs with the numeric keypad was tedious. WAP pages were often difficult to navigate. Most WAP sites were written one time for all phones and did not account for individual phone specifications. It didn’t matter if the end user’s phone had a big color screen or a stamp-sized monochrome screen.

The developers could not alter the user’s experience. The result was ordinary and not very absorbing experience for everyone involved.

Content providers often didn’t bother with a WAP site and instead just advertised SMS short codes on TV and in magazines. In this case, the user sent a premium SMS message with a request for a specific wallpaper or ringtone, and the content provider sent it back. Mobile operators generally liked these delivery mechanisms because they received a large portion of each messaging fee.

WAP fell short of commercial expectations. In some markets, such as Japan, it flourished, whereas in others, such as the United States, it failed to take off. Handset screens were too small for surfing. Reading a sentence fragment at a time, and then waiting seconds for the next segment to download, ruined the user experience, especially because every second of downloading was often charged to the user. Critics began to call WAP “Wait and Pay.”

Ultimately, the mobile operators who provided the WAP gateway (the default home page loaded when you started your WAP browser) often regulated which WAP sites were accessible. The gateway enabled the operator to restrict the number of sites users could browse and to funnel users to the operator’s preferred content providers and disregard competing sites. This kind of walled garden approach further discouraged third-party developers, who already faced difficulties in monetizing applications.

It came as no surprise that users wanted more - they will always want more. Writing robust applications with WAP, such as graphic-intensive video games, was nearly impossible. The 18-year-old to 25-year-old demographic, the kids with the disposable income most likely to individualize their phones with wallpapers and ringtones—looked at their portable gaming systems and asked for a device that was both a phone and a gaming device or a phone and a music player. They argued that if devices such as Nintendo’s Game Boy could provide hours of entertainment with only five buttons, why not just add phone capabilities? Others looked to their digital cameras, Palms, BlackBerries, iPods, and even their laptops and asked the same question. The market seemed to be teetering on the edge of devices merging.

Memory was getting cheaper, battery life was getting better, and PDAs and other embedded devices were beginning to run compact versions of common operating systems such as Linux and Windows. The traditional desktop application developer was suddenly involved in the embedded device market, especially with the emergence of smartphone technologies such as Windows Mobile, which they were accustomed to.

Handset manufacturers recognized that if they wanted to continue to sell traditional handsets, they needed to change their protective policies relating to handset design and expose their internal frameworks to some extent.

A variety of different proprietary platforms emerged - and developers are still actively creating applications for them. Some smartphone devices ran Palm OS (now Garnet OS) and RIM BlackBerry OS. Sun Microsystems took its popular Java platform and created J2ME now known as Java Micro Edition [Java ME]. Chipset maker Qualcomm developed and licensed its Binary Runtime Environment for Wireless (BREW). Other platforms such as Symbian OS were developed by handset manufacturers; Nokia, Sony Ericsson, Motorola, and Samsung. The Apple iPhone OS (OS X iPhone) joined the ranks in 2008.

Many of these platforms have accompanying developer programs. These programs keep the developer communities small, vetted, and under contractual agreements on what they can and cannot do. These programs are often required developers to pay for them.

Each platform has benefits and drawbacks. Of course, developer’s love to debate about which platform is “the best”. The truth is that no one platform has emerged victorious. Some platforms are best suited for commercializing games and making millions - if your company has brand backing. Other platforms are more open and suitable for the hobbyist. No mobile platform is best suited for all possible applications. As a result, the mobile phone has become increasingly fragmented, with all platforms sharing a piece of the market.

For manufacturers and mobile operators, handset product lines quickly became convoluted. Platforms market penetration varied greatly by region and user demographic. Instead of choosing just one platform, manufacturers and operators have been forced to sell phones for all the different platforms to compete in the market. We’ve even seen some handsets supporting multiple platforms. (For instance, Symbian phones often also support J2ME.)

The mobile developer community has become as fragmented as the market. It’s nearly impossible to keep track of all the changes in the market - niches have formed. The numerous platforms development requirements vary greatly. Mobile software developers work with noticeably different programming environments, different tools, and different programming languages. Porting among the platforms is often costly and not straightforward. Keeping track of handset configurations and testing requirements, carrier relationships, and application marketplaces have become intricate spin-off businesses of their own.

It’s a nightmare for the any company that wants a mobile application. Should the company develop a J2ME application? BREW? IPhone? Windows Mobile? Everyone has a different kind of phone. The company is forced to choose one or, worse, all of the platforms. Some platforms allow for free applications, whereas others do not. Vertical market application opportunities are limited and expensive.

As a result, many great applications have not reached their desired users, and many other great ideas have not been developed at all.

Enter search engine and advertising giant Google. Now a household name, Google has shown an interest in extending its vision, its brand, its search and ad-revenue-based platform, and its suite of tools to the wireless market. The company’s business model has been remarkably successful on the Internet and, technically speaking, wireless isn’t that different.

The company’s initial ventures into mobile were plagued with all the problems you would expect. Mobile phone users did not share the freedoms Internet users enjoyed. Internet users can choose from the wide variety of computer brands, operating systems, Internet service providers, and web browser applications.

Nearly all Google services are free and ad driven. Many applications in the Google Labs suite directly compete with the applications available on mobile phones. The applications range from simple calendars and calculators to navigation with Google Maps - not to mention corporate acquisitions such as Blogger and YouTube.

When this approach didn’t yield the intended results, Google decided on a different approach - to revamp the entire system upon which wireless application development was based, hoping to provide a more open environment for users and developers: the Internet model. The Internet model allows users to choose between freeware, shareware, and paid software. This enables free market competition among services.

With its user-centered and it targeted design ideas, Google has led a movement to turn the existing closely secured wireless market into one where phone users can move between carriers easily and have unrestricted access to applications and services. With its vast resources, Google has taken a broad approach, studying the wireless infrastructure, handset manufacturers requirements, application developer needs, and mobile operator desires.

Next, Google joined with other like-minded associates in the wireless community and presented the following problem: What would it take to build a better mobile phone?

The Open Handset Alliance (OHA) was formed in November 2007 to answer that very problem. The OHA is a business alliance comprised of many of the largest and most successful mobile companies from around the world. Its members include chipmakers, handset manufacturers, software developers, and service providers. The entire mobile supply chain is well represented.

Andy Rubin has been recognized as the father of the Android platform. His company, Android Inc., was acquired by Google in 2005.Working together with the OHA members, including Google, began developing a nonproprietary open source platform based upon technology developed at Android Inc. that would aim to improve the above-mentioned problems obstructing the mobile community. The result is the Android project. To this day, Rubin’s team at Google, where he is Vice-President of Engineering and manages the Android platform roadmap, completes most of the android platform development.

Google’s participation in the Android project has been so extensive that the line between who takes responsibility for the Android platform (the OHA or Google) has blurred. Google hosts the Android open source project and provides online Android documentation, tools, forums, and the Software Development Kit (SDK) for developers. All major Android news emerges from Google. The company has also hosted a number of events including conferences and the Android Developer Challenge (ADC), a contest to encourage developers to design killer Android applications - for $10 million dollars in prizes to stimulus development on the platform. The winners and their apps are listed on the Android website.

Manufacturers: Designing the Android Handsets

More than half the members of the OHA are handset manufacturers, such as Samsung, Motorola, HTC, and LG, and semiconductor companies, such as Intel, Texas Instruments, NVIDIA, and Qualcomm. These companies helped design the first generation of Android handsets.

Handset manufacturer HTC with service provided by T-Mobile developed the first shipping Android handset the T-Mobile G1. It was released in October 2008. The platform gained momentum relatively quickly. Each new Android device was more powerful and exciting than the last. Over the following 18 months, 60 different Android handsets (made by 21 different manufacturers) debuted across 59 carriers in 48 countries around the world. By June 2010, at an announcement of a new, highly anticipated Android handset, Google announced more than 160,000 Android devices were being activated each day (for a rate of nearly 60 million devices annually). The advantages of widespread manufacturer and carrier support appear to be really paying off at this point.

The Android platform is now considered a success. It has shaken up the mobile market place, gaining ground progressively against competitive platforms such as the Apple iPhone, RIM BlackBerry, and Windows Mobile.

The latest figures??

Mobile Operators: Delivering the Android Experience

After you have the phones, you have to get them out to the users. Mobile operators from North, South, and Central America; Europe, Asia, India, Australia, Africa, and the Middle East have joined the OHA, ensuring a worldwide market for the Android faction. With almost half a billion customers alone, telephony giant China Mobile is a founding member of the alliance.

Much of Android’s success is also due to the fact that many Android handsets don’t come with the traditional “smartphone price tag” - quite a few are offered free with activation by carriers. Competitors such as the Apple iPhone have no such offers. For the first time, the average Jane or Joe can afford a feature-full phone.

Content Providers: Developing Android Applications

When users have Android handsets, they need the best apps, right?  Google has led the pack, developing Android applications, many of which are core features of the platform, such as a email client and web browser. OHA members are also working on Android application integration. eBay, for example, is working on integration with its online auctions.

The first ADC received 1,788 submissions, with the second ADC being voted upon by 26,000 Android users to pick a final 200 applications that would be judged professionally - all newly developed Android games, productivity helpers, and a slew of location based services (LBS) applications. We also saw humanitarian, social networking, and mash-up apps. Many of these applications have debuted with users through the Android Market now referred to as Google play – which is a software distribution mechanism for Android. For now, these challenges are over. The results, though, are still impressive.

For those working on the Android platform from the beginning, handsets couldn’t come fast enough. The T-Mobile G1 was the first commercial Android device on the market, but it had the appearance of a developer pre-release handset. Following Android handsets have had more impressive hardware, allowing developers to dive in and design amazing new applications.

***How many apps on the market place?*** Which is growing rapidly. This takes into account only applications published through this one marketplace.

***How many developers?***

Taking Advantage of All Android Has to Offer

Android’s open platform has been embraced by much of the mobile development community - spreading far beyond the members of the OHA.

As Android phones and applications have become more readily available, many other mobile operators and handset manufacturers have jumped at the chance to sell Android phones to their customers, particularly given the cost benefits compared to proprietary platforms. The open standard of the Android platform has resulted in reduced operator costs in licensing and royalties, and we are now seeing a migration to open handsets from proprietary platforms such as RIM, Windows Mobile, and the Apple iPhone. The market has cracked wide open; new types of users are able to consider smartphones for the first time. Android is well suited to fill this demand.

Android: A Next-Generation Platform

Although Android has many innovative features not available in existing mobile platforms. It’s true that many of these features appear in existing proprietary platforms, but Android combines them in a free and open fashion while at the same time solving many of the flaws on these competing platforms.

Android is the first in a new generation of mobile platforms, giving its platform developers a clear edge on the rivals. Android’s designers studied the benefits and problems of existing platforms and then combined their most successful features. At the same time, Android’s designers evaded the mistakes others experienced in the past.

Since the Android 1.0 SDK was released, Android platform development has continued at a fast and furious pace. For quite some time, there was a new Android SDK out every couple of months! In typical tech-sector jargon, each Android SDK has had a project name. In Android’s case, the SDKs are named alphabetically after sweets.

Android Platform Differences

Android is an open source platform. Neither developers or handset manufacturers pay royalties or license fees to develop for the platform.

The underlying operating system of Android is licensed under GNU General Public License Version 2 (GPLv2), a strong “copyleft” license where any third-party improvements must continue to fall under the open source licensing agreement terms. The Android framework is distributed under the Apache Software License (ASL/Apache2), which allows for the distribution of both open and closed-source derivations of the source code. Commercial developers handset manufacturers especially can choose to enhance the platform without having to provide their improvements to the open source community. Instead, developers can profit from enhancements such as handset-specific improvements and redistribute their work under whatever licensing they want.

Android application developers have the ability to distribute their applications under whatever licensing scheme they prefer. Developers can write open source freeware or traditional licensed applications for profit and everything in between.

Unlike some proprietary platforms that require developer registration fees, vetting, and expensive compilers, there are no upfront costs to developing Android applications.

The Android SDK and tools are freely available. Developers can download the Android SDK from the Android website after agreeing to the terms of the Android Software Development Kit License Agreement.

Developers have several choices when it comes to integrated development environments (IDEs). Many developers choose the popular and freely available Eclipse IDE to design and develop Android applications. Eclipse is the most popular IDE for Android development, and there is an Android plug-in available for facilitating Android development. Android applications can be developed on the following operating systems: Windows XP (32-bit) or Vista (32-bit or 64-bit) Mac OS X 10.5.8 or later (x86 only) Linux (tested on Linux Ubuntu 8.04 LTS, Hardy Heron)

Android applications are written in a well-respected programming language: Java. The Android application framework includes traditional programming constructs, such as threads and processes and specially designed data structures to encapsulate objects commonly used in mobile applications. Developers can rely on familiar class libraries, such as java.net and java.text. Specialty libraries for tasks such as graphics and database management are implemented using well-defined open standards such as OpenGL Embedded Systems (OpenGL ES) or SQLite.

In the past, handset manufacturers often established special relationships with trusted third-party software developers (OEM/ODM relationships). This elite group of software developers wrote built-in applications, such as messaging and web browsers, which shipped on the handset as part of the phone’s core feature set. To design these applications, the manufacturer would grant the developer limited inside access and knowledge of a handset’s internal software framework and firmware.

On the Android platform, there is no difference between built-in and third party applications, enabling healthy competition among application developers. All Android applications use the same libraries. Android applications have unprecedented access to the underlying hardware, allowing developers to write much more powerful applications. Applications can be extended or replaced altogether.

One of the Android platform’s most compelling and innovative features is well-designed application integration. Android provides all the tools necessary to build better applications by allowing developers to write applications that seamlessly control core functionality such as web browsing, mapping, contact management, and messaging. Applications can also become content providers and share their data among each other in a secure fashion.

Platforms such as Symbian have experienced from setbacks due to malware. Android’s robust application security model helps protect the user and the system from malicious software.

Android applications have none of the costly and time-intensive testing and certification programs required by other platforms such as BREW and Symbian.

Android developers are free to choose any kind of revenue model they want. They can develop freeware, shareware, or trial-ware applications, ad-driven, and paid applications. Android was designed to fundamentally change the rules about what kind of wireless applications could be developed. In the past, developers faced many constraints that had little to do with the application functionality or features:

* Store limitations on the number of competing applications of a given type
* Store limitations on pricing, revenue models, and royalties
* Operator unwillingness to provide applications for smaller demographics

With Android, developers can design and successfully publish any kind of application they want. Developers can adapt applications to small demographics, instead of just large-scale money making ones often insisted upon by mobile operators. Vertical market applications can be utilized to targeted users.

Because developers have a range of application distribution methods to choose from, they can pick the methods that work for them instead of being forced to play by others rules.

Android might be the next generation in mobile platforms, but the technology is still in its early stages. Early Android developers have had to deal with the typical obstructions related with a new platform: frequently revised SDKs, lack of good documentation, and market uncertainties.

On the other hand, developers jumping into Android development now benefit from the first-to-market competitive advantages we’ve seen on other platforms such as BREW and Symbian. Early developers who give feedback are more likely to have an impact on the long-term design of the Android platform and what features will come in the next version of the SDK. Finally, the Android forum community is active and friendly. Enticement programs, such as the ADC, have encouraged many new developers to dig into the platform.

Each new version of the Android SDK has provided a number of substantial improvements to the platform. In recent revisions, the Android platform has received some much-needed UI “polish” both in terms of visual appeal and performance. Although most of these upgrades and improvements were welcome and necessary, new SDK versions often cause some upheaval within the Android developer community. A number of published applications have required retesting and resubmission to the Android Marketplace to conform to new SDK requirements, which are quickly rolled out to all Android phones in the field as a firmware upgrade, rendering older applications out-of-date.

Some older Android handsets are not capable of running the latest versions of the platform. This means that Android developers often need to target several different SDK versions to reach all users. Luckily, the Android development tools make this easier than ever.

**The importance of the Internet in today world**

**What is the Internet?**

The Internet has become a basic part of daily life. While many people rely on it, most have very little or no understanding of how it work. The Internet has often been referred to as the information superhighway. This likens the internet to a system of highways and roadways the allow us to drive almost anywhere, the internet is an interconnected ‘network of networks’ and like a superhighway the internet transports huge amounts of information from one place to another at high speeds along infrastructure such as telephone lines and fibre-optic cable and by satellites. It allows user to connect with out users around the world for business and pleasure.

The web browser is how the user navigates around the Internet, it a piece of software used to locate and display web pages. The most popular browsers are ?? <http://www.w3schools.com/browsers/browsers_stats.asp>

If you are trying to connect to the Internet, you will more th an likely get access thru an Internet Service Provide or ISP. ISPs are companies the allow you to connect to there servers and in turn are connected to the Internet. They provide the user with all the necessary requirement to connect to the internet.

A router is a device that sits on a network; it is used to direct messages to their final destination. Routers have become important due to the fact that messages do not follow the same path to their final destination. As the Internet grew, the volume of traffic on the network grew and there could offer be a backlog on some path so a router simply sends the data along a different more convenient route.

The backbone of the Internet is made up of three components.

Phone lines and cables make up the first component, which the information travels along.

Network Service Providers (NSPs) provide high-speed Internet access and services to ISPs who pass on these services to their customers. Finally Network Access Points (NAPs) allow messages to transfer from one network to another, allowing connection between different networks. These components work together to facilitate the movement of information around the world.

There are a number of other different technologies that makes the Internet work, these technologies are how computers send, receive and understand messages:

Protocols: these are rules and procedures that allow messages to navigate to its chosen destination. These rules and procedures are referred to as protocols in Internet jargon. Protocols describes a set of rules for encoding and decoding information so that messages can be exchanged between computers so both computers can understand the message meaning.

TCP/IP: which stands for Transmission Control Protocol/Internet Protocol provides a procedure for allowing different platforms such as Mac OS or Windows to communicate as well as different software applications to communicate. Even though TCP and IP are two different protocols, the terms TCP/IP is used to refer to a set of protocols that include many different protocols.

Packets: Messages are broken down into smaller piece of information called packets, because there is limited room for message transmission along the Internet. Once the message is broken down into packets, each packet is transmitted individually. The packets can even be sent along different routes by a router; upon arrival the receiving computer reassembles the packet to form the original message. Each packet is given a header that contains information such the receiver’s address, the sender’s address and how to combine the packets to form the original message. This information is called the packet header.

TCP (Transmission Control Protocol) is used to break the messages down into packets and when the receiver’s get the packets it uses TCP to put the packets back together into the original message

IP (Internet Protocol) is used to make sure the message get a proper destination. This is achieved by using URL (Uniform Resource Locator) know as a domain name which is actually verbal code for a numerical address the signifies the location of the receiving computer

**Trends**

The rise in “apps”

**Mobile Content providers**

Eircom (eMobile)

O2

Meteor

Vodafone

3 Ireland

**Data transmissions and speeds**

2G

3G

4G

**Making content accessible via api’s/web services**

API stands for application programming interface. An API can provide a hook for colleagues, partners, or third-party developers to access data and services to build applications rapidly. There are APIs that are open to any developer, APIs that are open only to partners, and APIs that are used internally to help run the business better and facilitate collaboration between teams.

Technical definition: An API is a way for two computer applications to talk to each other over a network (predominantly the Internet) using a common language that they both understand.

APIs follow a specification, meaning:

* The API provider describes exactly what functionality the API will offer
* The API provider describes when the functionality will be available and when it might change in an incompatible way
* The API provider may outline additional technical constraints within the API, such as rate limits that control how many times a particular application or end user is allowed to use the API in a given hour, day, or month
* The API provider may outline additional legal or business constraints when using the API, such as branding limitations, types of use, and so on
* Developers agree to use the API as described, to use only the APIs that are de- scribed, and to follow the rules set out by the API provider  In addition, the API provider may offer other tools, such as:
* Mechanisms to access the API and understand its terms of use
* Documentation to aid in understanding the API
* Resources such as example programs and developer communities to support those using the API
* Operational information about the health of the API and how much use it is getting

**Reference:**

***Apress Beginning Android Tablet Programming, Starting with Android Honeycomb for Tablets (2011)***

***Addison-Wesley Android Wireless Application Development 2nd (2011)***

***APIs: A Strategy Guide (2011)***