Market Research on US Software Industry

(Evolution from the last 30 yrs)

Contents

1. Introduction

Software Industry, consists of that part of computer programming activity that is traded between software-producing organizations and corporate or individual software consumers. Traded software represents only a fraction of domestic software activity, whose extent cannot be reliably estimated, since much computer programming takes place within firms and its value is not captured[1] by the industrial census or software industry analysts.

The traded software industry consists of three main sectors: programming services, enterprise software products, and shrink-wrapped software products.

There are four general types of computer software which includes system software, middleware, application software, and engineering software. Many of the major software companies have been consolidating or acquiring other companies to provide products or services in multiple fields. For instance, rising demand for open source and On-Demand software, has led many larger companies to acquire smaller companies that specialize in these services. Maturity in more traditional software markets such as enterprise software have also contributed to consolidation.

There are more than 100,000 software and information technology (IT) services companies in the United States, and more than 99 percent are small and medium-sized firms (i.e., under 500 employees). This total includes software publishers, suppliers of custom computer programming services[2], computer systems design firms, and facilities management companies. The industry draws on a highly educated and skilled U.S. workforce of nearly two million people, a number which has continued to grow during the past decade.

The International Data Corporation (IDC) estimates that U.S. demand for software will increase more than 7 percent to $163.9 billion in 2012, and that demand for information technology services will rise 4.2 percent to $235.6 billion from the previous year. IDC surveys show that infrastructure projects are high priorities for U.S. businesses, and interest is growing rapidly in collaborative tools, green information technology, cloud computing and mobile applications.

[1] http://www.answers.com/topic/software-industry
[2] http://selectusa.commerce.gov/industry-snapshots/software-and-information-technology-services-industry-united-states

Industry Subsectors

Cloud Computing Services: The IDC expects that global revenue from public cloud computing services will grow four times as fast as information technology spending generally, increasing by 27.6 percent year-on-year from $21 billion in 2010 to more than $76 billion in 2015. The U.S. market currently represents about half of this demand, and U.S. companies routinely dominate the annual rankings of cloud services providers. Software vendors and developers will benefit from this expansion as Software-as-a-Service (SaaS) products are hosted on clouds.

Entertainment Software: Combined revenues in entertainment software from computer and video games expanded by 10 percent from 2005 to 2009 to $10.5 billion. The subsector employs more than 120,000 people directly and indirectly.

Electronic Commerce: Use of the Internet for selling and buying retail products continue to expand across the globe at an exciting pace. The United States is a leader in electronic commerce or "e-commerce". According to statistics compiled by Census, U.S. retail e-commerce spending for 2011 reached $195 billion.

2. Systems Software

The software business has long been one of the leading and most innovative industries in the United States[3]. What began in the late 1950s as a highly specialized technology business servicing a small number of customers that owned large mainframe computers has proliferated to become an industry where software is embedded in nearly everything, from computers and medical equipment to mobile phones and home appliances.

2.1 Evolution[4]

1981: IBM selects Microsoft as the supplier of the operating system software for IBM's personal computer. The selection of Microsoft lays the foundation for further growth of independent software vendors. The IBM design, with Microsoft's Disk Operating System (DOS) at its core, becomes a standard for personal computers developed throughout the 1980s.

1982-84: Hundreds of independent software developers, including Lotus Development and WordPerfect, grow rapidly in response to explosive demand for new types of software applications running on the PC platform.

1984: Apple introduces the Macintosh operating system for its line of personal computers. The system incorporates a graphical user interface (GUI), making user interaction with the computer more intuitive. Developers of Macintosh-based software quickly flood the market with new products.

1989: Outsourcing emerges as a key service business in the United States when Eastman Kodak becomes the first large company to turn over its internal information systems management completely to an outside contractor (IBM). As corporate downsizing gains popularity in the United States, the market for firms offering outsourcing services expands rapidly.

1990: Microsoft launches Windows 3.1, a graphical user interface running on personal computers that quickly becomes a standard around which thousands of PC-based software applications are developed. By late 1994, over 65 million copies of Windows are installed on personal computers around the world.

1993: Microsoft builds upon the success of Windows with the release of a network-based operating system, Windows NT. This product release reflects the rapidly growing demand for distributed computing platforms, particularly client/server-based systems.

CURRENT STATE OF THE INDUSTRY

The software industry continues to evolve rapidly, expanding due to innovative new product and service applications (apps), hardware platforms, delivery mechanisms and growing global markets. Four key trends dominate the industry today:

[3] http://www.protiviti.co.in/en-US/Pages/A-Profile-of-Software-Industry-Risk.aspx
[4] http://www.usitc.gov/publications/332/working_papers/software.pdf

1. Innovation and Product Development

The first trend is the fast pace of innovation and product development that is creating many new applications, services and platforms on which software is used. The most visible of these is the rise of SaaS (Software as a Service), which started off with standardized off-the-shelf, packaged applications but has moved into customized apps built to the specific requirements of a company. SaaS applications for the business market have been rapidly expanding because many companies perceive them as offering a lower total cost of ownership, potentially reducing the need for in-house IT staff to manage the applications. A variant of SaaS is PaaS (Platform as a Service), whereby customers or third party software companies add their own applications to an associated SaaS platform (e.g., Salesforce Appxchange and NetSuite SuiteFlex).

2. Mobile Applications

The second trend is the burgeoning field of mobile applications, whose growth is being driven by the flurry of new platforms and readily available low-cost devices like smart phones and tablet computers. Businesses and consumers are turning to mobile apps because of their ease of use and constant availability. Even large IT organizations are integrating mobile apps to increase the productivity within their company.

3. Mergers and Acquisitions (M&As)

The third trend in the industry is an uptick in mergers and acquisitions (M&As). A distinguishing factor of today's M&As, however, is that many of them are oriented towards vertical integration between hardware and software companies. In some cases, it is a hardware firm acquiring a software firm, such as HP's recent announcement of its purchase of Autonomy. In other cases, it is the opposite, such as Oracle's purchase of Sun. Such M&As are driving change in the software industry as companies re-architect themselves to become not just software developers, but "providers of full solutions" for their clients.

4. New and Expanding Markets

The final trend is the meteoric rise of new software developers who appear in the market and rapidly define a brand new playing field. As in the dot-com era, several upstart companies have managed to achieve enormous success by initially hitting it big with a single application, such as game developer Zynga with its popular Farmville game, and then building on that initial success. Social media companies have gone down this path as well. These developers often drive established companies to compete with them as well as inspire other new upstarts to enter the market, creating a virtual cycle of growth and innovation.

2.2 Trends

System software provides an interface between the hardware and application software, for example operating systems and network management software.

Microsoft (MSFT)

Red Hat (RHT)

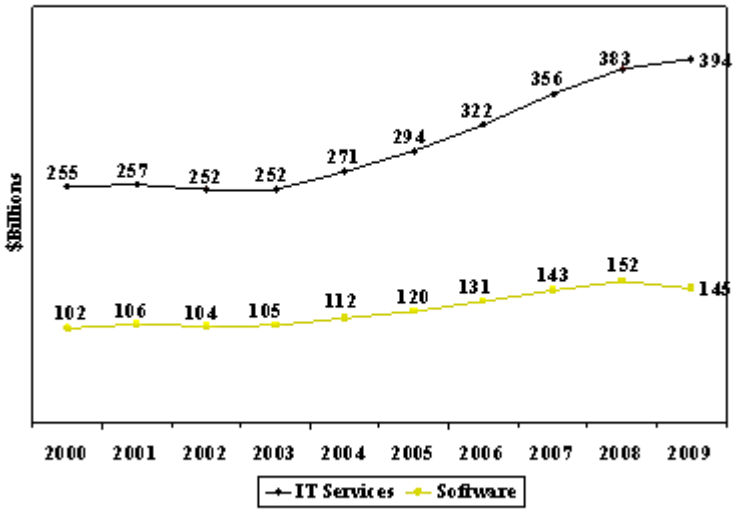
Oracle (ORCL)

Hewlett-Packard (HPQ)

- Windows[5] remains the most popular platform with a 49% share of filters. Web-based (aka SaaS or Cloud Computing) is #2 with 26%, and Mac is #3 with 11%.
- The same thing applies to number of users. If the number of software users is just one, then Windows is the choice 65% of the time, with web-based coming in second at 15%, and Mac third at 13%. When 2-9 users are involved, Windows and Mac interest decline to 56% and 8% respectively, while web-based interest grows to 28%. At 10 or more users, web-based interest grows to 35-40% interest.
- When comparing 2011 to 2010, interest in web-based solutions grew by 2%. Interest in Mac solutions grew by 10%. The main platform that took a hit was not Windows, but instead Linux/Unix, which declined by 16%.

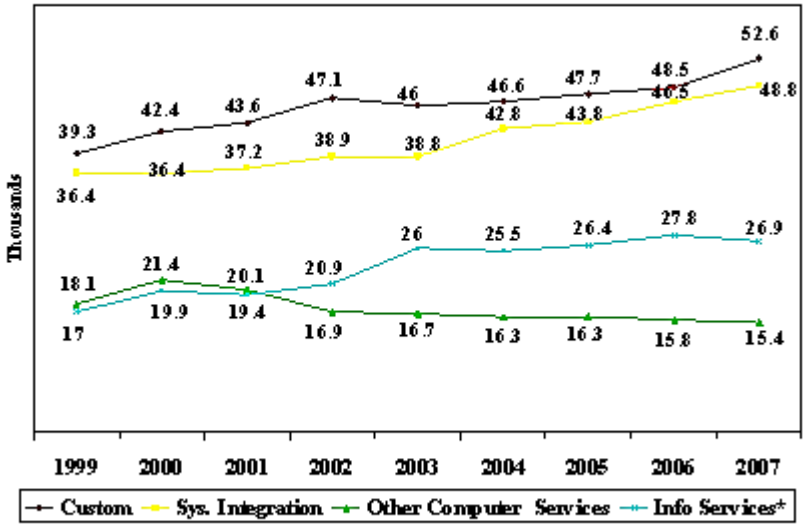
[5] http://www.forbes.com/sites/tjmccue/2012/04/04/14-billion-software-as-a-service-industry-growth-influences-maker-companies/

U.S. Packaged Software and IT Services Industries Revenues
2000 to 2009 (NAICS)



Source: Census Service Annual Survey

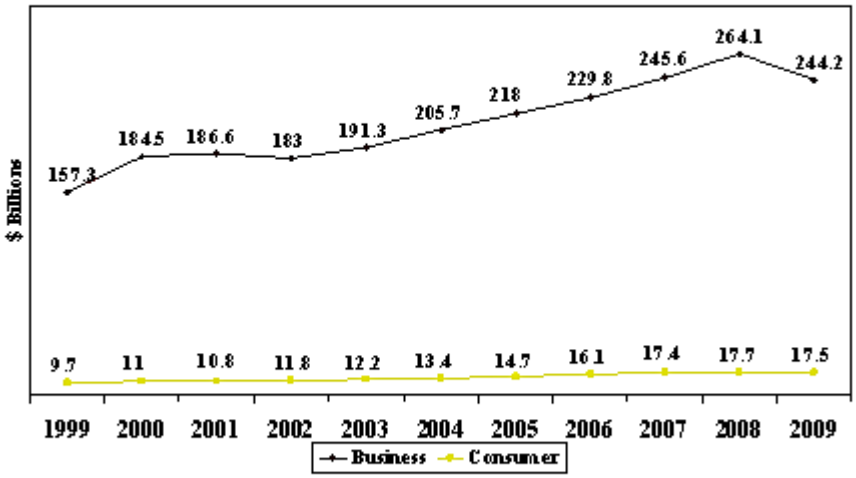
U.S. IT Services Industries Establishments
1999 to 2007 (NAICS)



Source: County Business Patterns, Census

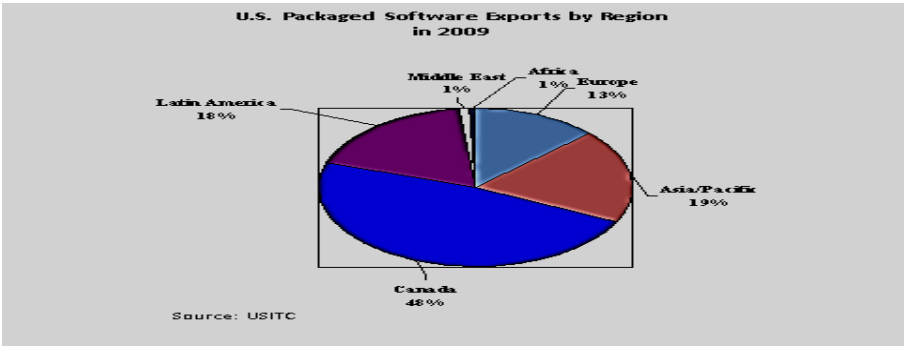
*Includes DP Services, ISPs, Internet Pub.

U.S. Business and Consumer Investment in Software
1999 through 2009*

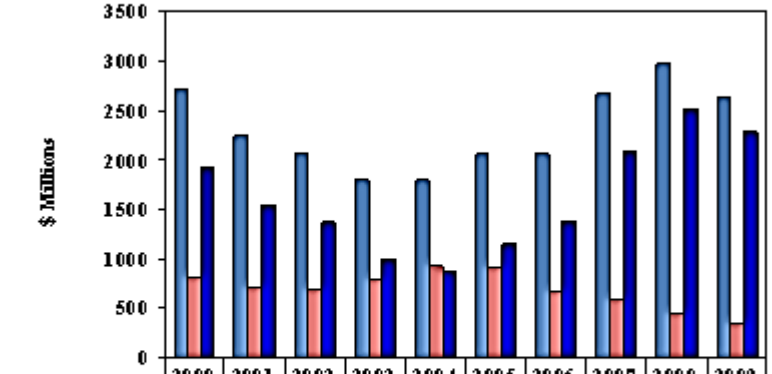


* Excludes software embedded or bundled in computers or other equipment.

Source: BEA

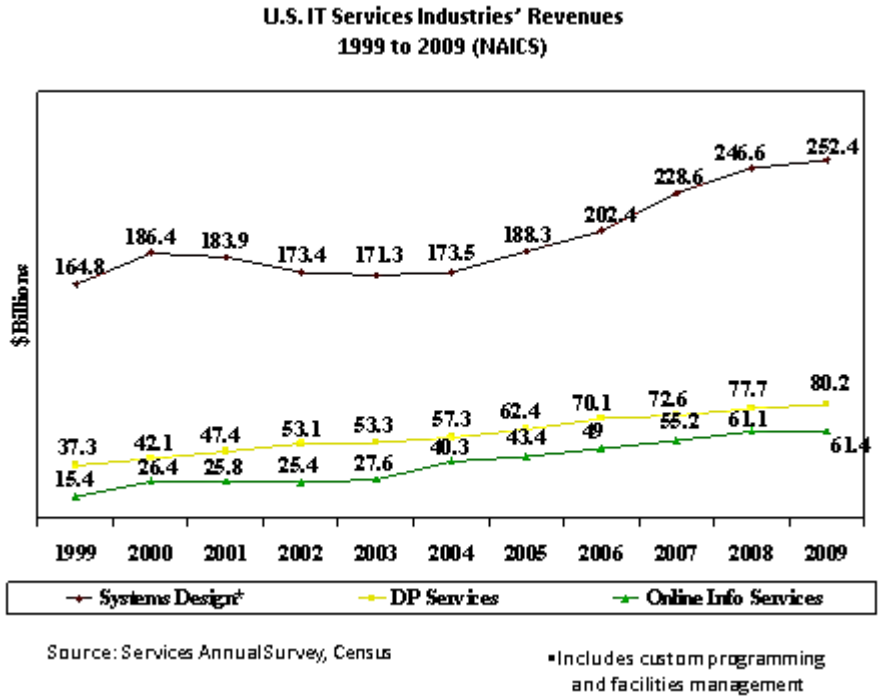


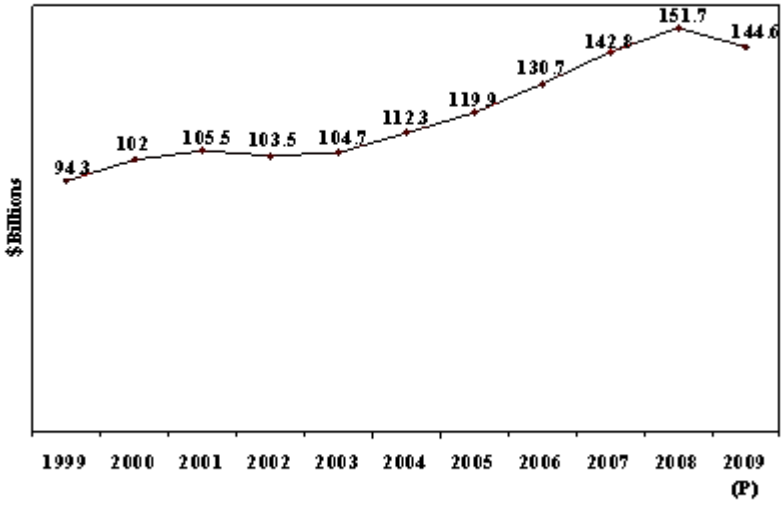
U.S. Packaged Software Trade
2000 to 2009 (HTS)



	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009
Exports	2728	2249	2069	1795	1795	2062	2059	2674	2979	2631
Imports	804	712	700	801	926	908	679	586	453	344
Trade Balance	1924	1537	1369	994	869	1154	1380	2088	2526	2287

Source: USITC



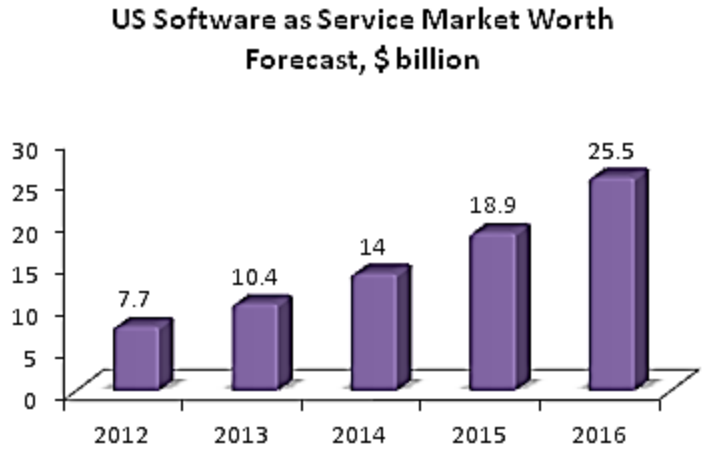


In worldwide, according to the report from MarketLine, the software market exceeded $265 billion in 2010. Market growth is expected to exceed 6% yearly between 2010 and 2015, bringing the market to almost $357 billion. While the US software market reached almost $101 billion in 2010. The market is expected to record almost 5% yearly growth between 2010 and 2015 to reach $126 billion.[6] Network and database management are the leading segment,

[6] http://www.businessvibes.com/blog/industry-insight-software-industry-us

reaching almost $25 billion in 2010, which represents almost 25% of the overall US software market.

Network and database management sales proved the most lucrative for the US software market in 2010, with total revenues of $24.7 billion, equivalent to 24.5% of the market's overall value. The performance of the market is forecast to accelerate, with an anticipated CAGR of 4.6% for the five-year period 2010 - 2015, which is expected to drive the market to a value of $125.9 billion by the end of 2015. Currently, about 75% of the world's largest software companies have based in the United States. The largest software companies in the United States are Microsoft, IBM and Oracle. Together they also form the top 3 software firms in the world.



2.3 Operating System

In the early 1980s, IBM designed the IBM PC and looked around for software to run on it. People from IBM contacted Bill Gates to license his BASIC interpreter. They also asked him if he knew of an operating system[7] to run on the PC. Gates suggested that IBM contact Digital Research, then the world's dominant operating systems company. When IBM came back, Gates realized that a local computer manufacturer, Seattle Computer Products, had a suitable operating system, DOS (Disk Operating System). He approached them and asked to buy it (allegedly for $50,000), which they readily accepted. Gates then offered IBM a DOS/BASIC package, which IBM accepted. IBM wanted certain modifications, so Gates hired the person who wrote DOS, Tim Paterson, as an employee of Gates' fledgling company, Microsoft, to make them. The revised system was renamed MS-DOS (MicroSoft Disk Operating System) and quickly came to dominate the IBM PC market. A key factor here was Gates' (in retrospect, extremely wise) decision to sell MS-DOS to computer companies for bundling with their hardware, compared to Kildall's attempt to sell CP/M to end users one at a time (at least initially).

[7] http://www.informit.com/articles/article.aspx?p=24972

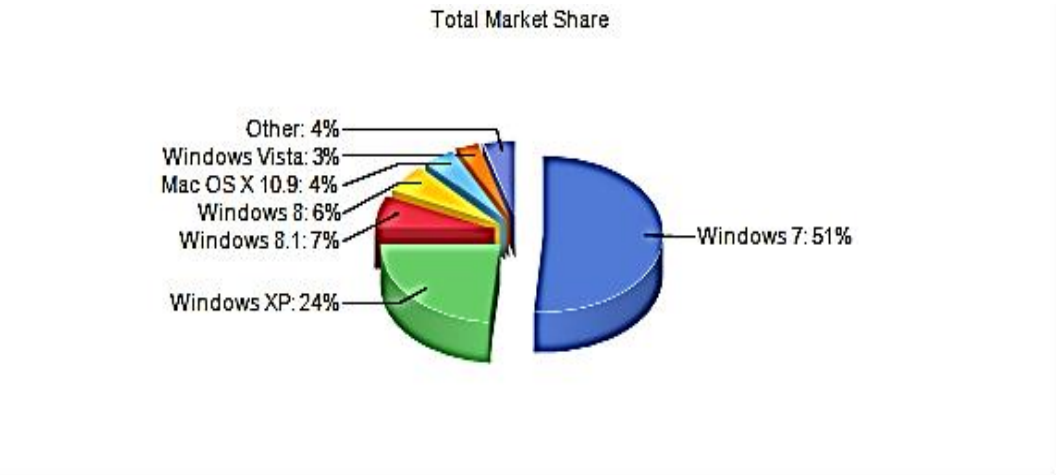
One day, Steve Jobs, who co-invented the Apple computer in his garage, visited PARC, saw a GUI, and instantly realized its potential value, something Xerox management famously did not (Smith and Alexander, 1988). Jobs then embarked on building an Apple with a GUI. This project led to the Lisa, which was too expensive and failed commercially. Jobs' second attempt, the Apple Macintosh, was a huge success, not only because it was much cheaper than the Lisa, but also because it was user friendly, meaning that it was intended for users who not only knew nothing about computers but furthermore had absolutely no intention whatsoever of learning.

When Microsoft decided to build a successor to MS-DOS, it was strongly influenced by the success of the Macintosh. It produced a GUI-based system called Windows, which originally ran on top of MS-DOS (i.e., it was more like a shell than a true operating system). For about 10 years, from 1985 to 1995, Windows was just a graphical environment on top of MS-DOS. However, starting in 1995 a freestanding version of Windows, Windows 95, was released that incorporated many operating system features into it, using the underlying MS-DOS system only for booting and running old MS-DOS programs. In 1998, a slightly modified version of this system, called Windows 98 was released. Nevertheless, both Windows 95 and Windows 98 still contain a large amount of 16-bit Intel assembly language.

Another Microsoft operating system is Windows NT (NT stands for New Technology), which is compatible with Windows 95 at a certain level, but a complete rewrite from scratch internally. It is a full 32-bit system. The lead designer for Windows NT was David Cutler, who was also one of the designers of the VAX VMS operating system, so some ideas from VMS are present in NT. Microsoft expected that the first version of NT would kill off MS-DOS and all other versions of Windows since it was a vastly superior system, but it fizzled. Only with Windows NT 4.0 did it finally catch on in a big way, especially on corporate networks. Version 5 of Windows NT was renamed Windows 2000 in early 1999. It was intended to be the successor to both Windows 98 and Windows NT 4.0. That did not quite work out either, so Microsoft came out with yet another version of Windows 98 called Windows Me (Millennium edition).

An interesting development that began taking place during the mid-1980s is the growth of networks of personal computers running network operating systems and distributed operating systems (Tanenbaum and Van Steen, 2002). In a network operating system, the users are aware of the existence of multiple computers and can log in to remote machines and copy files from one machine to another. Each machine runs its own local operating system and has its own local user (or users).

Desktop Operating System Market Share[8]



Operating System	Total Market Share
Windows 7	51.21%
Windows XP	23.89%
Windows 8.1	7.09%
Windows 8	6.28%
Mac OS X 10.9	4.29%
Windows Vista	3.02%
Linux	1.67%
Mac OS X 10.6	0.78%
Mac OS X 10.8	0.65%
Mac OS X 10.7	0.60%
Mac OS X 10.1	0.22%
Mac OS X 10.5	0.15%
Windows NT	0.06%
Mac OS X 10.4	0.04%
Windows 2000	0.03%
Mac OS X (no version reported)	0.01%
Windows 98	0.00%
Win64	0.00%

[8] http://www.statista.com/statistics/272667/market-share-held-by-operating-systems-in-the-us-since-2009/

2.4 RDMS

Relational Database Management Systems (RDBMS) Vendors					
Total Software Revenue, Worldwide, 2010-2011 (Millions of U.S. Dollars)					
Vendor	(Ctrl) 2010	2011	Share of 2010	Share of 2011	Growth 2011
Oracle	9,990.5	11,787.0	48.2%	48.8%	18.0%
IBM	4,300.4	4,870.4	20.7%	20.2%	13.3%
Microsoft	3,641.2	4,098.9	17.6%	17.0%	12.6%
SAP/Sybase	744.4	1,101.1	3.6%	4.6%	47.9%
Teradata	754.7	882.3	3.6%	3.7%	16.9%
Other Vendors	1,315.3	1,389.7	6.3%	5.8%	5.7%
Grand Total	20,746.6	24,129.5	100.0%	100.0%	16.3%
Source: Gartner (March 2012)					

3. Applications Software

Application Software is the software used to accomplish a broad range of different tasks with a computer. The primary subcategories of Application Software includes Database Software, Office Suite Software[9] (Word Processing, Presentations, Spreadsheets), and Enterprise Software (Business Intelligence, Content Management, Customer Relationship Management, Data Security, HR Management Knowledge Management, Predictive Analytics/Forecasting, and Price Optimization).

Application Software makes up just under half of all packaged software sales in the world.

Adobe Systems (ADBE)
Epicor Software (EPIC)
ILinc Communications (ILC)
Intuit (INTU)
Lawson Software (LWSN)
Magic Software Enterprises (MGIC)
Microsoft (MSFT)
MicroStrategy (MSTR)
Oracle (ORCL)
Progress Software
PROS Holdings Inc (PRO)
QAD (QADI)
Salesforce.com (CRM)
SAP AG (SAP)
Software AG (SOW-FF)

[9] http://www.wikinvest.com/industry/Computer_Software

3.1 Trends

The Rise of Open Source Software

The IDC, an information market research firm, projected that growth of Linux, an open source operating system, would be about double the growth of windows between 2007 and 2011. As its name suggests, Open source software has source code that is "open" and readily available and can be changed by anyone. Open source software is usually sold for a small license fee, but the majority of open source revenues come from selling support and training. Open source is a much cheaper option than traditional application software offerings, but raises security concerns for many companies. However, major traditional competitors such as Microsoft and Oracle realize the threat of open source and are either developing their own open source software or selling support to existing open source software.

Cloud Computing Could Change the Way Software is Delivered

Traditionally, application software is sold for a license fee and installed on the computer. Cloud computing completely changes that. Cloud computing provides On-Demand software, which means a central server would be responsible for the delivery and maintenance of your software over the internet. In most cases, the end user would pay a subscription fee to use this software. Cloud computing eliminates a lot of costs for companies because a company would no longer need to pay for on-site servers, maintenance fees, or licensing fees. Traditional software companies like Oracle (ORCL) , SAP AG (SAP) , and Lawson Software (LWSN) are threatened by the adoption of cloud computing.

3.2 Evolution of Enterprise Software

The term 'enterprise software' describes the applications that large companies use to conduct line-of-business operations such as accounting, business intelligence (BI), communication and collaboration, customer relationship management (CRM) and human resources (HR). These tools are traditionally deployed[10] in on-premise data centers, often as multi-faceted enterprise resource planning (ERP) suites from software giants like Oracle, SAP, IBM and Microsoft. 'Enterprise software' also encompasses vertical, industry-specific solutions, which are commonly developed as custom in-house apps that IT departments then need to integrate with commodity enterprise applications or suites.

A major recent systematic change in the environment for traditional on-premise enterprise software has been the post-banking-crisis economic recession. This reduces its revenue-earning potential by causing the delay or cancellation of complex and costly upgrade projects, and by increasing the attraction of more affordable third-party maintenance services (such as Rimini Street). Vendors that fail to adapt to the new economic climate — by cutting their maintenance rates, for example — are likely to find themselves in their very own 'struggle for existence'.

[10] http://www.zdnet.com/the-evolution-of-enterprise-software-an-overview-7000014006/

Adaptive radiation in enterprise software is taking place thanks to the rise of cloud computing. The ability to implement platform and infrastructure services in the cloud quickly and cost-effectively has led to a profusion of scalable pay-as-you-go SaaS applications that now address all areas of enterprise software (see the listings in ZDNet's recent SaaS special feature). The SaaS model is not without its drawbacks (there are often worries about security, outages, compliance, performance, data mobility and integration for example), but the advantages (cost savings, scalability, accessibility, easy upgrades and resilience) are increasingly compelling for many organisations.

For the foreseeable future, then, companies will be faced with managing existing on-premise enterprise applications as cost-effectively as possible while exploiting what's on offer in the cloud (be it private, public or hybrid cloud).

The new world of enterprise apps

Consumerisation, widespread connectivity and cloud computing are shaping more flexible working patterns, in both space and time. Employees want to use devices, software and services that they're comfortable with, wherever and whenever the need should arise. Meanwhile, businesses want the ability to select and integrate best-of-breed enterprise applications rather than becoming (or remaining) locked into inflexible and expensive suites.

And, touching all aspects of the enterprise, there's a growing need to capture, organise and exploit a mass of 'Big Data', both structured and unstructured, that can give internal and external business processes an edge over the competition.

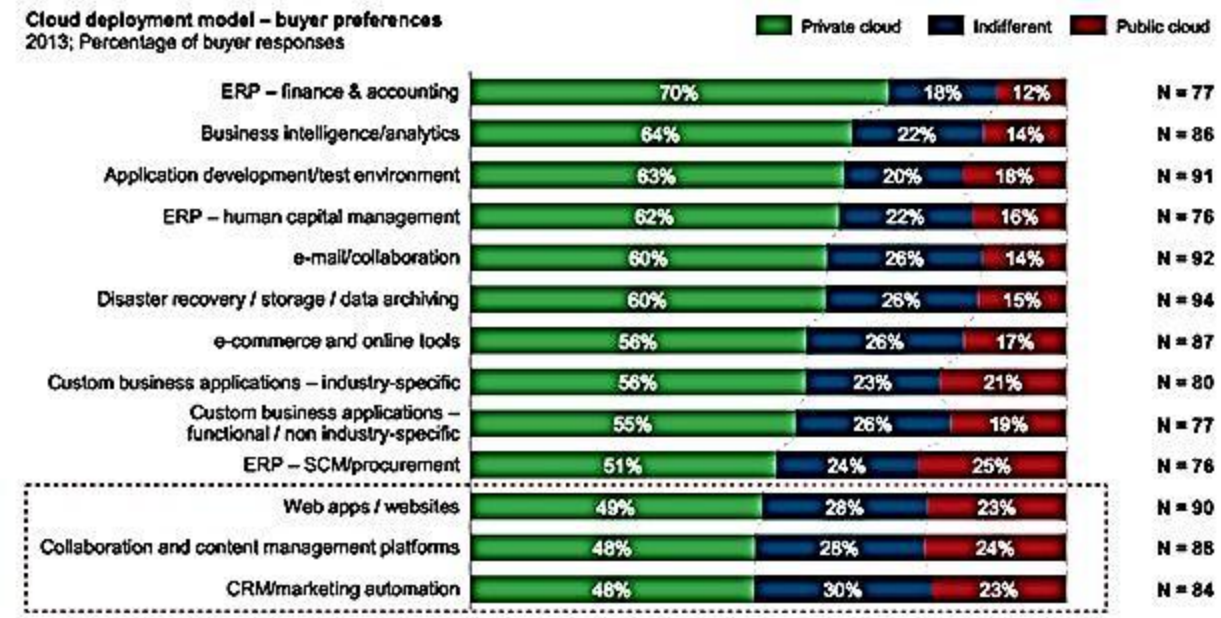
This is the current 'climate' for enterprise software. Here's the 'weather forecast':

More cloudy

Enterprises are increasingly adopting cloud technology, but what does this adoption look like at a more granular level? Management consulting firm The Everest Group recently addressed this question in a survey timed to coincide with the April 2013 Cloud Connect conference. The survey's 302 respondents were divided among enterprise/cloud service buyers (33%), third-party cloud solution providers (30%) and advisory professionals (37%), most of whom (79%) were from North America.

Enterprise buyers exhibited a strong preference for private cloud deployments for most workloads — the highest being for ERP/finance & accounting, with 70 percent favouring the in-house option. The biggest biases towards public cloud deployment were for less mission-critical workloads such as collaboration and content management platforms, web apps and websites, and CRM/marketing automation.[11]

[11] The Everest Group Cloud Connect Enterprise Cloud Adoption Survey 2013 shows that buyers currently favour private clouds for most workloads.



3.3 Enterprise Software Products

During the 1980s, the software products market finally matured and grew at a sustained annual compound rate of 30 percent—from aggregate sales of$2.7 billion in 1980 to over $30 billion by the end of the decade. As with programming services, the software products industry was low in concentration. For example, a survey in 1982 showed that the top fifty or sixty firms accounted for only 50 percent of sales, leaving the remainder to approximately two thousand medium and small firms. (By contrast, in the mainframe computer industry, less than twenty firms accounted for virtually the entire industry, and one firm, IBM, for more than 50 percent.) The software industry was (and is) sometimes characterized as being like "boulders, pebbles, and sand"; that is, a few tens of global players, a few hundred second-tier firms, and thousands of very small firms with a dozen or fewer employees.

The leading firms have generally grown through consolidation or by dominating a particular software genre through organic growth. Consolidation has been particularly important in gaining market share in the software industry for two reasons: it has proved extremely difficult to imitate a successful software product that may have taken years to evolve, and software products have proved insensitive to price, buyers valuing reliability and security of supply above cost. Among the consolidators, Computer Associates, founded in Islandia, New York, in 1976, has been the most prominent and successful. Following its initial public offering in 1981, Computer Associates set out on a course of acquisitions that made it the largest software vendor by 1990 and it has retained its place as one of the top three vendors ever since (usually jockeying for position with Microsoft and Oracle). Computer Associates' acquisitions have included other consolidators, so that if one were to construct a "family tree" of the firm it would contain several hundred names (including ADR and Informatics, mentioned above).
By the 1990s, two major software genres accounted for perhaps a half of all corporate software sales: relational database software and enterprise resource planning (ERP) software. Relational database technology emerged in the early 1970s in the research environment of IBM's San Jose Research Laboratory and the University of California, Berkeley. Relational technology was a

major, though technically challenging, advance over earlier database systems. The technology was first exploited by northern Californian software entrepreneurs, including Oracle, founded in Belmont, California, in 1977. The region still remains the world center of relational technology. Oracle has consolidated its early start advantage, out-manoeuvring and out-growing all competitors, frequently vying with Computer Associates as the number-two software company. ERP software emerged in the 1980s as a single-product solution to replace the aggregation of numerous application products that computer users typically had to use in the 1970s and 1980s. The leading vendor is SAP, a German firm, which invented the ERP concept in the early 1980s; although there are now several U.S. competitors, none has yet overcome SAP's first-mover advantage. SAP is the only non-U.S. software-product firm in the top ten (and one of only a handful in the top 100).

3.4 The Internet Era

The diffusion of the PC in the 1980s dramatically changed the working lives of office employees: senior managers began to type their own memoranda, while junior executives spent a disproportionate amount of their days tinkering with spread sheets. The typewriter was consigned to the dustbin of history, while typists were promoted to being general administrators.[12]

Since the mid-1990s, the Internet has had an equally dramatic effect on the lives of office workers and increasingly the domestic computer user. Although the Internet had been in widespread use in technical communities since the early 1980s, it was the introduction of the World Wide Web in the early 1990s that made the Internet accessible to ordinary users. The enabling software technology of the World Wide Web was the Web browser used in a desktop computer, of which most users are conscious, and the invisible software in "servers" that made Web pages available to remote users on demand. The first and most important supplier of software for the Web was the Netscape Communications Corporation of Mountain View, California, founded in 1994. Run by a 24-year-old wunderkind, Netscape grew from nothing to a billion-dollar corporation in two years. In 1996, however, Microsoft introduced a competing product, Internet Explorer, which quickly achieved a dominant market share. Microsoft's hardball tactics in this achievement added weight to its antitrust prosecution.

At first, the Web was a largely passive experience, but in the late 1990s it became increasingly interactive and participative, with the provision of financial services, travel information, entertainment including pornography, auctions and retail services, and information products of every kind. In a couple of years new brands became household names—Motley Fool, Ameritrade, Yahoo, Travelocity, Amazon, and many more. Some of these are new enterprises, while others are new initiatives from old established firms. Just as the Web is changing the old order of information services and retailing, it is even more profoundly changing the world of software. For example, the metaphor of "shrink-wrapped" software is breaking down as software products are increasingly supplied by electronic download. Some industry observers predict that the concept of a software product or artefact will become obsolete, and programs will be supplied online, on demand, and metered according to use. Whether or not this comes to pass, the phrase software industry will continue to be the collective term for firms engaged

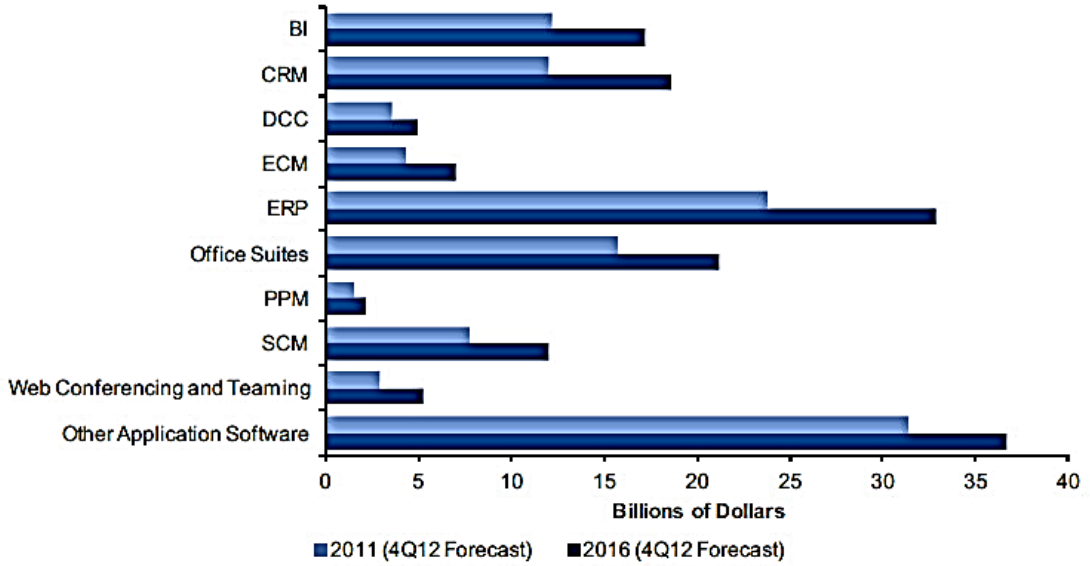
[12] http://www.answers.com/topic/software-industry

in supplying programming goods and services, in whatever way the technology of the day demands.

3.5 Growth/Forecast

- Enterprise sales of ERP systems will grow to $32.9B in 2016, attaining a 6.7% CAGR in the forecast period of 2011 to 2016. CRM is projected to be an $18.6B global market by 2016, attaining a CAGR of 9.1% from 2011 to 2016. The fastest growing category of enterprise software will be Web Conferencing and Team, growing at a 12.4% CAGR through the forecast period. The following graphic compares 2011 actual sales and the latest forecast for 2016 by enterprise software product category.

Enterprise Application Software Forecast Comparison by Segment, Total Software Revenue, Worldwide, 2011 and 2016



DCC = digital content creation
ECM = enterprise content management
PPM = project and portfolio management

Source: Gartner (January 2013)

- Enterprise application software spending will reach $158B by 2016, with 20.8% being spent on ERP systems, 13.4% for Office Suites and 11.7% on CRM. The following table breaks out spending by category and CAGRs for 2011 – 2016.

Enterprise Application Software Forecast Comparison by Segment (4Q12 Update), Total Software Revenue, Worldwide, 2011 and 2016 (Billions of Dollars)

	2011 (Actuals)	2016 (4Q12 Forecast)	Five-Year CAGR (%)
BI	12.2	17.2	7.0
CRM	12.0	18.6	9.1
DCC	3.6	5.0	7.0
ECM	4.3	7.0	10.1
ERP	23.8	32.9	6.7
Office Suites	15.7	21.2	6.4
PPM	1.5	2.2	7.8
SCM	7.7	12.0	9.1
Web Conferencing and Teaming	2.9	5.2	12.4
Other Application Software	31.4	36.7	3.2
Total	**115.1**	**158.0**	**6.5**

Source: Gartner (January 2013)

- SaaS and cloud-based business application services revenue will grow from $13.5 billion in 2011 to$32.8 billion in 2016, achieving a five-year CAGR of 19.5%.

- By 2014, IT organizations in 30% of Global 1000 companies will broker (aggregate, integrate and customize) two or more cloud services for internal and external users, up from 5% today.

- In 2012, 36% of CRM applications in the CRM software market were delivered on software as a service (SaaS) infrastructure. Gartner forecasts that SaaS use will increase to 40% in 2013, and that the crossover point to more than 50% SaaS in the CRM market will be reached during 2016.

- SaaS-based CRM will grow three times as fast as on-premises solutions. The compound annual growth rate (CAGR) for all CRM software is forecast to be 9.1% from 2012 through 2016, and 15.4% for SaaS-delivered CRM during the same period.

4. Applications Tools/Languages

4.1 Evolution of programming languages

- Years 80: Experimenting other ways including objects. ML. Smalltalk. On computers, we now use C, Pascal, Basic compiled.

- Years 90: Generalization of object-oriented programming[13] with the performance of microcomputers. Java, Perl, Python languages in addition to microphones.

- 2000s: Internet Programming (and future innovations, see end of text).

- Years 2010: Concurrency and asynchronicity. JavaScript and Go languages among others help to create online fluid applications.

Ada - 1980+ - Nickname of Ada Byron de Lovelace, first woman to program

Designed by a committee headed by Jean Ichbiah, for the U.S. Department of Defense.

Inspired by Pascal and Algol

Introduces GENERICITY of algorithms and a kind of primitive object orientation, but becomes really object oriented later-- Introduces PACKAGES, that are independent modules.

C++ - 1981-1986

Bjarne Stroustrup.

Description

Object oriented version of C.

Introduces OPERATOR OVERLOADING. Methods may be inline.

Use // for one-line comment, that comes from BCPL, of which C is a successor!

Further, multiple inheritance and template (generic classes or functions) has been implemented.

Objective C, invented by Brad Cox in 1984, is another object oriented version of C, inspired by smalltalk. No operator overloading. Used to write Next Step, the operating system of the Next computer, it has become the programming language of Apple with the return of Steve Jobs and therefore that of iPhone.

[13] http://www.scriptol.com/programming/history.php-(Denis Sureau)

Standard ML - 1984

R. Milner, University of Edinburgh and Cambridge and Irina.

Search for Standard ML Moscow on a search engine.

An implementation of ML.

Eiffel - 1985

Bertrand Meyer.

Description

Procedural language fully object oriented, implementing persistency and programming per contract (using precondition and post condition on functions). Was designed for security of software.

Compiled in C. May be interfaced with other languages. Has features of functional languages, generic classes, garbage collector.

An open source version exist, Sather, (from the name of a tower at Berkeley).

GAP - 1986 - Groups, Algorithms and Programming

Johannes Meier, Werner Nickel, Alice Niemeyer, Martin Schönert and others.

The language has been defined to program mathematical algorithms.
It is interpreted, interactive an untyped. List and records are complex variables.
The syntax is that of Pascal with some differences. Comments for example are introduced by #.
An end of bloc is denoted by inverted keywords: if fi, do od.
The for loop has the forms: for in list, for in from to.
The language has procedures and functions.
What made it apart if that variables point out values and not memory addresses, and the form of a definition of function is as a call: x := function(arguments) body.
A function may be embedded inside another function.

Miranda - 1989 - From the name of a Shakespeare's heroin (Miranda, means for admirable in latin)

D. Turner.

Inspired by Sasl and ML. Lazy evaluation: arguments of functions are evaluated only when they are used. Embedded pattern-matching, modules.

Caml - 1987 - Categorical Abstract Machine Language

Suarez, Weiss, Maury.

Inria

Caml and Objective caml in 1996, has implemented ML.

Perl - 1987 - Practical Extracting and Report Language

Larry Wall.

Destinated to replace the command line language of Unix, Sh, Sed and Awk, it kept the same ugly syntax. Used mainly for system administration, CGI scripts.

Includes lists and associatives arrays. The FOREACH control structure allows to scan lists.

Oberon - 1988

Niklaus Wirth.

Successor of Modula 2 (and Pascal).

Several commonly used constructs are suppressed to reduce the risk or error! A garbage collector is added to.

Haskell - 1990 - Nickname of a logician, Haskell Curry

Purely functional language. Inspired by Miranda and Sasl.

Functional arrays, pattern matching.

ABC 1980-90 **-** ABC (equivalent to EZ in english)

CWI - Meertens, Pemberton and Guido Van Rossum.

Scripting language elaborated at CWI in Netherlands, and the goal of which was to become a successor to Quick Basic or scripting languages of Unix.

Perhaps the first to use INDENTING to denote statements of a block: no markers as begin/end.

Another innovation, there is no file management, but rather persistency of the global variables: the value is stored from a session to another!

There are five types: number, string, list, composed (structure without fieldnames), array.

Python - 1991 - From the english TV movie "Monty Python Flying Circus"

Guido Van Rossum.

Description

Scripting language with dynamic types. This is a replacement to Perl.

Inspired by ABC, but is extensible with C libraries, and object oriented.

As ABC, used evolved types: tuple, list, dictionary.

The slicing operator [a : b] allows to extract a sub-list from a list.

There is a version that compiles in Java bytecode, jython and ports for .NET.

Pov-Ray - 1991 - Persistence of Vision (title of a mediocre science-fiction book).

D. & A. Collins, and contributors.

Pov-Ray is a language for describing 3D images.

DisCo - 1992 - Distributed Co-operation

Reino Kurki-Suonio.

Disco is a specification language for reactive systems with Pascal-like syntax. Constructs of the language are objects, event-driven functions (named here actions), and relations. A function is activated when a state of the system occurs and may be overwritten, this is named "refinement" in the language. Disco focuses on collective behavior. Layers are modules of the language. It is an system oriented language with objects and behavior (not action oriented as it is said in the presentation).

Ruby - 1994 - As the jewel, analogy with Perl

Yukihiro Matsumoto.

Description

Ruby has been designed as successor to Perl and alternative to Python, to be clearer than the first one, and more object oriented than the second one. The syntax comes from these two languages, it want to be without surprise and natural but may be complex.

There is no new control structure as in Scriptol, but a lot of minor innovations to make the code smaller.

It is an interpreted language easy to extend. Statements are terminated by end of line. Blocks of statement and loop are delimited by "end". Most Python's features are present: associative arrays, iterators...

The originality is the dynamic object feature (adding methods to instances) and scope of variables denoted by a prefix.

Java - 1994 - Java (coffee)

James Gosling and other programmers at Sun.

Description

Conceived at the beginning, in 1991, as an interactive language named Oak, was unsuccessful. But in 1994 has been rewritten for Internet and renamed Java. In 1995 navigators can run applets. In January 1996, Javasoft distributes JDK 1.0, the Java Development Kit.

Java is a object-oriented language, near C++. It compiles in byte code, interpreted on any computer. ..

It is simpler than C++: one class by file, automatic memory management, no pointers. No multiple inheritance nor operator overloading, but integrated multi-tasking.

Unlike C and C++, has only dynamic arrays.

PHP - 1995 - Personal Home Pages Hypertext Processor

Rasmus Lerdorf.

Description

Multi-platforms scripting language, embedded inside HTML.

Near C but not typed. Variables are prefixed by the $ symbol as the shell of Unix or as Perl. The interpreter parses a html page that embeds php code and delivers a pure html page.

An extended library of functions allows webmasters to build dynamic pages.

Microsoft uses an equivalent language under Windows, ASP, near Basic.

JavaScript - 1995 (Has been firstly named Live Script)

Brendan Eich at Netscape.

Description

Scripting language to embedd procedural code into web pages.

May be used to other applications, XML based languages for example.

Share the syntax of C or Java, but with untyped variables. The element of the web page (window, table, etc...) are accessed through the Document Object Model.

UML - 1996 - Unified Modeling Language

Standard by OMG (Object Management Group) - Grady Booch, Jim Rumbaugh, and Ivar Jacobson.

UML is the union of three modelling languages designed by the three authors above. The language uses a graphical notation to design software projects. A source is a diagram expressing objects and their interactions. A model is made of views and the combination of them describes a complete system. The model is abstract and domain-independent.

ECMA Script - 1997

Standard by the European standardization organisation E.C.M.A.

Standard to the language invented by Netscape, to let dynamic HTML pages client-side.

Rebol - 1997 (The design is older) - Relative Expression Based Object Language

Carl Sassen Rath.

Interpreted, extensible scripting language that produces compact code. It is aimed at communication on Internet and distributed computing.

Has 45 types using same operators (Ex: date, money...). Use [ ] to enclose blocks of statements.

C# - 2000 - (C-sharp), want to succeed to C++

Anders Hejlsberg / Microsoft.

Description

This is the main language of the .NET environment, to program software working thought Internet. As Java, it keeps the C syntax, a 34 years old language, with same improvements: garbage collector, no pointer, interfaces, multi-tasking...

C# compiles to intermediate language, the MSIL (MicroSoft Intermediate Language), and uses a multi-languages library, the CLR (Common Language Runtime). The originality of the .NET system is that various language may be compiled to MSIL and share their classes.

Other new features come with this language:

- structs are now special kind of object, passed by values.

- literals are objects also, with methods..

- attributes are descriptive objects attached to elements of the program and used by the runtime.

- properties: methods that may be used as variables (prop = 5 is equivalent to prop(5).

- for each () to scan arrays (new only for Java and C++).

- delegate replaces pointer of functions of C.

There are improvements on Java also:

- event management is improved.

- operator overloading is present.

- simpler access to the native system.

AspectJ - 2001 - Aspect for Java

Palo Alto Research Center.

Description

Aspect J is a Java extension that implements aspect oriented programming. A technique that modularizes crosscutting concerns. The unit is not the class, but a concern, that spans multiple classes. Concern may be, for example, properties; area of interest of a system and AOP describes their relationship, and compose them together into a program. Aspects encapsulates behavior that concerns multiple classes.

Scriptol - By Denis Sureau, 2001

Description

Scriptol (Scriptwriter Oriented Language) is either compiled in PHP or in C++ or native, giving it a great portability. It is both a language for applications, for scripting and to make dynamic web pages.

Blocs of statements and control structures are not closed by "end" or "}" but, as in XML, with the form "/if", " /for", "/while" and so on...

The language has new control structures: "for in", "while let", "scan by", etc... The "composite if"structure eases to implement rules.

Variables and literals are objects. Basic object (number, text, etc...) and compound ones are created by direct assignment of a value or a list of arguments to the name.

Scriptol is destinated to evolve and to have, along classes, other high-level structures to allow programs to be nearest human thought.

Since October 2003, Scriptol allows to use XML as internal data structure.

Scala - February 2004

Description

Ecole Polytechnique Federale de Lausanne

Scala is a pure object oriented language that implements some Python features in Java syntax. It is statically typed and both procedural and functional. It currently runs on JVM and .NET.

Go - By Google, 2009

Description

Created by Google for its own development, but placed in the public domain under a free license, it is designed specifically for compilation speed.

It is a modern version of C++ without header files, with a simplified syntax. Classes are replaced by simple interfaces and inheritance is gone. It brings concurrency and includes a garbage collector, but with no substantial contribution to the design of programming languages.

Julia - 2010

Jeff Bezanson, Stefan Karpinski, Viral B. Shah, Alan Edelman. Sponsored by DARPA.

Description

The authors wanted to implement the best features of all other programming languages: objects, concurrency, homoiconicité, distributed computing, macro, generics. This with the most concise and clear syntax as possible. Julia is a significant step in programming languages.

Julia code is compiled through the LLVM JIT compiler and it runs like an interpreter.

Its main areas of application are scientific programs thanks to the expanded library, cloud with distributed processing and concurrency, and with the ability of a program to change itself, robotics.

Dart - By Google, 2011

Description

To replace JavaScript, Google feels has irrecoverable design flaws (this is not the opinion of all actors in the Web), it is similar to static languages like Java, with classes, single inheritance, typed or dynamic variables. It does progress in features (concurrency, mixins, streams) but is a regression in design with respect to JavaScript whose dynamic features have been a real evolution. JavaScript in version 5 will have classes and inheritance too, which reduces the value of Dart on the browser.

Rust - By Mozilla, 2006-2011

Description

This system language originally created by Graydon Hoare in 2006 and taken in 2009 by Mozilla (its employer) has a compiler since 2011. Its syntax is derived from that of C with additions to manage concurrency and syntactic additions. It is object oriented and generic with polymorphic classes as in Haskell.

It is intended to be safer for the Web and taking better advantage of current processors.

Asm.js - By Mozilla, 2013

Description

This subset of JavaScript allows in combination with other tools to convert applications and libraries written in other languages and use them in the browser. This makes of the browser a universal interface for software working online or offline on all operating systems and all types of devices, from desktop to smartphone.

Swift - By Apple, 2014

Description

Designed to replace Objective-C on Apple's devices including the iPhone and iTab, it removes some of the defects of this ancient language. It has a classic design and is distinguished mainly by changing the name of the elements of language, interfaces becoming protocols, for example.

From the present to the future

We see that after the plethora of dialects of the 70s, the invention of languages stagnated about the syntax. Common languages, even recent ones like Java, C #, PHP, bring no change to the instructions, Go and Dart are even a regression. Only Julia is a real progress and exploits fully capabilities of current computers.
The use of JavaScript is spreading along with web applications offline.

Object-oriented languages

These languages allows to embed code inside HTML page and thus to combine statements and data. PHP, ASP, JavaScript are the most used ones. The .NET platform will allow any language to be embedded into HTML.
Classes are being replaced by dynamic objects and inheritance by combination.

Scripting languages

In the 2000s, several modern scripting languages offer a simpler and intuitive syntax: Python is the most widely used for now but is being replaced by Go. Ruby is mainly used for the Rail library but the trend is in Node and JavaScript.

Markup languages

XML was a major trend in the 2000-2010 but is then dedicated to graphical interfaces. For documents it trends to be replaced by JSON.
Microsoft uses XAML to define graphical interfaces, on the Web with the cross-browsers plug-in Silverlight or locally on .NET.
SVG is a format to embed graphics in webpages supported by all modern browsers. It may be used to design a user interface.

SQL

Thanks to Web applications SQL trends to be more and more popular, and so is now a part of modern programming.

Portability

Languages become more portable thanks to LLVM and Emscriptem. The first for the virtual machine which runs on any system, the second by converting the LLVM code in JavaScript.

4.2 Current Trends

Programming language evolution continues, in both industry and research. Some of the current trends[14] include:

- Increasing support for functional programming in mainstream languages used commercially, including pure functional programming for making code easier to reason about and easier to parallelise (at both micro- and macro- levels)
- Constructs to support concurrent and distributed programming.
- Mechanisms for adding security and reliability verification to the language: extended static checking, dependent typing, information flow control, static thread safety.
- Alternative mechanisms for modularity: mixins, delegates, aspects.
- Component-oriented software development.
- Metaprogramming, reflection or access to the abstract syntax tree
- Increased emphasis on distribution and mobility.
- Integration with databases, including XML and relational databases.
- Support for Unicode so that source code (program text) is not restricted to those characters contained in the ASCII character set; allowing, for example, use of non-Latin-based scripts or extended punctuation.
- XML for graphical interface (XUL, XAML).
- Open source as a developmental philosophy for languages, including the GNU compiler collection and recent languages such as Python, Ruby, and Squeak.
- AOP or Aspect Oriented Programming allowing developers to code by places in code extended behaviors.
- Massively parallel languages for coding 2000 processor GPU graphics processing units and supercomputer arrays including OpenCL

[14] http://en.wikipedia.org/wiki/History_of_programming_languages

4.3 Cloud Computing

References to cloud computing in its modern sense can be found as early as 1996, with the earliest known mention to be found in a Compaq internal document[15]. The popularization of the term can be traced to 2006 when Amazon.com introduced the Elastic Compute Cloud[16].

1980's

To make more efficient use of costly mainframes, a practice evolved that allowed multiple users to share both the physical access to the computer from multiple terminals as well as the CPU time. This eliminated periods of inactivity on the mainframe and allowed for a greater return on the investment[17]. The practice of sharing CPU time on a mainframe became known in the industry as time-sharing. During the mid 70s, time-sharing was popularly known as RJE (Remote Job Entry); this nomenclature was mostly associated with large vendors such as IBM and DEC.

1990's

In the 1990s, telecommunications companies, who previously offered primarily dedicated point-to-point data circuits, began offering virtual private network (VPN) services with comparable quality of service, but at a lower cost. By switching traffic as they saw fit to balance server use, they could use overall network bandwidth more effectively. They began to use the cloud symbol to denote the demarcation point between what the provider was responsible for and what users were responsible for. Cloud computing extends this boundary to cover all servers as well as the network infrastructure.[18]

As computers became more prevalent, scientists and technologists explored ways to make large-scale computing power available to more users through time-sharing, experimenting with algorithms to provide the optimal use of the infrastructure, platform and applications which prioritized the CPU and efficiency for the end users[19].

[15] Antonio Regalado (31 October 2011). "Who Coined 'Cloud Computing'?". Technology Review (MIT). Retrieved 31 July 2013

[16] "Announcing Amazon Elastic Compute Cloud (Amazon EC2) - beta". Amazon.com. 2006-08-24. Retrieved 2014-05-31

[17] Strachey, Christopher (June 1959). "Time Sharing in Large Fast Computers". Proceedings of the International Conference on Information processing, UNESCO. paper B.2.19: 336–341

[18] "July, 1993 meeting report from the IP over ATM working group of the IETF". CH: Switch. Retrieved 2010-08-22

[19] Corbató, Fernando J. "An Experimental Time-Sharing System". SJCC Proceedings. MIT. Retrieved 3 July 2012

Since 2000

In early 2008, Eucalyptus became the first open-source, AWS API-compatible platform for deploying private clouds. In early 2008, OpenNebula, enhanced in the RESERVOIR European Commission-funded project, became the first open-source software for deploying private and hybrid clouds, and for the federation of clouds.[20] In the same year, efforts were focused on providing quality of service guarantees (as required by real-time interactive applications) to cloud-based infrastructures, in the framework of the IRMOS European Commission-funded project, resulting in a real-time cloud environment[21]. By mid-2008, Gartner saw an opportunity for cloud computing "to shape the relationship among consumers of IT services, those who use IT services and those who sell them"[22] and observed that "organizations are switching from company-owned hardware and software assets to per-use service-based models" so that the "projected shift to computing ... will result in dramatic growth in IT products in some areas and significant reductions in other areas."[23]

In July 2010, Rackspace Hosting and NASA jointly launched an open-source cloud-software initiative known as OpenStack. The OpenStack project intended to help organizations offer cloud-computing services running on standard hardware. The early code came from NASA's Nebula platform as well as from Rackspace's Cloud Files platform.

On March 1, 2011, IBM announced the IBM SmartCloud framework to support Smarter Planet.[24] Among the various components of the Smarter Computing foundation, cloud computing is a critical piece.

On June 7, 2012, Oracle announced the Oracle Cloud. While aspects of the Oracle Cloud are still in development, this cloud offering is posed to be the first to provide users with access to an integrated set of IT solutions, including the Applications (SaaS), Platform (PaaS), and Infrastructure (IaaS) layers[25].

[20] Rochwerger, B.; Breitgand, D.; Levy, E.; Galis, A.; Nagin, K.; Llorente, I. M.; Montero, R.; Wolfsthal, Y.; Elmroth, E.; Caceres, J.; Ben-Yehuda, M.; Emmerich, W.; Galan, F. "The Reservoir model and architecture for open federated cloud computing". IBM Journal of Research and Development

[21] Kyriazis, D; A Menychtas; G Kousiouris; K Oberle; T Voith; M Boniface; E Oliveros; T Cucinotta; S Berger (November 2010). "A Real-time Service Oriented Infrastructure". International Conference on Real-Time and Embedded Systems (RTES 2010) (Singapore)

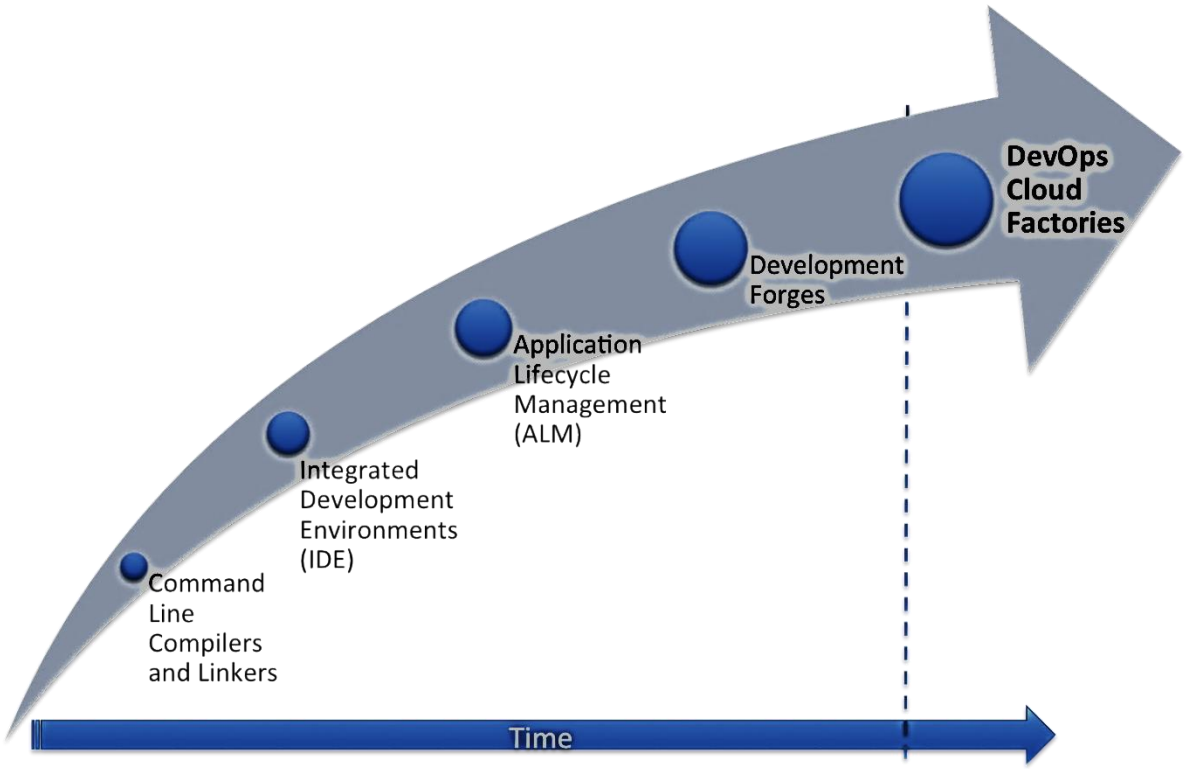
[22] Keep an eye on cloud computing, Amy Schurr, Network World, 2008-07-08, citing the Gartner report, "Cloud Computing Confusion Leads to Opportunity". Retrieved 2009-09-11

[23] Gartner (2008-08-18). "Gartner Says Worldwide IT Spending On Pace to Surpass Trillion in 2008"

[24] "Launch of IBM Smarter Computing". Retrieved 1 March 2011

[25] "Integrated IT Layers". Retrieved 4 March 2014.

Cloud Development Factor[26]



Continuing evolution of cloud application software

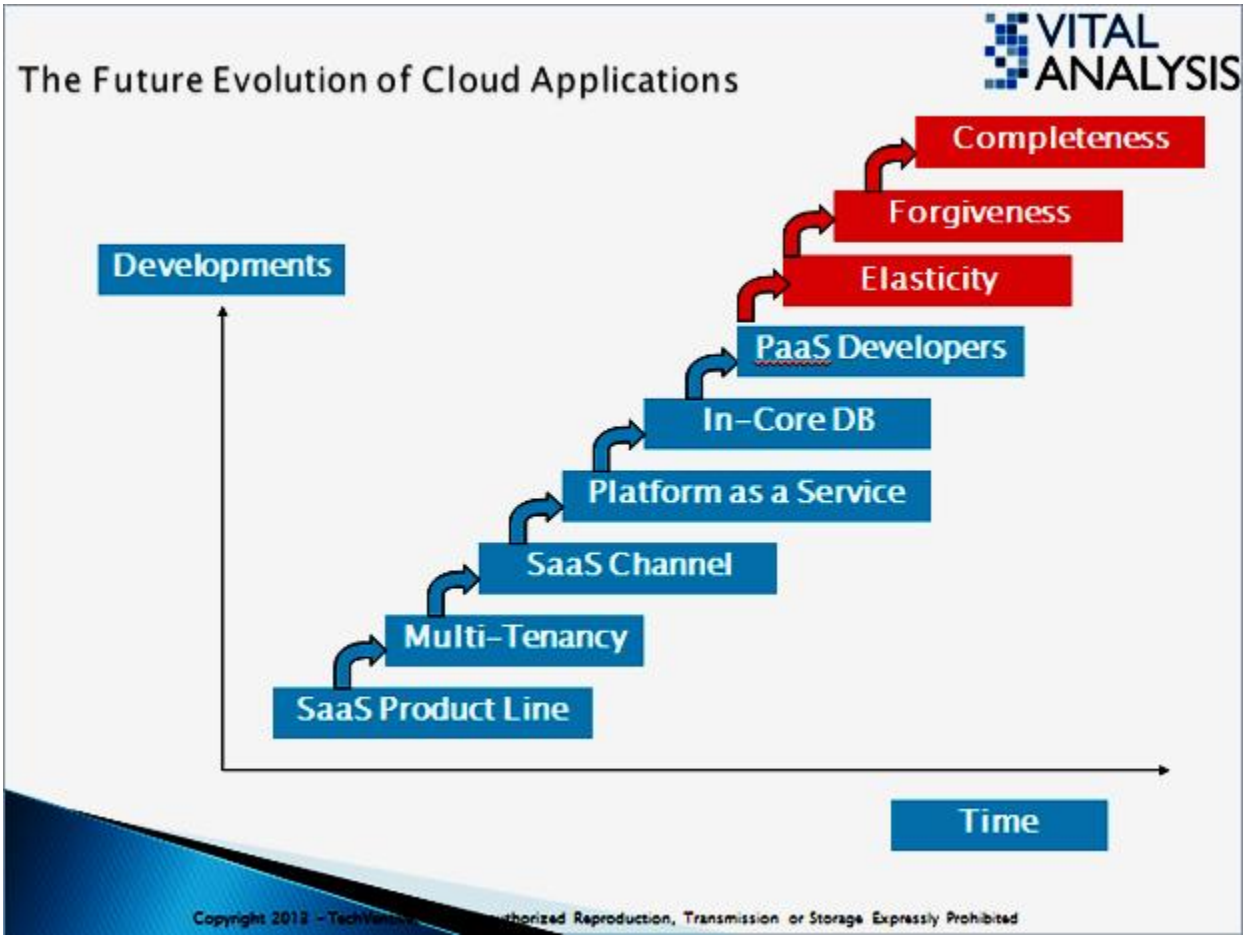
"Cloud computing" has dramatically changed how business applications are built and run. Delivering a new application is now as fast as opening your Internet browser. Platform as a Service — or PaaS — is a proven model for running applications without the hassle of maintaining the hardware and software infrastructure at your company. Enterprises of all sizes have adopted PaaS solutions like salesforce.com[27] for the simplicity, scalability, and reliability.

Just as Amazon.com, eBay, Google, iTunes, and YouTube made it possible to access new capabilities and new markets through a Web browser, PaaS offers a faster, more cost-effective model for application development and delivery.

PaaS provides all the infrastructure needed to develop and run applications over the Internet. Users can access custom apps built in the cloud, just like their SaaS apps, while IT departments and ISVs can focus on innovation instead of complex infrastructure. By leveraging PaaS, organizations can redirect a significant portion of their budgets from “keeping the lights on” to creating applications that provide real business value.

[26] server.dzone.com
[27] https://www.salesforce.com/paas/overview/

PaaS (platform as a service) become a reality a few years back. Salesforce.com introduced Force.com, its application software exchange and other tools to enable many other firms to develop applications on their cloud platform. NetSuite introduced its NS-BOS platform, too. Apple even has a vibrant ecosystem with a separate application store for commercial-ready software products. Platforms have given birth to all-new cloud application solutions like FinancialForce.com, Kenandy, Rootstock, etc. This evolutionary step to PaaS has been successful in some ways but needs more support/vibrancy to continue.[28]



By the end of 2012, the best solutions offered in-memory database technology, powerful analytics (also in the cloud) and a robust set of PaaS development tools. However, even the best solutions rarely offered all of the components mentioned to date in one solution[29].

[28] http://www.zdnet.com/continuing-evolution-of-cloud-application-software-7000017719/
[29] TechVentive, Inc

'Cloud computing' introduces more-granular and specific meanings of these terms: "workflows" and "processes" are used in different domains. At one level there are inter-company business processes, and at another level there are processes to get the wide area network ("the cloud") operational.[30] The main difference, then, between a workflow "automation" and an "orchestration" (in the context of cloud computing), is that workflows are processed and completed as processes[31] within a single domain for automation purposes, whereas orchestration includes a workflow and provides a directed action towards larger goals and objectives. *This software management layer will define the next era of computing.*

Orchestration allows IT teams and end users (assuming they are given appropriate permissions) to create virtual machines and application specific stacks of compute, networking and storage resources to get a particular job done. Orchestration software is what enables a true software defined datacenter and enables the time savings and cost savings [32]that companies are looking for. It does this by combining individual manual tasks performed by IT into a set of services that can be easily consumed by the end users.

Force.com versus Azure?

Both are multi-tenant cloud development platforms, oriented towards enterprise and ISV application development.

Force.com has more pre-built building blocks for typical business applications (UI elements, reporting, triggers, workflow, security model). If your app can use these building blocks, then Force.com allows for very rapid development and deployment, sometimes even by business analysts. Especially for custom apps for the enterprise-- their claim of "5x faster" is something we've seen play out in project development.

Using these building blocks [33]does bind your app to force.com (which is why some folks worry about lock-in), and requires your developers to learn new technology. Also, many force.com apps require "platform seats" which are priced per user, a pricing model that works very well for some apps but not for others. With the upcoming launch of Database.com and the

[30] "Cloud Computing". Retrieved 12 September 2012
[31] Thomas Erl. Service-Oriented Architecture: Concepts, Technology & Design. Prentice Hall, ISBN 0-13-185858-0
[32] http://www.dasher.com/blog/virtualization-to-orchestration/
[33] http://www.quora.com/What-are-the-advantages-and-disadvantages-of-Force-com-versus-Azure

acquisition of Heroku, Salesforce will address these issues by making it easier to develop in standard languages and supporting other pricing models.

Azure is a "lower level" PaaS, which offers fewer building blocks but more flexibility (direct access to the database, for example). This may make your developers more comfortable than they would be with force.com, especially if you're already a .NET shop. Pricing is also transaction based, which gives you more flexibility. But you shouldn't expect the same time time & productivity benefits that you can get with a higher level platform.

The evolution of portal tooling

To speed up the software system building process, a new concept of designing software is introduced in the '70s, called Computer Aided Software Engineering (CASE).[34]

The tools developed right now are evolutional products out of earlier tools. The first tools, developed in the mid '70s, were mainly introduced to automate the production and to maintenance structured diagrams. When this automation covers the complete life-cycle process, we speak of Integrated Computer Aided Software Engineering (I-CASE). When only one specific part of the life-cycle process is covered we speak of -just- Computer Aided Software Engineering (CASE).

Later on, integrated CASE (I-CASE) products were introduced. This was an important step because I-CASE products are capable of being used to generate entire applications from design specifications.

Of course, in the world of portal development, the big challenge development teams grapple with isn't so much which tool to use as much as it is which technologies to employ when drafting a solution. In years past, portlet development happened primarily on the server side, with JavaScript integration[35] or even the use of standard frameworks like Struts and JSF being more problematic than they were helpful. All in all, it's a massive change from the environment portal developers found themselves in four or five years ago. Not only have the tools progresses dramatically in both utility and usefulness, but the APIs have improved and the

[34] http://www.npd-solutions.com/case.html
[35] TheServerSide.com

portal backbone now makes integrating new technologies like JavaScript, JSF and even Spring based web frameworks all viable options. And with the ease of development greatly improved, more and more organizations will be looking at adopting portal based technologies in the future.

5. Mobile Applications

Mobile Application market is rapidly growing in global mobile market[36]. They are simple downloadable software program that runs on mobile devices and performs certain task for user of mobile phone. Apple has played a significant role in reenergizing the mobile apps space by bringing more consumers and developers into the ecosystem, there is significant activity outside the iPhone or smartphones space that is often not discussed.

5.1 Evolution of Mobile Applications

History of Mobile applications is traced back to the point when smart phones were launched. In2002 Blackberry launched its first Smart phone; at this point mobile applications market started growing. In 2006, Yahoo launched its Go Mobile Application Suite providing mobile user range of mobile applications. Major breakthrough in Mobile application development was in 2007, when Facebook launched its Development Platform and Apple came up with its Smart phone iPhone. With launch of Apple Appstore, in 2008 the application world started to blossom in earnest. First, it fundamentally changed the revenue model in favour of the developers which has become the current defacto standard (70/30) in the mobile apps business. Second, it brought more developers into the ecosystem as it fostered the notion of focusing on just 1-2 platforms rather than the entire device ecosystem to be relevant. It also reduced time to market for applications. In 2009 Blackberry and Nokia launched their application store. Currently we have the mobile application space across all platforms and on a global basis.

[36] http://www.slideshare.net/shwetaj/mobile-applications-7643425

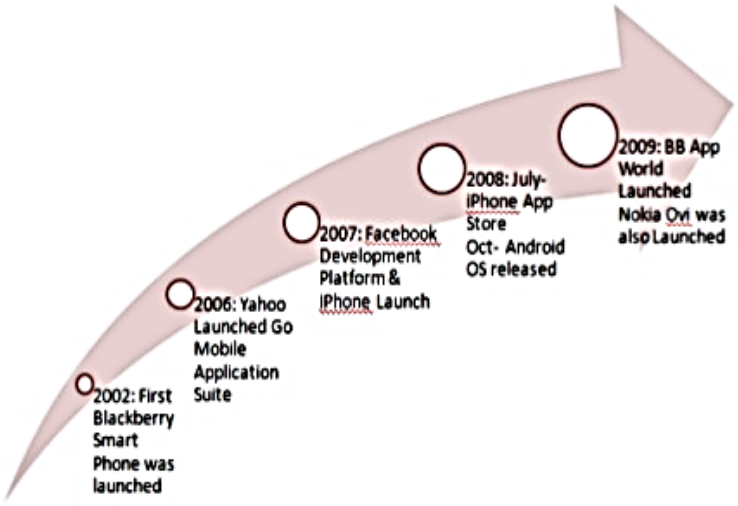


Fig: Evolution of Mobile Applications

Market Analysis: Facts and Predictions

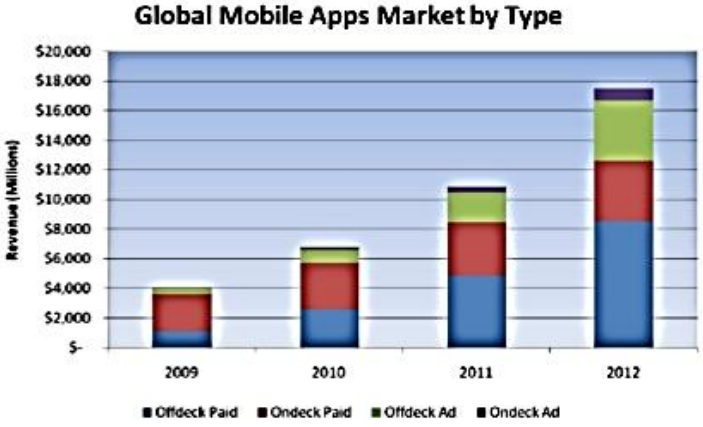
As per industry study done by Getjar the total number of apps downloads are expected to grow at 92% CAGR to almost 50 billion downloads per year by 2012. This is due to increasing number of feature phone users becoming active application users and increase in the number of apps downloads/user/month across the board.

*Source: Industry Study Commissioned by Getjar

The corresponding revenue figures for 2009 were over $4.1 billion growing 62% CAGR to $17.5 billion by 2012. It is observed that while Asia had the highest percentage of the download share, it has lowest percentage of revenue share. North America had the highest share of the apps revenue accounting for over 50% of the total revenue.

*Source: Industry Study Commissioned by Getjar

From the perspective of ondeck vs. offdeck and paid vs. Advertising, ondeck paid had the biggest share of the revenue followed by offdeck paid. In 2012, advertising based revenue would account for about 12% of the overall revenue and is expected to generate 28% of the app revenue.



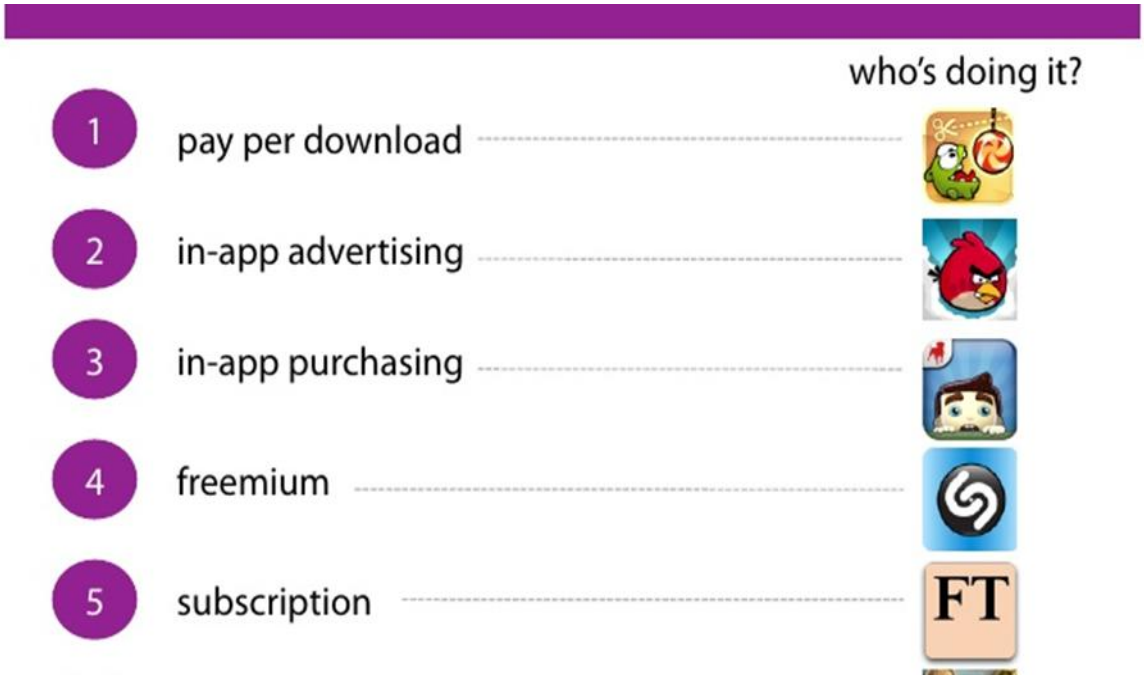
Top Selling Mobile Applications

Tetris is the best selling mobile app with over 100 million paid downloads.

Rank	Game	Share of Revenue
1	Tetris	7.00%
2	Bejeweled	4.00%
3	Guitar Hero III	3.60%
4	Wheel of Fortune	2.60%
5	PAC-MAN	2.50%
5	The Oregon Trail	1.90%
7	Ms. PAC-MAN	1.70%
8	Are You Smarter Than A 5th Grader?	1.60%
9	Tetris Mania	1.60%
9	Surviving High School	1.20%

5.2 Common Mobile Application Models

Revenue generation is of principal importance while developing apps. Most developers face a dilemma while deciding which source of revenue would be the best for their app.



There are multiple monetization options available for app developers today[37]:

- **Paid apps:** In this model, the app user pays upfront for the full app, usually at $0.99 or $1.99. It has been seen that price and revenue do not always correlate precisely and you should experiment with various price points. Expert suggest to start with the highest price point and put it on various levels of discount to test out sales.

This is the classic and proven monetization method[38], well-known both by users and publishers. Some apps priced below one dollar have had a staggering success.

- ✓ The model is very straightforward, the revenue is proportional to the downloads
- ✓ 70% for the developer / 30% for the platform
- ✓ Pricing can be different from one OS to the other
- ✓ Some users may be reluctant to buy apps, mainly on Android
- ✓ With paid apps, you are less likely to reach a lot of users

- **Freemium models / In-App Purchases:** The developer provides the basic features of the app for free and then requests a price to unlock the rest of the app functionality or provide enhanced features. In-App Purchases is a flexible model that can be used with free (freemium) or paid apps. Different types of in-app purchases exist. Users use in-app purchases to unlock/proceed to a new level, receive new features, use a new weapon in a game, receive an update and many others. Only quality, useful or engaged apps can be used with this model.
- **In-app Purchases:** In-app purchases drive a large chunk of the revenues for a lot of mobile apps these days. There are basically three types of in-apps monetization:

1. *Consumable Items* – the user pay real money for an item that can be used only once.
2. *Non-consumable Items* (like weapons, armors, etc) – the user pay real money for an item that can be used for an unlimited amount of times.
3. *Game Modes* (e.g., kids or hardcore)- the user pay real money for a new game mode.

- **Advertising:** Allowing banners and other forms of advertising on mobile apps is one of the most popular and simple way to monetize app these days. It allows app developers to acquire a lot of users without asking them to pay for it.

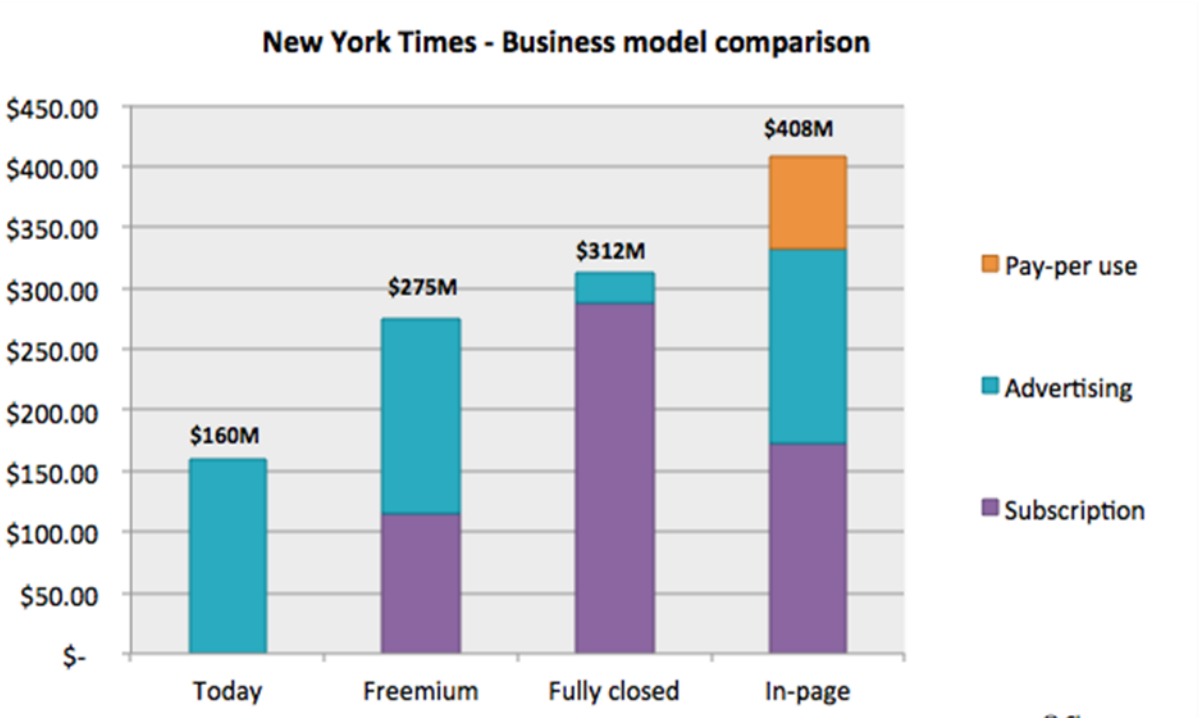
Mobile advertising has been touted as the next big thing since Apple's iPhone debuted in 2007 by Shira Ovide and Greg Bensinger in The Wall Street Journal. The research firm eMarketer Inc. has projected less than 2% of all U.S. marketing spending, or just $2.6 billion, to go toward mobile advertising in year 2012.
The different ad units for mobile are:

[37] http://adiquity.com/mobile-app-monetization-models
[38] http://applidium.com/en/news/an_overview_apps_monetization

1. Text Ads
2. Banner Ads
3. Rich Media & Video Ads

➢ **Subscription:**

1. Live Data Feed – Have the user pay a monthly fee in return for up to date data (Newspaper subscriptions, Magazine subscriptions, Stock Ticker Data)
2. Pay Per Data Usage – Have the user pay a recurring fee depending on how much they use (Examples: Dropbox)
3. Software As a Service – Pay for usage of the software as-you-go.



5.3 Web 2.0 Monetization Models[39]

a. affiliate network—A business (such as Commission Junction and Link Share) that connects web publishers with cost-per-action affiliate programs.

b. affiliate program—A deal offered by a company to share a portion of the revenues earned from traffic coming from web publisher websites. Affiliates provide text and image ads to post on the publishers' sites. If a user clicks through to the affiliate site and takes a specified action (e.g., makes a purchase, fills out a registration form, etc.) the publisher is paid a portion of the revenue or a flat fee. Companies offering affiliate programs include Amazon (the Amazon Associates program), Indeed, Click Bank, eBay and thousands more.

c. blog advertising—Advertising specifically designed for display on blog sites. Companies include Federated Media and Blogads.

d. contextual advertising—Advertising that is targeted to the content on a web page. Contextual ad programs include Google AdSense, Yahoo! Publisher Network, Vibrant Media, Kontera and Tribal Fusion.

e. cost-per-action (CPA)—Advertising that is billed to the advertiser per user action (e.g., purchasing a product or filling out a mortgage application). Companies include Amazon and Indeed. See also performance-based advertising.

f. cost-per-click (CPC)—Advertising that is billed by user click. The web publisher receives revenue each time a user clicks an ad on the publisher's site, regardless of whether the user makes a subsequent purchase.

g. cost-per-thousand impressions (CPM)—Advertising (usually banner advertising) that is billed per thousand impressions, regardless of whether the user clicks on the ad. Companies include DoubleClick, Value Click and many more.

h. e-commerce—Selling products and/or services directly through a website. Companies include Amazon, Dell,CafePress.com and thousands more.

i. interstitial ad—An ad that plays between page loads. Companies include Tribal Fusion, DoubleClick, and many more.

j. premium content—Content on a website that is available for an extra fee (e.g., e-books, articles, etc.).

k. RSS ad—An ad included in RSS feeds. Companies include Feedster, Feed burner and Yahoo! Search Marketing.

[39] http://www.deitel.com/Books/Web2eBook/Web20MonetizationModels/tabid/2497/Default.aspx

5.4 Mobile Application Market Sizing

There are approximately 100 key application stores. We can divide market into four parts:-

- Device Manufacturer:- Samsung, Apple, Android, iPhone
- Os developer:- Google, Symbian, Windows, Mac
- Mobile Operator
- Independent(third party/off deck)

App Stores

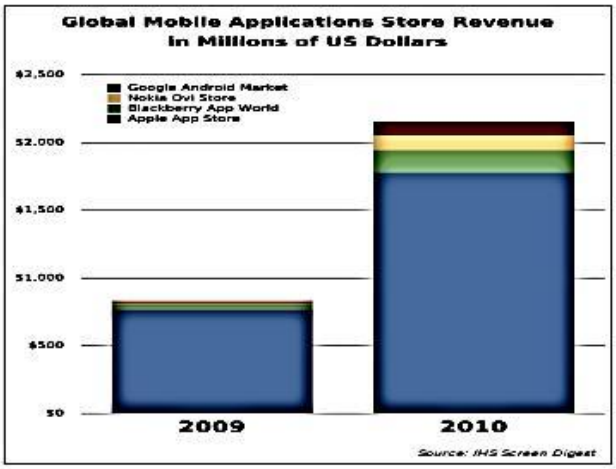
Revenue Share

Top 4 App Stores are Apple , Blackberry App worls, Nokia Ovi Store and Google Android. Apple App store has major market share of approx, 83%. Currently market is dominated by Apple, but it is predicted that in coming years Google Andriod will give Apple strong competition

Global Mobile Applications Store Ranking in 2010 and 2009
(Ranking by Revenue in Millions of U.S. Dollars)

2010 Rank	Store	2009 Revenue	2009 Share	2010 Revenue	2010 Share	Year-Over-Year Growth
1	Apple App Store	$769	92.8%	$1,782	82.7%	131.9%
2	BlackBerry App World	$36	4.3%	$165	7.7%	360.3%
3	Noka Ovi Store	$13	1.5%	$105	4.9%	719.4%
4	Google Android Market	$11	1.3%	$102	4.7%	861.5%
	Total	$828	100.0%	$2,155	100.0%	160.2%

Source: IHS Screen Digest February 2011



- Gartner predicts that in 2013, 21.6 billion apps will be downloaded, generating nearly $30 billion in revenue
- The growth between 2010 and 2014 is forecast be over 1,000 percent.
- By 2013, according to Gartner's numbers, the average revenue per app will be $1.36.
- By the end of 2014, advertising will be generating a little under a third of the revenue generated by application stores, up from 16 percent in 2010

Top 4 App Store

Brief overview of top four App stores and their business model

Apple App Store

- For **app download on Apple's app capable Devices** (the iPhone, iPod Touch, iPad and Mac) offered by Apple Inc. in 2008
- Allows users to browse and download applications from the iTunes Store
- **Jan 7, 2011** - Mac App Store Downloads **Top One Million in First Day**
- Jan 22, 2011 - **Apple's App Store Downloads Top 10 Billion**
- Feb 15, 2011 - **Apple Launches Subscriptions on the App Store**
- **App Store revenue estimated to reach $2 billion in 2011**

Blackberry App world

- An application distribution service and application by Research In Motion (RIM) in 2009 for a majority of BlackBerry devices
- Provides BlackBerry users with an environment to browse, download and update third-party applications
- Applications are both free and paid from $0.99-$999.99 in the U.S. developers pay a $200 USD fee (Per every 10 Apps) to participate in the program.
- **150,000+** apps available for download
- RIM offers Blackberry programmers 80%
- On Dec 3, 2010, RIM announced that **daily downloads were 2 million apps per day**

Android Market

- An online software store developed by Google for Andriod Devices
- Amazon Opens Android App Store to Developers
- **98.9%** of all applications downloaded are free apps
- **300,000+** apps available for download
- **8 million Android users downloaded 289 million apps**
- Developers of software (apps) receive 70% of the application price, with the remaining 30% distributed among carriers and payment processors

Nokia Ovi Store

- The Ovi Store was launched world wide in May 2009 by Nokia
- customers can download mobile games, applications, videos, images, and ringing tones to their Nokia devices
- Applications are both free and paid
- **350,000+** apps available for download
- Downloads reached 4 million in January 2011

Business Model

Below is the description of business model followed by top App stores

Name	Available Apps	Installed Base	Available Countries	Developer Submission Fee	Developer Revenue	Payment Models For Users	Payment Models for Developers
Apple App Store	300.000 (Nov 2010)	125 million (Nov 2010)	90 free / 90 paid	US$99/year	% 70	-Itunes	Developers paid via direct deposit into bank account
Android Market	170.000 (Nov 2010)	13 million (March 2010)	48 free / 36 paid	$25	% 70	- Google checkout - Operator billing	-Google checkout
BB App World	14.486 (Oct 2010)	46 million (July 2010)	67 free / 13 paid	US$200/10 apps	% 70	- Credit card - Operator billing - Paypal	-PayPal
Ovi Store	28.000 (Oct 2010)	250 million (Jul 2009)	224 free / 232 paid	1 €	% 70	- Credit card - Operator billing	-Credit Card

*Source: http://bestmobilephonesoftware.com

Third-Party Platforms

Third party platforms for downloading mobile applications are becoming quite popular. Below is the comparative study of various third party platforms and their business model.

Name	Available Apps	Installed Base	Device platform	Available Countries	Developer Submission Fee	Developer's Revenue
Getjar	68.625 (May 2010)	Unknown	Multiple Android, BlackBerry OS, Flash Lite, iOS, Java, Palm OS,Symbian, Windows Mobile	More than 200	Free	% 100 except Apple based apps
PocketGear	140.000 (June 2010	Unknown	Multiple Android, BlackBerry OS, Flash Lite, iOS, Java, Palm OS,Symbian, Windows Mobile	More than 175	Free	42%
VZAppZone	Unknown	86.6 million (April 2009)	Multiple BlackBerry OS, BREW,Windows Mobile	Unknown	US$750/app per update; 1st platform included, then US$150/platform	Varies

*Source: http://bestmobilephonesoftware.com

5.5 Mobile Industry History

Since the mid 2000's, the industry[40]—especially in terms of software, apps, and web interactivity—design has absolutely boomed and gone through immense advancements in every way imaginable way. This is because during the last decade (and even more so during the last five years), mobile phones have gone from being simple phones to being fully-functional pocket-sized computers, with a mobile equivalent for almost every single feature of a full-blown desktop or laptop computer.

This general trend has naturally led to the development of an enormous industry that can basically be divided into two interrelated parts: mobile app development and mobile-friendly design. Mobile development has grown alongside thousands and eventually millions of increasingly-sophisticated mobile applications developed for ever-growing smartphone marketplaces. Modern mobile applications take advantage of built-in hardware abilities in astoundingly clever ways and perform robust web-based and organizational functions.

With the rising advancement of high-speed, large-bandwidth mobile networks like 3G, WiFi, and 4G, consumers can access the web on their smartphones as quickly and reliably as they can via a normal computer. At the same time, the growing access has prompted an explosion in the number of mobile-friendly or purely mobile websites; this in turn has increased demand for mobile design enormously, and that's where developers like you come in.

Today, the mobile phone market is outstripping ownership and use of landline phones in virtually every country on Earth, the steady conversion of more and more of these phones into smarter, more multimedia-friendly devices that work off WiFi, 3G, or 4G networks suggests that the future of mobile development and design will only grow stronger.

Let's go over some of the major ingredients that led to the rise of modern mobile design.

The First Cellular Networks

Starting from simple wireless analog-based (1G) portable phones, it wasn't until the late 1990's that cell phones turned into more sophisticated devices as the technology inside them started to spawn an ever larger number of features (features that nonetheless seem basic by today's standards). These first phones gave basic calling abilities to users and their convenient portable capacities are what established them as widely-used communications devices.

2G/GSM Networks and the Rise of Mobile Media

It wasn't until the mid to late 1990's that a new communications network, known as GSM—or 2G, as it was less often called—began to develop in which more mobile services could be offered. At this point, the first pre-smartphone devices began to appear, and the fact that data transmission over these next-generation devices was digital instead of analog allowed them to

[40] http://www.sitepoint.com/the-advancements-in-mobile-design-and-how-it-has-developed-into-a-strong-industry/

carry many of the more basic smartphone features that we use as the basis for modern app development. Capabilities such as text messaging, downloadable content, and extremely basic web access gave consumers the ability to send emails, view a small selection of online multimedia, and download simple digital applications such as ringtones and music files.

The growth of the GSM, or 2G networks, is what really expanded mobile phone use so broadly that mobile devices eventually eclipsed landline communication tools. Despite the 2G networks explosive popularity amongst users of all income levels worldwide, these machines were still pretty basic compared to today's mobile devices.

3G Arrives

It wasn't until the early 2000's, with the development and service offering of the first 3G wireless digital networks that true smartphones arrived. In 2002 and 2003, network operators began to offer widespread 3G access based on more powerful wireless transmission technology that depended on efficient packet switching data transmission found in computer-based web connections instead of the 2G networks circuit switching mechanism.

With the arrival and rise of 3G, the modern era of wireless mobile smartphones as pocket-sized computers truly began, especially after 2005, when High-Speed Downlink Packet Access (HSDPA) was implemented into 3G and expanded its data carrying ability even more. The resulting explosion in online media accessibility created a tandem explosion in online media creation. At the same time, in order to take full advantage of all these web-based data options that phone networks now offered, mobile applications started appearing for smart phones—at first in small quantities but later at a rapid development pace. Also, the devices themselves had to be redesigned so that they could better display digital media and other interactive systems, taking full advantage of the growing apps market.

4G Replacing 3G

Currently, even 3G itself is being slowly replaced by the much more powerful, purely packet-switching-based, data optimized 4G network. With this new technology, ten-fold increases over 3G in data transmission ability are coming into the picture, making access to digital media even more robust and further bolstering the demand for media-rich mobile applications.

Bye Bye Buttons

This is where the touchscreen phone with its large visual screen interface comes into the picture. Today, this is the replacement to the antiquated button control and small display screen based phones of several years ago.

The end result of this mix is the rapid replacement of old phones for new touchscreen devices and the deliberate obsolescence of older networks in favor of 3G and its even more powerful successor, 4G. In many countries, anything older than 3G is no longer even available and almost all new phones being sold feature a predominantly buttonless touchscreen design.

As of the most recent figures, there were over 1.6 billion 3G/4G mobile subscribers worldwide (up from only 297 million in 2007)

Mobile Operating Systems and App Marketplaces

Finally, we come to the growth of mobile device operating systems. Since modern smartphones are more like computers than cell phones in a classical sense, they naturally needed a fully-functional OS of their own. Because of this, several companies such as Research in Motion, Apple, Google, and Microsoft all came out with their own competing mobile operating systems that gave a full-scale interface to digital media access and software applications compatibility

Thanks to all these features, the mobile apps development landscape has exploded like few other industries ever have in history. Since 2010-2011, app marketplaces have grown for mobile operating system developers such as Apple with its iOS platform, Google's Android OS, and Microsoft's Windows Phone.

As of 2012, the Apple mobile apps market alone houses over 600,000 smartphone apps and has had over 30 billion downloads to date. Similarly, the Android OS market gives access to hundreds of thousands of additional applications and sees some 3 million downloads per day from its online platform.

5.6 Application Usage & Forecast

The following is a table showing mobile application usage by software segment [41]-

Mobile Application Usage by Software Segment and Region (Percentage of Respondents)

	Total	North America	Latin America	Western Europe	Eastern Europe	Asia/Pacific
Email and calendaring	46	51	44	52	40	43
IM	37	38	39	30	34	40
Office and personal productivity	26	28	28	24	12	28
Web conferencing	25	27	28	21	17	26
E-commerce	20	14	25	17	20	23
Social software suites	20	20	19	15	12	23
CRM	19	17	22	12	20	23
Collaboration	18	20	19	15	12	23
Industry-specific	18	16	23	18	17	16
ERP	17	15	17	9	10	23
Enterprise search	16	17	18	9	15	17
SCM	16	16	12	9	19	19
PPM	15	15	19	12	6	15
BI	15	13	17	9	11	18
ECM	14	16	10	10	16	17
MDM	14	16	10	10	16	17
DCC	14	16	15	8	16	13
None of the above	9	11	6	12	10	7

BI = business intelligence
DCC = digital content creation
ECM = enterprise content management
MDM = master data management
PPM = project and portfolio management
SCM = supply chain management
Notes: Number of respondents equals 1,443.
Question we asked: "Which, if any, of these applications are mobile-device-enabled today?"

[41] http://www.forbes.com/sites/louiscolumbus/2013/06/09/roundup-of-mobile-apps-app-store-forecasts-2013

Mobile App Store Forecasts

- Gartner predicts that by 2016, there will be 310B downloads with an estimated value of $74B in revenue from app stores . Gartner has also predicted that by 2017, 25% of all enterprises will have an app store. This includes both new application purchases and recurring revenues form subscription pricing models.
- 90% of global mobile app store downloads in 2013 are forecast to be free, increasing to 93% in 2017. 73.2B free downloads will occur in 2013, increasing to 287.9B by 2017. Paid-for downloads will increase from 8.1B in 2013 to 21.6B in 2017.
- In-app purchase will drive 41% of the store revenue in 2016. While the market is moving toward free and low-priced apps, in-app purchase will increase in both the number of downloads and in the contribution to the store revenue. As a result, we see a shift in user spending from upfront purchases to in-app purchases.
- 99% of the paid-for app store downloads cost less than $3 each. Similar to free apps, lower-priced apps will drive the majority of the downloads. We estimate that apps between $0.99 and $2.99 will account for 87.5% of the paid-for downloads in 2012, up from 86.8% in 2011. That percentage will further increase to 96% by 2016.
- Global mobile app store revenue is projected to reach $24.5B in 2013, increasing to $74B in 2017. Paid-in downloads (69%); in-app purchase (17.3%) and advertising (13.7%) are the three revenue sources in 2013. In 2017, revenue shifts significantly to paid-for downloads contributing 45.2% of revenue, in-app purchases, 40.9% and advertising, 13.9%.

5.7 Mobile Worker Population

By 2015, the world's mobile[42] worker population will reach 1.3 billion, representing 37.2% of the total workforce. The Americas region, which includes the United States, Canada, and Latin America, will see the number of mobile workers grow from 182.5 million in 2010 to 212.1 million in 2015.

5.8 Conclusion

“With consumers becoming more accustomed to the extensive availability of mobile content, the mobile content market faces the challenge of matching consumer expectations,” said Harry Wang at Parks Associates. “Success in the mobile content market will ultimately reflect a service provider’s ability to deliver content to a wide range of platforms and integrate a program interface that will improve the experience of the consumer.”

[42] http://www.businesswire.com/news/home/20120105005455/en/Mobile-Worker-Population-Reach-1.3-Billion-2015#.VBEt28KSyn9

6. Key Success Factors

Two major trends are driving the review and overhaul of technology companies' existing economic models.

The first trend[43] is increased competitiveness for total available markets (TAMs). Previously, technology companies had clear, delineated lines of what kinds of products and services they sold and what markets were served. There was a distinct server market, storage market, enterprise software market and so forth. Today, companies offer a wider variety of products and services that fall outside of their previous scope.

The second trend occurs outside of the technology sector. Non-technology companies are expediting efforts to digitize their offerings and are changing the way they buy from technology companies.

Companies are now relying on software and online services for more than just marketing. Instead, software is now being embedded into products as a valuable capability and as a competitive differentiator. As technology companies move from being 'pure play' operations, their entire development, sales and distribution models will be upended, according to PwC's findings.

The blurring of TAMs and the shift in the way companies buy technology has created a tipping point for the technology sector.

LEADING RISKS OF THE SOFTWARE INDUSTRY

Given the current state of the software industry, the following key risks emerge:

1. Competition from SaaS Competitors

The movement to on-demand software is a top risk for traditional software developers. Arising out of the shift to online (cloud) applications and data storage, SaaS companies are taking on many tasks formerly fulfilled by off-the-shelf software. What remains to be seen is how quickly they will penetrate the market. At this time, SaaS apps have heavily penetrated the customer relationship management (CRM) and human capital management (HCM) functions and less so the order fulfillment, supply chain, manufacturing and core financial functions. They also have higher levels of penetration in smaller and medium-sized organizations or smaller divisions or distributed locations of larger organizations.
In the future, SaaS developers will clearly target more industries and improve their applications to handle more complex functions; this will push the leading traditional developers to compete or lose their own market dominance. While SaaS apps are not yet a silver bullet, companies and people are eagerly adopting them because they are perceived to be less complicated, more intuitive and built on platforms that people use every day, such as iPads and smart phones.

[43] http://www.siliconrepublic.com/enterprise/item/36094-software-industry-shifts-fo

2. Product Obsolescence

In today's world, every industry is subject to technological innovation and radical disruption. New hardware devices, platforms and software approaches can result in profitable opportunities for nimble and savvy software companies. However, they can also bring on rapid product obsolescence for a software company that lets its prior successes blind it to change. To mitigate this risk, companies need to monitor trends and anticipate where the market is going. They need to invest in significant R&D efforts to enhance their own technologies and better understand their customers' needs. Given the length of the development lifecycle for cutting-edge technology, however, this is often not protection enough. Ariba, a leading provider of sourcing solutions, is an example of a company that anticipated the SaaS movement and now offers only SaaS solutions.

3. Privacy and Security Risks

The software industry has always been at risk for security and privacy breaches. Despite their best efforts, developers have been unable to prevent hackers from penetrating their code to gain unauthorized access to data that their applications maintain for users. In today's SaaS environment, protecting against privacy and security breaches becomes even more complicated. Even with assurances of security protection, SaaS apps are highly vulnerable to intrusion and the theft of personal or financial records of a company's employees and customers.

In an effort to regulate this threat, the U.S. and most governments throughout Europe have enacted strict rules about the privacy rights of individuals as well as penalties for breaches of confidential personal information, identity theft and financial data such as credit card numbers. To mitigate this risk, SaaS and other software developers must continuously develop stronger methods of security to stay one step ahead of those who would seek to gain unwarranted access to applications.

4. Intellectual Property (IP) Protection

The IP value of an innovative application has risen sharply in a world that is now more global, competitive and prone to piracy. Not only do software developers need to protect the investment they made in engineering, programming and design costs of their applications, but they also need to be sure they can maximize the financial value from licensing any inventions, patents and service marks affiliated with them. As a result, developers must take many steps to ensure the value they put into a project is not lost to illegal sharing, product imitation, patent infringement or counterfeit production and distribution of their software.

5. Pricing Models

As little as two decades ago, no one would imagine that pricing models would be a risk in the software industry. But today, pricing models are increasingly complex as companies transform themselves from being just developers of software to providers of solutions. One result is that many developers are moving from a license model to a subscription model. Another is that we see more companies bundling software, services and maintenance into a package for customers, or entirely customizing a solution for each client. These trends complicate the rules for revenue recognition, forcing the adoption of new accounting standards and, in some cases, damaging the financial reputation of companies that have stretched GAAP rules too far.

Furthermore, the economic downturn has created significant foreign currency fluctuations, especially between the dollar and euro, that have impacted the stability of pricing models. This has demonstrated that the software industry is sensitive to global economic conditions, a risk that will need further analysis and new solutions for companies that are increasingly selling software products and services on a global basis.

A BULLISH FUTURE

The software industry appears to be climbing a wave of growth and profitability. Trends indicate that it will continue to be an industry in high demand as new technologies, platforms and innovative solutions are created. For every function or task for which software has already been applied, there is the never-ending pursuit of more “elegant” solutions that work faster, smarter and better. Meanwhile, delivery models and new generations of smart phones and tablet computers are rapidly improving the speed at which content is delivered and the quality it is displayed in. Although there is still a need for traditional off-the-shelf software, improvements like these will continue to drive the growth of SaaS solutions, mobile apps, the video gaming industry, 3-D entertainment and as-of-yet undiscovered software applications.

If there is anything clear about the future, it is that the leading risks in this industry – especially security and privacy, competition from SaaS apps, and product obsolescence – will increasingly challenge software companies to manage and mitigate. These risks will require extensive resources invested in continuous R&D efforts and keen management oversight.