

Telecoms Infotech Forum

Briefing paper

The Economics of Software Development

June 2004

www.trp.hku.hk/tif

Telecoms InfoTechnology Forum

TIF is an industrial and policy forum run by the Telecommunications Research Project (TRP) at the University of Hong Kong, director Dr John Ure. The TRP provides background briefing papers for each TIF and posts these, together with presentations and proceedings papers, on the website <http://www.trp.hku.hk/tif>. TIF is the source of funding of the TRP and relies upon sponsorship.

The output of the TRP is public domain research into economic, policy and regulatory aspects of telecommunications and related sectors such as IT, new media, Internet and e-commerce.

This TIF conference is supported by:

- The Office of the Telecommunications Authority in Hong Kong (OFTA)
- Cable & Satellite Broadcasting Association of Asia (CASBAA)
- Hong Kong General Chamber of Commerce (HKGCC)
- Hong Kong Information Technology Federation (HKITF)
- Hong Kong Internet Service Providers Association (HKISPA)
- Hong Kong Java Users Group (HKJUG)
- Hong Kong Telecommunications Users Group (HKTUG)
- Hong Kong Wireless Technology Industry Association (WTIA)
- Internet and Innovation Asia (IandI)
- Internet and Telecom Association of Hong Kong (ITAHK)
- Internet Professionals Association (iProA)
- Information and Software Industry Association (ISIA)
- World Teleport Association (WTA)

The objective of TIF is to stimulate informed interest in the policy and regulatory aspects of information and communications technologies (ICTs), to foster greater transparency and a better understanding of the economic and technological dynamics of the sector, its impact on social welfare and its policy implications.

For further details of TIF and TIF membership, please contact:

Jenny Wan at the *Telecommunications Research Project*: tel: 2859-1919; fax: 2857-9434
or Email: Trproj@hkucc.hku.hk

The Economics of Software Development

The Market for Software

Despite Moore's 'Law' that tells us the cost of electronic circuits falls 50 per cent every eighteen months, the total cost of ownership (TCO) of IT systems remains a concern to many IT users. Costs involve licence fees and the effects of network economics which pressure users into continually having to adapt and respond to requirements placed by changing business needs or by the emergence of new and disruptive technologies like the Internet. However, a major component in the TCO is technical staff training costs. According to a published IDC study, only about 10 per cent of the TCO in common enterprise systems are usually accounted for by the initial purchase price, the rest are mostly associated with the labor costs for computer technicians who install, maintain and upgrade the systems, and the cost of training and downtime.¹

Following on the heels of the Asian economic crisis, the collapse of the dot.com bubble triggered a telecoms, IT and stock market plunge. Companies have become wary of claims that IT can improve business prospects. Certainly most of the e-business markets collapsed as liquidity disappeared, and e-commerce has been slow to take-off, leaving electronic messaging and conferencing among the prime applications. On the other hand, IT as a backend tool for enterprise resource management is only indirectly influenced by market conditions. But in all the gloom and doom, managements have become more judgmental about the benefits. Y2K spending boom also deterred additional investments. And that boils down the issue for most enterprises to when is IT spending a cost and when is it an investment? For most SMEs in Hong Kong is it generally seen as a cost to be avoided unless the benefits are overwhelming, or unless the costs are minimal as in the case of cellphones, PDAs, and fax machines. According to the latest Government figures, only 50 per cent of Hong Kong SMEs use computers and only 43 per cent connect to the Internet.

The markets are clearly reviving slowly as new systems become legacy systems and as confidence in economic growth returns, albeit hesitantly. But the world has changed quite radically from the 1990s. The rise of Linux in the server market is now well established, and it begins to seep into consumers markets, on computers of SOHOs and even cellphone operating systems. Companies such as IBM, HP and Novell have agreed to market Linux. On the other hand, Microsoft has shifted strategy somewhat, first by placing major resources into the home consumer and electronics market, including cellphones, and second by responding to the open source market trend by opening its source code to a wider range of developers. Under the Microsoft Shared Source Initiative, code access for Windows 2000, Windows XP, Windows Server 2003, Windows CE 3.0, Windows CE .NET and components of Visual Studio .NET are available to developers in many communities such as governments, academics, industry partners and competitors. Cyber Security in all its dimensions (commercial, personal and political) has become a major concern to the IT industry as well as to corporate customers.

¹ IDC (2002) *Windows 2000 Versus Linux in Enterprise Computing: An Assessment of Business Value for Selected Workloads*. An IDC White Paper Sponsored by Microsoft Corporation. The study reported that in four out of five applications surveyed by IDC, Windows Server 2000 was 11-22% cheaper over a five-year period than a Linux solution due to better software tools for fixing problems that saved staff time.

These changes, and many others too numerous to examine here, are opening up new dimensions for software developers. These dimensions consist of cheaper and easier to use development tools, greater access to source codes for software innovation, greater opportunities and demand for customization, a growing demand for inter-operability between standards through the harmonization of standards but not through the creation of single standards, and an associated demand for off-the-shelf standard software packages for easy deployment. In parallel, there is a greater demand for IT knowledge training as opposed to skills training, where the former gives senior staff and managers grasp of the fundamental functionality of IT systems, of the problems and issues involved (technical, organizational and human) as well as the costs and benefits. This is an educational job to be done, and part of that is to understand the economics of software development.

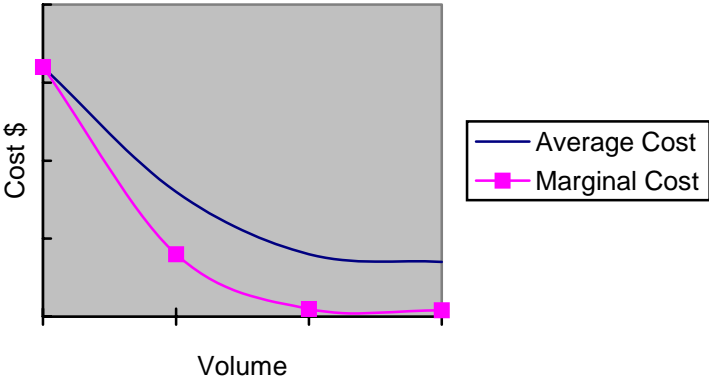
Economics of Software Development

Software takes enormous resources in time and effort to create and design, to write the code, run tests, etc. But once created its marginal cost is minimal. It can be replicated endlessly and distributed globally within seconds. Furthermore, because software output can be input into writing other software programmes, the ability of people to write new software programmes is growing every day and the cost of writing them is falling. Software that writes software makes the process even simpler, faster and cheaper.

This is counter-balanced by a shifting outwards of the technological frontier. Rapid advancements in electronics are continually opening new avenues for new products, for example wireless devices, imaging products, calibration algorithms for testing and measuring equipment. Due to the first trend software rapidly becomes a commodity. Due to the second, software development continues to demand enormous resources for research and development. The result is an IT industry under pressure from two directions.

In economic theory textbooks, a perfectly competitive market will generate marginal cost pricing, but given the high average cost of software production and the near zero marginal cost of software distribution, marginal cost pricing is commercially unsustainable. See Figure 1.

Figure 1.



'Free' software is non-commercial by its nature. Software that is distributed for free or at little cost can be commercially sustainable only if revenue comes from other sources. In this context there is an important meaning to the term 'sustainable'. If 'sustainable' only means cost recovery, then 'free' (to use) software can be sustainable. For example, 'free to use' but it may cost a small amount to distribute (see below) or 'free download' from a website that carries advertisements that bring in revenue, or 'free' distribution of software that has been funded by a research grant, and so on.

If 'sustainable' means able to fund continuing product research and development then 'free' can only be commercially sustainable under certain 'models'. One possibility is the services model. If 'free' means free as in 'free as a puppy' (see below) then it implies a related and ongoing revenue source, such as a services agreement covering installation, customization, trouble-shooting and future enhancement/upgrade. This is the Open Source model that uses a variant of the General Public Licence (GPL) that guarantees access to the source code. (See below.) The GPL licence requires that any subsequent development of the software must itself be distributed under a GPL-type conditional licence, which is to say the source code must be made available in some form or other. This has been called the 'viral effect' because it carries over to each subsequent generation of software development. Given the classic marginal pricing problem mentioned earlier, the GPL has the effect of freezing prices at this level, thereby excluding what economists call the 'Second Best' pricing solution, namely 'marginal cost plus markup' where the mark-up tries to recoup some of the average costs involved.^{2,3}

A second model is the subsidy model. In theory subsidies can come from anywhere, but the principal example is government subsidy of commercial software development. For example, in the 2004 version of *Digital 21* the Hong Kong Government sets out its policy of software promotion.

To widen product choice and maximize the potential for cost savings, we will promote the use of open source software (OSS) technologies and solutions within the Government through showcases and trials organized by the ITSD's IT Solution Centre. We will also promote OSS development and adoption in the private sector through the provision of funding support from the SME Development Fund and the ITF for projects that develop OSS or assist SMEs in using OSS. We are conducting a survey of OSS adoption in the business sector to identify barriers to OSS adoption and measures to promote wider adoption. (p. 31)

The issue here is where to draw the line between public sponsorship of a software development industry, and public subsidies that may favour or determine one direction for the

² Mark-ups are typical of utility industries such as water and power which face the same marginal cost pricing problem. There are various ways to determine the optimum mark-up scheme, for example Ramsey pricing will differentiate mark-ups between customers with low price elasticity of demand (high mark-ups) and high price elasticity of demand (low mark-up). Uniform mark-ups tend to be the acceptable political option.

³ For a discussion, see Stefan Kooths, et al. (December 2003) *Open-Source Software: An Economic Assessment*, Muenster Institute for Computational Economics, MICE Economic Research Studies, v.4 at: http://mice.uni-muenster.de/mers/mers4-OpenSource_en.pdf

industry over another.⁴ Can and should subsidies be market neutral? Are subsidies to commercial companies as opposed to public sector research institutions justified at all? Will this policy subject government to the risks of ‘backing winners’ when government is not well placed to forecast market trends, technological development, etc?

The third model is the commercial model. Vendors of commercial software argue that license fees generated from sale of their intellectual property (IP) can in turn be fed into their next research and development (R&D) cycle. This will ensure that innovation be sustained, customer requirements be met, jobs and thus tax revenue be generated for the benefits and prosperity of the whole society. User software licence fees are the bedrock of the commercial model. On the other hand, open source communities argue that by distributing technology freely and openly in an open source model will allow technology to reach many, thus creating return for society through productivity gains at the macro level.

There is a fourth route vendors can take, although it is not exactly a model yet, more like a pricing strategy at this point, which is to embed software into hardware. This is done for example by mobile phone handset manufacturers, using various operating systems. Mobile cellular operators and content and application developers are less than happy about this because it restricts the range of software applications that can run on any one handsets. Developers then have to rewrite the programmes to conform to different operating systems.⁵

Linux and MS-DOS – a Contrast in Philosophies

There is clearly a whole spectrum of industrial strategies available to companies, and in the software arena they are obviously best represented by the Linux operating system and Microsoft’s MS-DOS. The contrast in approaches to software development could not be starker, and goes beyond mere commercial considerations as it involves the personalities of the founders, and their philosophies of life.

Even the titles of their books reveal a lot. Linus Thorvald’s account of how he came to develop Linux appears in his 2001 book (with David Diamond) *Just for Fun*. From a modest background in Finland, he was a schoolboy geek who loved playing with computers. He set upon the task of improving the Unix operating system (hence the name Linux) and made use of the Internet to marshal resources by inviting software engineers and programmers worldwide to share his efforts. This approach was a highly cost-effective strategy for someone lacking resources and backing from any large IT company. His one important control mechanism was to have the final say on which suggestions to adopt for Linux. The next important step was to generalize this approach by adopting the General Public Licence (GPL) principles of the Free Software movement pioneered by Richard Stallman (see below). The subtitle of *Just for Fun* is ‘The Story of an Accidental Revolutionary’.

⁴ For example, Japan and Korea subsidized the development of CDMA wireless technologies, while Europe favoured the GSM technology. In telecommunications, as a user and importer of technology, Hong Kong has always maintained a technology-neutral stance. Should the same apply to IT?

⁵ For an operator’s viewpoint, see the points raised by Bruce Hicks, Group MD for Sunday, reporting back on the GSM Association discussions <http://www.trp.hku.hk/tif/papers/2004/mar/0403summ.pdf>.

Bill Gates tells the Microsoft story in his 1995/6 book *The Road Ahead*, a title suggesting purpose and drive. It provides a fascinating account of the story behind Microsoft's disk operating system, MS-DOS.

The Early History of MS-DOS

From the mid-1960s IBM invested in the path-breaking concept of a scalable family of computers System/360 that would each use the same basic operating system, so applications and hardware peripherals could be easily transferred across different machines. Wanting to break into the market for small computers, IBM marketed a PC, arranged with Intel to provide the processor and invited Microsoft, who had already marketed Microsoft BASIC, to write the OS for the PC. In fact IBM offered three disk operating systems on their first PC, but the commercial strategy of Microsoft was decisive.

"We saw three ways to get MS-DOS out in front. First was to make MS-DOS the best product. Second was to help other software companies write MS-DOS-based applications software. Third was to ensure that MS-DOS would be inexpensive to licence. We gave IBM a fabulous deal – a one-time fee of about \$80,000 that granted the company the royalty-free right to use Microsoft's operating system forever... Our goal was not to make money directly from IBM but to profit from licencing MS-DOS to computer companies who wanted to offer machines more or less compatible with the IBM PC." (Bill Gates *The Roads Ahead*, 2nd ed. p.54)

Microsoft won the battle of operating systems not necessarily because their product was better than others, but because it provided the market with what it wanted and leveraged off the fact that IBM had pioneered the use of a common, albeit proprietary, operating system that opened a door to cheaper cloned models that could be mass marketed. By selling MS-DOS to the manufacturers of clones and branded but compatible PCs, Microsoft hit the jackpot, rather like selling wheel-barrows to gold miners in 1849, making money along part of the value-chain that was least vulnerable to the vagaries of final demand and theft – "one reason we wanted to sell to computer companies rather than consumers was software piracy." (p.46)

Microsoft's rise to power is one of the legends of modern industrial history. It is controversial, but there is no denying that it has set the pace for mass computing and computing for the masses. What is important now is indeed the road ahead, where does computing software go from here? The term 'pervasive computing' has been coined to describe the evolution of micro chips into a ubiquity of networked 'things' spread throughout society, from items of personal clothing to vehicles of all types, from home networking to global trace and tracking systems. To network 'things' and then to monitor, record, analyze and use the information requires operating systems and the layers of protocols and applications stacked upon them. The important issue is not in which direction does Microsoft or any other individual company

go,⁶ but what opportunities does ‘pervasive computing’ offer to software developers world wide? And does the sharing of source code offer a road ahead?

Free Software and GNU (Gnu Not Unix)

When a technology becomes so pervasive that all economic and social, political and cultural life is influenced by it, it is inevitable that society should reach the conclusion that the terms and conditions on which it can be accessed and used become issues of general concern and public policy. One manifestation of this thinking is laws governing security, anti-competitive behaviour, data privacy, content issues, etc. Another is public debate over the digital divide and affordable access. Conflicts of interests and of philosophy inevitably arise. The manifesto of the Free Software movement, for example, wants to restore what are regarded as the values of cooperation and collaboration associated with the early days of software development, reflecting the ideals of a more egalitarian society.⁷ These views are available on the website of the GNU project that began when Richard Stallman, working at the Artificial Intelligence Labs at MIT, issued the GNU Manifesto in 1983.⁸

I consider that the golden rule requires that if I like a program I must share it with other people who like it. I cannot in good conscience sign a nondisclosure agreement or a software license agreement. So that I can continue to use computers without violating my principles, I have decided to put together a sufficient body of free software so that I will be able to get along without any software that is not free.

The GNU project pioneered the General Public Licence (GPL) that enshrines the ideals of the Free Software movement.⁹ A confusion that arises is over the word ‘free’ by which the GNU project means freedom, not price. The software may be charged, but the user then becomes totally free to use it.

Free Software¹⁰

“Free software” is a matter of liberty, not price. To understand the concept, you should think of “free” as in “free speech,” not as in “free beer.”

- The freedom to run the program, for any purpose (freedom 0).

⁶ ‘One of the lessons of the computer industry – as well as life – is that it’s impossible to do everything well.’ Bill Gates, The Road Ahead, p.279.

⁷ At: <http://www.gnu.org/gnu/gnu-history.html> ‘The GNU Project was conceived in 1983 as a way of bringing back the cooperative spirit that prevailed in the computing community in earlier days – to make cooperation possible once again by removing the obstacles to cooperation imposed by the owners of proprietary software.’

⁸ <http://www.gnu.org/gnu/initial-announcement.html>

⁹ At: <http://www.gnu.org/copyleft/gpl.html> ‘The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software – to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation’s software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.)’

¹⁰ <http://www.gnu.org/philosophy/free-sw.html>

- The freedom to study how the program works, and adapt it to your needs (freedom 1). Access to the source code is a precondition for this.
- The freedom to redistribute copies so you can help your neighbor (freedom 2).
- The freedom to improve the program, and release your improvements to the public, so that the whole community benefits (freedom 3). Access to the source code is a precondition for this.

A program is free software if users have all these freedoms. Thus, you should be free to redistribute copies, either with or without modifications, either gratis or charging a fee for distribution, to anyone anywhere. Being free to do these things means (among other things) that you do not have to ask or pay for permission.

As the box makes clear, ‘free’ software refers to the freedom to use it, not necessary without charge, not as in ‘free beer’, but as in a ‘free puppy’. Downloading and installing the software is just the first step along a road that involves cost outlays.

Open Source Software

Open Source doesn’t just mean access to the source code.¹¹

The Free Software movement and the Open Source movement are today separate movements with different views and goals, although we can and do work together on some practical projects. The fundamental difference between the two movements is in their values, their ways of looking at the world. For the Open Source movement the issue of whether software should be open source is a practical question, not an ethical one.¹²

The first steps towards the commercialization of free software were business models based on value-added services, such as maintenance and support services, customization services, etc. The range of licenses offered under Open Source goes beyond pure GPL to include software with partially-restricted access to source codes and even some proprietary programs. At the margins, even Microsoft’s recent approach to source codes comes close. Most of the companies associated with the Open Source Initiative are vendors of Linux, Red Hat being among the best known. In 2004 Red Hat introduced a new business model that for the first time moves away from flat rate charges for providing support services and introduced a per-CPU fee for its commercial version of Enterprise Linux, although still claiming the fees remained service charges. Even so, this has raised charges by the Free Software movement because this places an economic restriction upon the freedom to use the software on different servers.

¹¹ <http://www.opensource.org>

¹² <http://www.gnu.org/philosophy/free-software-for-freedom.html>

Business Models for the Software Sector?

Red Hat's evolving business model is just one of many Open Source licensing schemes that offer various terms and conditions governing access to the source code.¹³ Apart from access to the code, licensing schemes may charge only for support services, or may charge according to the number of servers, or may limit the number of clients that can access a server using the software, or may specify a maximum number of total connections, or may charge per CPU, and so forth. What appears to be true of most enterprise users is that total cost of acquisition (TCA), that is to say the cost of purchase and installation, is less important than the total cost of ownership (TCO) taking into account the downtime, technical support and maintenance, upgrade costs, technician costs, and so on.

What also seems to be true is that some companies will rely initially upon proprietary software and systems for mission critical tasks and for other tasks will experiment with systems charging less in licence fees. As the industry rides these learning curves so a new balance in the business models will emerge. One possibility is that competition will move more towards ongoing services, as in the long-term contract approach, but this could lock vendors into expensive relationships and what are the possibilities of hedging against unexpected turns in technology developments? Another approach could be that vendors themselves will focus more on off-the-shelf component solutions. This is an area in which local software developers, either working closely with vendors or independently of them, could make their mark.

Future Directions

What options are open for software developers in Hong Kong? The Government runs many support programmes. IT has for a long time been a priority for universities, tertiary institutions and schools, and the sector itself has many very active industry associations working in close collaboration with their Mainland counterparts. Are these initiatives sufficient, and are they succeeding? How do we measure success? Can Hong Kong benefit from outsourcing to Mainland China or India or Philippines? Is our education system stimulating sufficient creativity and interest by young people in their ICT skills? How specialized should IT knowledge as opposed to skills training be within the public sector given the ubiquity of computing? Can Government do more as a user of ICTs? How urgent are the issues of standards and interoperability for the public sector and for the private sector? Government has moved towards harmonization of standards and interoperability within its branches and departments, but should this process of harmonization be extended across the entire public sector, such as schools, hospitals, public utilities, etc.? Would this cut costs? Would this offer greater scope for local developers to create specialist and customized enterprise software for these sectors?

We do not attempt to answer those questions in this Briefing Paper. Rather we confine ourselves to a list of suggestions where software developers in Hong Kong could expect to reap some rewards. The list is purely suggestive. Some of the industry trends that will influence future opportunities can also be identified. They would include the convergence of Information and Communication Technologies (ICTs) or Telecoms, Computers and Media

¹³ See <http://www.opensource.org/licenses> for a selection.

(TCM), broadband developments in both fixed and mobile cellular sectors and the infotainment industry, a greater stress with enterprises on total cost of ownership (TCO) and ways to simplify software packages into off-the-shelf components, security issues and digital rights management issues, the need also to customize software to local language and usage requirements, and trends in various industries such as the growing interest in RFID and trace and tracking applications in the logistics sector, and the need to optimize resource allocations in all kinds of industries, and the growing need for medical data processing and communication with the Mainland.

A List of Suggestions - Some Areas in which Hong Kong Developers Could Do Well?

1. Chinese character set software text and voice translation and recognition systems.
2. Chinese character set data mining for structured and unstructured data.
3. E-commerce, security and digital rights management (DRM) systems
4. Fixed – mobile telecoms convergence applications (P2P, P2M and M2M) such as remote monitoring.
5. Middleware for gaming, such as special effects
6. IP telecoms traffic data mining
7. Digital ink and digital paper applications
8. Xtm, metatags and cataloging
9. Optical technologies

Some useful urls of interested Industry Associations

Internet Professionals Association - <http://www.iproa.org>

Information and Software Industry Association - <http://www.isia.org.hk>

Hong Kong ASP Industry Consortium - <http://www.tdctrade.com/hksar/433.htm>

Hong Kong Computer Society - http://www.hkcs.org.hk/en_hk/home/home.asp

Hong Kong Internet Service Providers Association - <http://www.hkispas.org.hk>

Hong Kong Linux Industry Association - <http://www.hklia.org>

Hong Kong Linux Player Group - <http://www.hklpg.org/cgi-bin/index.pl>

Hong Kong Linux User group - <http://www.linux.org.hk/org/index.php?lang=en>

Hong Kong Web Hosting Association - <http://www.hkwha.org>

Hong Kong and Mainland Software Industry Cooperation Association - <http://www.hmsica.org/eng>

Pearl River Delta Software Alliance - [http://www.hkcs.org.hk/HKCS09_03\(E\)em.pdf](http://www.hkcs.org.hk/HKCS09_03(E)em.pdf)